

Toward a Container Migration Data-Auditing Mechanism for Edge Computing Environment

Toshihiro Uchibayashi[1(✉)], Bernady Apduhan[2], Takuo Suganuma[3], and Masahiro Hiji[3]

[1] Kyushu University, Fukuoka, Japan
uchibayashi.toshihiro.143@m.kyushu-u.ac.jp
[2] Kyushu Sangyo University, Fukuoka, Japan
bob@is.kyusan-u.ac.jp
[3] Tohoku University, Sendai, Japan
{suganuma,hiji}@tohoku.ac.jp

Abstract. With the widespread use of IoT sensors and devices, it has become common to store data in the cloud. The huge and various types of data with different purposes and forms are not directly stored in the cloud but are sent to the cloud via edge computing devices. Applications are running in containers and VMs to collect data from edge computing devices and sent to the cloud. However, the current deployment and migration mechanisms for containers and VMs does not consider the conventions and regulations of the applications and data they contain. The problem is that it is easy to deploy or migrate a container and VM even if the edge computing device to which it is deployed or migrated violates the licensing terms of the application it contains, the terms of the organization, or the laws and regulations of the country in which it is located. We previously proposed a data-auditing migration control mechanism for VMs. The same problem was expected with containers as the number of edges increased. Therefore, this paper proposed a data-auditing mechanism for container migration. Implementation and evaluation of the proposed system in an edge computing environment showed that adding the proposed mechanism has minimal impact on migration time and is sufficiently practical.

Keywords: Container · Edge computing · Migration · Policy · Data audit

1 Introduction

In recent years, sensors and devices have become increasingly popular. A wide range of types of sensors and devices suitable for various applications is available in the market, including sensors for measuring environmental conditions as well as wearable devices for measuring personal health conditions. The internet of

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
O. Gervasi et al. (Eds.): ICCSA 2022 Workshops, LNCS 13379, pp. 90–102, 2022.
https://doi.org/10.1007/978-3-031-10545-6_7

things (IoT), where the data collected by these sensors and devices are stored in the cloud via the edge computing devices, has become common. Web services using these data are required to handle the huge volumes of data as well as data containing personal information.

Therefore, pre-processing at the edge computing device is performed before storing the data in the cloud. The aim is to reduce the volume of data or to convert information associated with individuals into data that are not associated with the individuals through secondary processing. The edge computing device is expected to be indispensable for pre-processing huge volumes of data at high speeds in the future. Pre-processing applications running at the edge computing device need to be deployed on edge computing devices in parallel, which handle huge amounts of data. For this purpose, containers and virtual machines (VMs) need to be employed. These containers and VMs can be deployed on many edge computing devices by managing them as images. It is also easy to extract an image from an active edge computing device and deploy and migrate it to other edge computing devices. Significant research work has been conducted on these migration technologies for optimizing the network connectivity by dynamically and automatically relocating edge computing devices, and likewise considering the sensors and devices density as well as the geographic location. Some of these migration technologies have already been put into service [1–3].

However, current mechanisms for deploying and migrating VMs and containers do not consider the conventions and regulations related to the applications and data they contain. VMs or containers can be easily deployed or migrated, however the edge computing device on which they are deployed or migrated may violate the licensing terms related to the application it contains, the terms of an organization, or the laws and regulations of a particular country (Fig. 1). When VMs or containers are deployed or migrated, they must be in locations that conform to the applicable license terms and conditions or the organization's policies. The restrictions are even stricter, where national laws and regulations are applied. The administrator of an edge computing device is not necessarily the administrator of a particular application. As a result, in some cases, the terms related to this application can be violated. There is a high possibility of violating national laws and regulations such as the general data protection regulation (GDPR) [4], which is a very problematic situation.

To overcome this problem, we have proposed a live migration-control mechanism for VMs. In recent years, containers, which constitute a more lightweight and flexible environment than VM, have been attracting increasing attention. Web applications previously realized in VM are being migrated to containers, one after another. Furthermore, as edge computing becomes more pervasive, the demand for containers increases and the container migration attracts more attention. Containers are often used on edge computing devices, which are very inefficient compared to the cloud. However, the inability to handle multiple containers simultaneously limits the number of containers to be used simultaneously. In this case, the container processes must be migrated to a neighboring edge computing device. The same problem with VM is expected to occur in container, as

the number of edge computing devices increases. In this paper, we proposed a data-auditing mechanism for the migration of containers.

In Sect. 2, the related work along with our previous work is introduced. In Sect. 3, the current status of container migration is described. In Sect. 4, a data-auditing mechanism is proposed. In Sect. 5, the proposed mechanism is implemented and evaluated. Finally, Sect. 6 describe our conclusion and the future work.

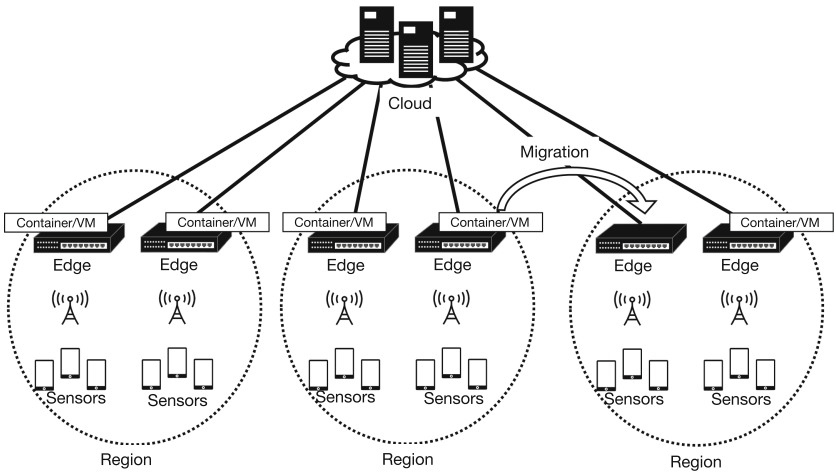

Fig. 1. Mobile sensors in edge and cloud computing environment.

2 Related Works

Shubha et al. [5] proposed a method to significantly increase the aggregation rate of data center environments by effectively managing the container state. In cloud application services, the challenge is to design efficient mechanisms, which can quickly reactivate idle containers and prevent degradation of the application's quality of service. In this paper, the proposed system is implemented on Amazon EC2, and it was confirmed that the proposed algorithm was capable of managing the container state and improving the container aggregation rate.

Omogbai [6] clarified the concept of container placement and movement on edge computing servers as well as the scheduling model and algorithm developed for this purpose. As a result, the container placement problem can be solved using a multi-objective optimization model or a graph network model and the proposed scheduling algorithm. This algorithm includes heuristic-based algorithms, which can quickly produce suboptimal solutions. It was also found that a few container-scheduling models consider distributed edge computing tasks.

Shunmugapriya et al. [7] proposed a framework, which enables a containerized evolved packet core (EPC) virtual-component migration using an open-source

migration solution. This framework also provides a comprehensive experimental analysis of a live migration for two virtualization technologies (VMs and containers) and further scrutinizes container migration approaches. In their experiments, they compared several system parameters and configurations, including the arbor (image) size, the network characteristics, the processor's hardware architecture model, and the CPU load on backhaul network components. The results showed that the proposed framework successfully reduces the live migration completion time on container platforms and the end-user service interruption time for virtualized EPC components by approximately 70%.

These studies propose solutions for the quality of service and scheduling of containers in edge computing. In other words, these algorithms can automatically control a large number of containers, making it difficult for a human to determine whether a container can be movement and placement in a specific location. In this paper's method can be used to solve this problem, allowing an automated decision to be made as to whether or not to movement and placement of containers.

3 Data-Auditing for VM Migration

VM migration in the cloud is the physical transfer of a VM running in the cloud to another host machine. There are two types of migration: "live migration" and "cold migration (offline migration)". In live migration, VMs can be migrated while processes are still running without stopping. In this way, the services in the VM can be migrated while they are still alive. The memory image of the VM running on a physical host machine is entirely transferred to a VM on a different physical host machine. The operation continues on the destination physical host machine, without stopping or disconnecting the running operating system, the application software, or network connections. Although there is strictly a millisecond-long pause during switchover, but the network session is not disconnected, and the VM user is unaware that the switchover has taken place. During the cold migration, the VM is shut down, and then it is migrated to another host machine. The VM running on the physical host machine is temporarily stopped. Its memory image is moved to another physical host machine via storage, etc., and resumes operation on the destination computer. Migration has become an indispensable technology for cloud computing operations. However, the problem is that the migration process has been simplified and can be easily performed as long as the user satisfies the cloud's permissions. In this process, only the cloud's authority and the possibility of physical migration are determined, without considering any VM internal data. This could lead to violations of the internal company rules and national laws (Fig. 2). Suppose that a VM is running on the cloud of an IaaS Cloud Provider offering global IaaS services. And, the VM that contains personal data cannot cross the country borders, and that according to internal regulations, this personal data can be stored only in Region A and Region B. There is no problem if the VM manager, who controls which region the VM is launched in, migrates the VM from Region A to Region

B. However, if the VM is migrated to Region C by the VM manager's mistake ([MISTAKE] in Fig. 2), it will be a violation of the company policy. Also, if a malicious person (VM Manager (Fake) in Fig. 2) maliciously migrates the VM to Region D in Country Y ([ATTACK] in Fig. 2), it may violate the laws of the Country X. This is a very problematic situation and requires a control mechanism capable of determining whether or not a migration is allowed, considering the permissions and physical availability of the cloud and the data contained in the VM.

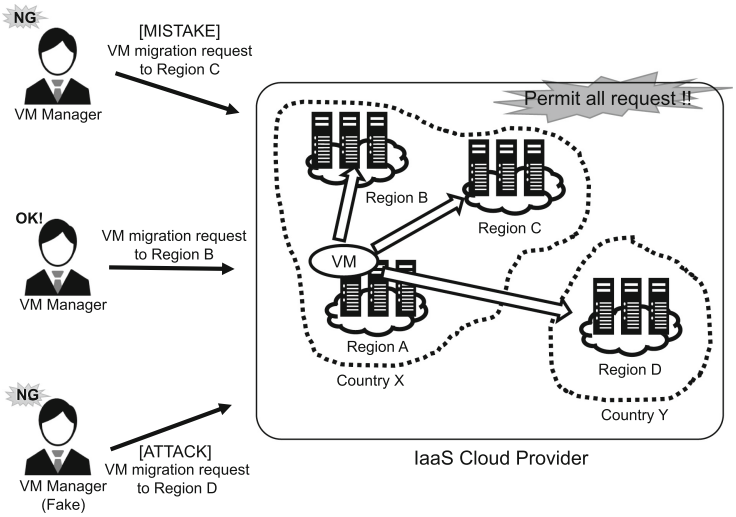

Fig. 2. VM migration problem.

To overcome the problem of an inappropriate live migration regarding data contained in VMs, we proposed a policy-based control mechanism. [8–10] The operation of the control mechanism is shown in Fig. 3. In the REGULATION, the VM administrator describes a list of identifiers of countries and organizations to which the VM data can be moved, according to the regulations attached to the data. The CountryCode indicates the countries where data movement is permitted (according to the regulations), and the OrganizationCode represents the organization that instantiates and uses the VM. When the migration is executed, the REGULATION of the VM to be moved is compared with the COUNTRY and REGULATION of the destination host machine. First, it is checked whether the CountryCode of the REGULATION is included in the CountryCode of the destination host machine.

Similarly, it is checked whether the OrganizationCode of REGULATION is included in the OrganizationCode of the destination host machine. If both checks pass, the data acquired by the VM to be moved can be moved to the country or organization where the destination host machine exists. In this way, the migration is executed. This migration process is the same as the traditional migration

process. If both checks fail, the migration is not performed because the VM cannot be moved to the country where the destination host machine is located. Using these mechanisms, unintentional data breaches (caused by data migration) of the terms and conditions, which are granted by different owners of various data, national laws and regulations, and organizational rules, are avoided. In addition, by managing the policies in blockchain, and if the REGULATION, COUNTRY, or ORGANIZATION is tampered with by a malicious administrator on a host machine running various VMs, an involuntary migration can be triggered, and an inappropriate movement may occur. Therefore, to prevent malicious data tampering, a method for preventing tampering using blockchain technology was proposed. This method features a robust data protection mechanism.

The proposed mechanism was implemented in a cloud environment by registering the policies regarding VMs and host machines in the blockchain, and the decision of allowing or disallowing a VM live migration in compliance with the policies was evaluated. The execution time of the live migration using the proposed mechanism was also measured, and no overhead cost was observed.

This paper extends our proposed mechanism targeting to container in the edge. In the next section, we introduce the state of the art containers and container migration.

4 Containers and Container Migration

There are several container platforms that implement container migration.

Docker [11] is the de facto standard for containers platform. In 2016, Docker's announcement of the live migration of containers sparked much attention. Since the live migration allows containers to be migrated while processing continues, it has become an essential technology in IoT environments, where edge computing will be the de facto standard in the future. However, Docker still does not offer a pure live migration. It supported migration by Checkpoint/Restore In Userspace (CRIU).

Linux Container (LXC) [12] is a system container, which has replaced KVM and Xen. Different for LXC from Docker, which is intended to run lightweight virtual machines, the image it boots is a full OS image, complete with init and systemd. It supported migration by CRIU and live migration is implemented natively in LXD.

OpenVZ [13] runs on a single kernel, which is shared by multiple VMs on the same machine, and it has the ability to run on multiple Linux systems such as Fedora Core, CentOS, SUSE Linux, and Debian GNU/Linux. Migration functionality is available for kernels 2.6.9 or higher. It supported migration and live migration.

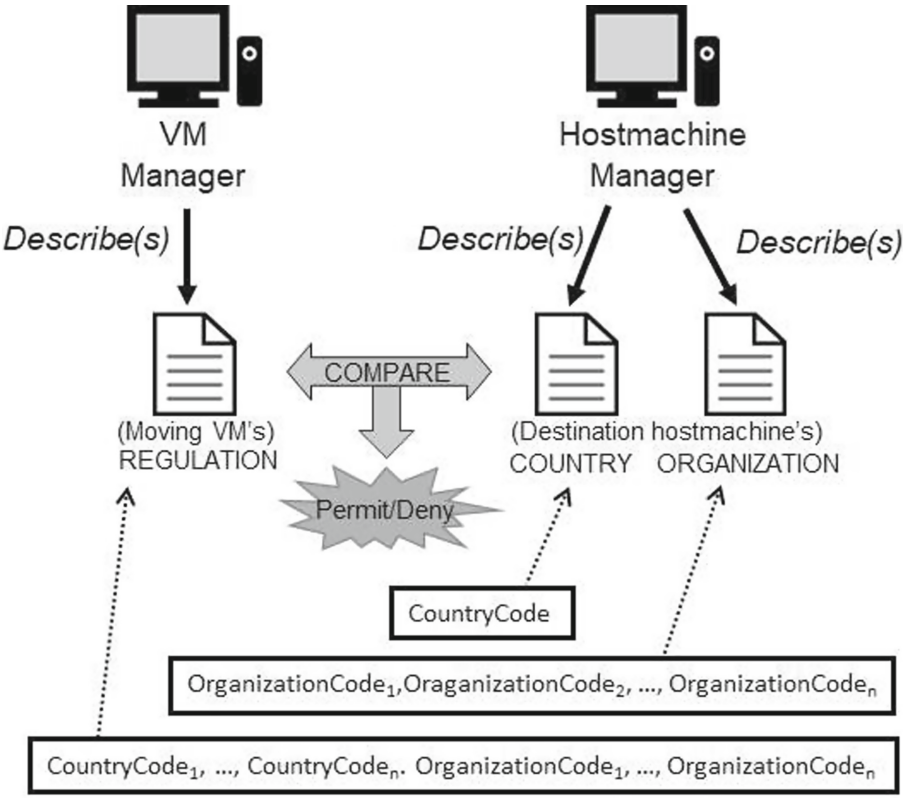

Fig. 3. The data-auditing mechanism for the live migration of VMs

5 Proposed Data-Auditing Policy Mechanism

We considered that improper movement of data is a concern in container migration like VM migration. Therefore, we propose a data-auditing mechanism and its policy during migration targeting containers by extending the policy of the data auditing mechanism for VM. In difference to the VM's host machine, the edge computing device does not have huge resources, and hardware requirements are needed in addition to information such as organization and country. Therefore, our proposed mechanism adds hardware requirements in addition to data audit requirements such as organization and country. A data-auditing operation checks whether it is acceptable to migrate data in the software. A hardware-requirement operation checks whether it is acceptable to migrate data in the hardware. These policies are stored within the container at the edge computing device. The container administrator writes and stores the container, according to the container's application and data conventions within the container. At the edge computing device, the edge computing device administrator describes and stores information similar to that of the edge computing device (Fig. 4). Usually,

these written policies are assumed to be unchanged. During container migration, the container policy in the container to be migrated is compared with the policy of the destination edge computing device to determine whether migration is permitted or not. If the destination edge computing device satisfies the requirements of the policy in the container, the migration will be executed.

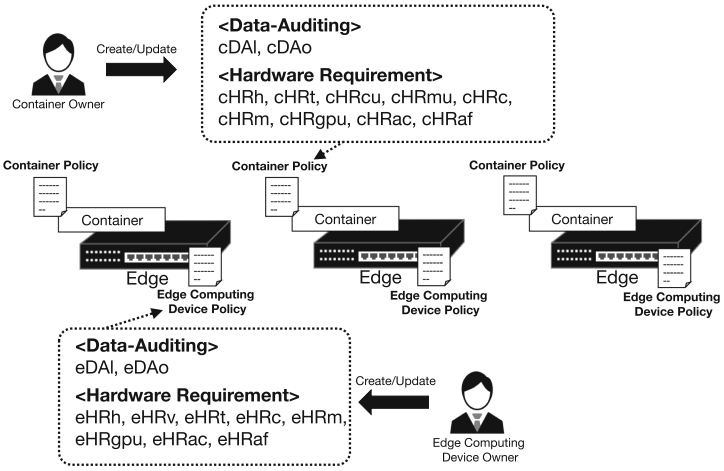

Fig. 4. Container and edge computing device policy allocation.

5.1 Container Policy

This policy establishes rules for determining whether the container created/updated by container owner can be migrated or not. Table 1 shows the elements and description of Data-Auditing in container policy. It has two Data-Auditing policies and nine Hardware Requirement policies. cDAl describes the countries or regions where containers can be placed. cDAo describes the platforms allowed to deploy containers, according to the conventions of the organization managing the containers. Table 2 shows the elements and description of Hardware Requirements in container policy. cHRh describes the allowed edge computing devices; cHRt describes the virtualization tool used by the container; cHRcu describes the CPU utilization threshold of the destination edge computing device; cHRmu describes the memory utilization threshold of the destination edge computing device; cHRc describes the minimum CPU frequency threshold of the destination edge computing device; cHRm describes the minimum memory capacity threshold of the destination edge computing device; cHRgpu describes whether the destination edge computing device must support AI processing; cHRac describes the AI accessor required at the destination edge computing device; cHRb describes the minimum memory capacity threshold of the destination edge computing device; cHRc describes the minimum memory capacity

threshold of the destination edge computing device; cHRaf describes the AI framework required at the destination edge computing device.

Table 1. Data-auditing in container policy.

Element	Description
cDAl	Countries or regions that are allowed
cDAo	Administrative organizations that are allowed

Table 2. Hardware requirements in container policy.

Element	Description
cHRh	Name of edge computing device manufacturers that are allowed
cHRt	Virtualization tools to be used
cHRcu	CPU utilization at the destination edge computing device (%)
cHRmu	Memory utilization at the destination edge computing device (%)
cHRc	Min CPU frequency at the destination edge computing device (GHz)
cHRm	Min memory capacity at the destination edge computing device (GB)
cHRgpu	AI support availability the destination edge computing device
cHRac	AI accessor required the destination edge computing device
cHRaf	AI framework required the destination edge computing

5.2 Edge Computing Device Policy

This policy establishes the rules for determining whether the edge computing device which was created/updated by edge computing device owner can be migrated or not. Table 3 shows the elements and description of Data-Auditing in edge computing device policy. It has two Data-Auditing policies and eight Hardware Requirement policies. eDAl describes the country or region where the container is located; eDAo describes the organization managing the container; Table 4 shows the elements and description of Hardware Requirements in edge computing device policy. eHRh describes the manufacturer of the edge computing device; eHRv describes the available edge computing environment (VM or container); eHRt describes the name of the virtualization tool that can be used; eHRc describes the minimum CPU frequency; eHRm describes the edge computing device memory capacity threshold; eHRgpu describes whether AI processing is supported; eHRac describes the available AI accessors; eHRaf describes the available AI frameworks.

Table 3. Data-auditing in edge computing device policy.

Element	Description
eDAl	Country of placement
eDAo	Managing organization

Table 4. Hardware requirements in edge computing device policy.

Element	Description
eHRh	Edge computing device manufacturer name
eHRv	Available environment
eHRt	Available virtualization tools
eHRc	CPU frequency (GHz)
eHRm	Memory capacity (GB)
eHRgpu	AI support
eHRac	Available AI accessors
eHRaf	Available AI frameworks

6 Implementation

In this Section, an edge computing environment consisting of multiple edge computing devices, which are virtually divided into multiple regions, is implemented. All edge computing devices are assumed to run LXC and container-generated applications. In addition, since CRIU is deployed at all edge computing devices, the containers can be migrated among the edge computing devices. Furthermore, the components required to realize the proposed mechanism are assumed to be applied at all edge computing devices. The implementation configuration is shown in Fig. 5.

The implemented edge computing environment consists of eight edge computing devices and four regions. Edge 1 has a container running an application. These edge computing devices are running in an ESXi-6.5.0, which is a high-performance server. The server consists of 20 CPUs (Intel(R) Xeon(R) CPU E5-2660 v3 @ 2.60 GHz, 128 GB memory) and 3.45 TB HDD storage. All edge computing devices are configured with 2 vCPUs, 8 GB memory, 80 GB storage and Ubuntu 20.04 LTS OS. LXC and CRIU are installed and ready to run containers. The container policy for App1 which is running on edge 1 is shown in Table 5. The edge computing device policies are shown in Table 6. These policies are stored on the edge computing device for each row. App1 allows containers to be placed in JP and US. Additionally, it allows containers to be deployed on edge computing devices managed by org1, org3, and org4. This means that Application1 cannot be migrated to Edge 3, Edge 4, Edge 7 and Edge 8.

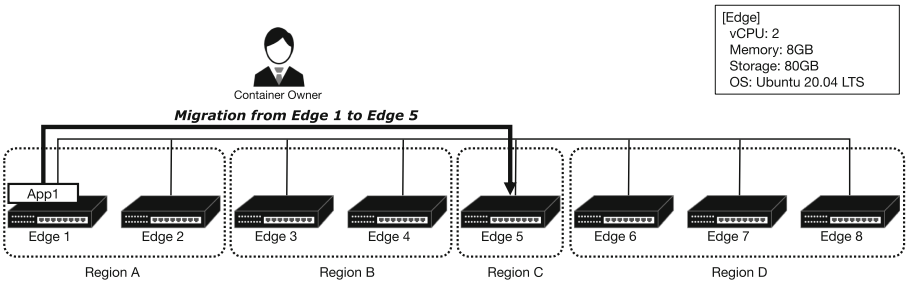

Fig. 5. Edge computing environment implementation configuration.

The execution time is measured when the container is migrated from Edge 1 to Edge 5. The average execution time is the result from ten runs for same migration condition. The difference in execution time with and without the proposed data-auditing mechanism is clearly observed. The case with the data-auditing mechanism is 220 ms. The case without the data-auditing mechanism is 217 ms. The case with the policy mechanism is slower than the case without the policy mechanism. Since the effect of the policy mechanism on the migration time is very minimal. We can say that the proposed data-auditing mechanism is sufficiently practical.

Table 5. Container attributes running on edge 1.

cDAl	cDAo	cHRh	cHRt	cHRcu	cHRmu	cHRc	cHRm	cHRgpu	cHRac	cHRaf
jp us	org1 org3 org4	app1	lcx	40	40	1.6	4	na	na	na

Table 6. Edge device attributes.

	eDAl	eDAo	eHRh	eHRv	eHRt	eHRc	eHRm	eHRgpu	eHRac	eHRaf
Edge 1	jp	org1	edge1	ctr	lxc	2.6	8	na	na	na
Edge 2	jp	org1	edge2	ctr	lxc	2.6	8	na	na	na
Edge 3	de	org2	edge3	ctr	lxc	2.6	8	na	na	na
Edge 4	de	org2	edge4	ctr	lxc	2.6	8	na	na	na
Edge 5	jp	org3	edge5	ctr	lxc	2.6	8	na	na	na
Edge 6	us	org4	edge6	ctr	lxc	2.6	8	na	na	na
Edge 7	us	org5	edge7	ctr	lxc	2.6	8	na	na	na
Edge 8	us	org5	edge8	ctr	lxc	2.6	8	na	na	na

7 Conclusion

In this paper, we focused on containers and container migration in edge computing environments, which have been in a rapidly growing demand in recent years. We proposed a data-auditing mechanism for container migration. Extended policies for VM and created policies for container in the edge. The evaluation of the proposed system in an implemented edge computing environment showed that the addition of the proposed policy mechanism has minimal impact on the migration time. We believe that using the proposed mechanism will reduce incorrect container movement and placement of containers. Therefore, the proposed policy mechanism is sufficiently practical. In the future, we aim to extend our investigation from the very compact and short-range environment examined to a real network extending over a wide area.

Acknowledgement. This work was supported by JSPS KAKENHI Grant Number JP20K19778.

References

1. Puliafito, C., Vallati, C., Mingozzi, E., Merlino, G., Longo, F., Puliafito, A.: Container migration in the fog: a performance evaluation. Sensors **19**(7), 1488 (2019)
2. Deshpande, L., Liu, K.: Edge computing embedded platform with container migration. In: 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), pp.1–6 (2017)
3. Karhula, P., Janak, J., Schulzrinne, H.: Checkpointing and migration of IoT edge functions. In: Proceedings of the 2nd International Workshop on Edge Systems, Analytics and Networking (EdgeSys 2019), pp. 60–65 (2019)
4. General Data Protection Regulation, GDPR. Accessed April 2022. https://eur-lex.europa.eu/eli/reg/2016/679/oj
5. Nath, S.B., Addya, S.K., Chakraborty, S., Ghosh, S.K.: Container-based service state management in cloud computing. In: 2021 IFIP/IEEE International Symposium on Integrated Network Management (IM), pp. 487–493 (2021)
6. Oleghe, O.: Container placement and migration in edge computing: concept and scheduling models. IEEE Access **9**, 68028–68043 (2021)
7. Ramanathan, S., Kondepu, K., Razo, M., Tacca, M., Valcarenghi, L., Fumagalli, A.: Live migration of virtual machine and container based mobile core network components: a comprehensive study. IEEE Access **9**, 105082–105100 (2021)
8. Uchibayashi, T., et al.: A control mechanism for live migration with data regulations preservation. In: Gervasi, O., et al. (eds.) ICCSA 2017. LNCS, vol. 10404, pp. 509–522. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-62392-4_37
9. Uchibayashi, T., Apduhan, B.O., Shiratori, N., Suganuma, T., Hiji, M.: Policy management technique using blockchain for cloud VM migration. In: IEEE International Conference on Dependable, Autonomic and Secure Computing, International Conference on Pervasive Intelligence and Computing, International Conference on Cloud and Big Data Computing, International Conference on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), pp. 360–362 (2019)

10. Uchibayashi, T., Apduhan, B., Suganuma, T., Hiji, M.: A cloud VM migration control mechanism using blockchain. In: Gervasi, O., et al. (eds.) ICCSA 2020. LNCS, vol. 12252, pp. 221–235. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58811-3_16
11. Home - Docker. Accessed April 2022. https://www.docker.com/
12. Middelkamp, A.: Online. Praktische Huisartsgeneeskunde **3**(4), 3–3 (2017). https://doi.org/10.1007/s41045-017-0040-y
13. Open source container-based virtualization for Linux. Accessed April 2022. https://openvz.org/