Design and Implementation of a Software Disaster Recovery Service for Cloud Computing-Based Aerospace Ground Systems

Xiao Yu✉
Beijing Institute of Tracking and Telecommunications Technology
Beijing China
yuxiao801026@126.com

Dong Wang
Beijing Institute of Tracking and Telecommunications Technology
Beijing China
529492845@qq.com

Xiaojuan Sun
Aerospace Information Research Institute
Chinese Academy of Sciences

Key Laboratory of Technology in Geo-Spatial Information Processing and Application System
Chinese Academy of Sciences

School of Electronic, Electrical and Communication Engineering
University of Chinese Academy of Sciences
Beijing China
xjsun@mail.ie.ac.cn

Bingbing Zheng
Qilu Research Institute,
Aerospace Information Research Institute
Chinese Academy of Sciences

Key Laboratory of Technology in Geo-Spatial Information Processing and Application System
Chinese Academy of Sciences
Jinan China
zhengbb@aircas.ac.cn

Yankai Du
Beijing Aerospace Titan Technology Co. LTD
Beijing China
du_yankai@163.com

2022 11th International Conference on Communications, Circuits and Systems (ICCCAS) | 978-1-6654-6989-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCCAS55266.2022.9825253

***Abstract*—The data centers of cloud computing-based aerospace ground systems and the businesses running on them are extremely vulnerable to man-made disasters, emergencies, and other disasters, which means security is seriously threatened. Thus, cloud centers need to provide effective disaster recovery services for software and data. However, the disaster recovery methods for current cloud centers of aerospace ground systems have long been in arrears, and the disaster tolerance and anti-destruction capability are weak. Aiming at the above problems, in this paper we design a disaster recovery service for aerospace ground systems based on cloud computing. On account of the software warehouse, this service adopts the main standby mode to achieve the backup, local disaster recovery, and remote disaster recovery of software and data. As a result, this service can timely response to the disasters, ensure the continuous running of businesses, and improve the disaster tolerance and anti-destruction capability of aerospace ground systems. Extensive simulation experiments validate the effectiveness of the disaster recovery service proposed in this paper.**

Keywords—disaster recovery service mode, aerospace ground systems, cloud computing

I. INTRODUCTION

With the development of cloud computing technology, moving business systems to clouds has become the conventional mode of most user departments. Accordingly, aerospace ground systems have been gradually deployed on clouds. With the increase of the number of users, the software applications and data scales of aerospace ground systems are growing geometrically, and the concentrations of important software systems are becoming higher and higher. As a result, the demand for the continuous running of cloud centers is also increasing. In this situation, for the aerospace ground systems based on cloud computing, how to achieve the disaster recovery backup of software and data in a standardized and resource optimized way, and how to ensure system security, have become an important problem that must be considered and faced by the current development of aerospace ground systems.

In recent years, the researches on system disaster recovery problems are more and more in-depth. Hayes et al. simulate the impact of disasters on system data and propose a system importance ranking method based on the AHP model in [1]. Reference [2] create a metric and give different ranks for the applications in data centers to classify different disaster recovery levels, including application-level, database-level, host-level, and so on. Li et al. propose a cost-effective data reliability management mechanism based on the generalized data reliability model for disaster recovery of single cloud architecture in [3]. This mechanism proactively checks the

replicas to ensure data reliability. Alshammari et al. design a preventive data recovery method to minimize the number of replicas before disasters occur. This method ensures the high reliability of data in [4]. Hamadah et al. propose a model for the disaster recovery on clouds based on recovery time objectives and recovery point objectives to ensure the security and business continuity in [5]. Challagidad et al. propose an enriched genetic algorithm that uses four cloud backup servers to recover the lost data. When the main server loses data and cannot provide data to users, this algorithm provides users with the flexibility to collect information from any standby server in [6]. Yu et al. design a virtual network storage model and a disaster recovery architecture based on this model, according to the requirements of the disaster recovery systems in [7]. This architecture designs different virtual memory mapping mechanisms and cache mechanisms and utilizes the virtual disk transparent encryption mechanism to transparently protect backup data. However, the above studies mainly focus on the backup and recovery of data. Their main goal is to ensure the data integrity and reliability. Therefore, the post-disaster reconstruction and recovery of software applications cannot be achieved.

In current cloud environments, there are few studies that focus on software-level backup and recovery services. Reference [8] build a host-based disaster recovery solution based on I2Map, which recovers VM instances by maintaining the records of instance modifications. Wang et al. analyze the applicable scenarios of infrastructure-level and application-level disaster recovery, and design infrastructure-level and application-level disaster recovery strategies for the applications hosted in cloud centers in [9]. Reference [10] design a redundancy method, system, and equipment to obtain the description information of cloud applications that need redundancy, so as to recover the cloud applications at the disaster recovery site. However, all these studies and designs cannot well solve the problem of reconstructing software locally or remotely after disasters.

In view of the above shortcomings, according to the in-depth research on the demand of software and data disaster recovery of aerospace systems, this paper truly realizes the software disaster recovery mode of aerospace ground systems based on cloud computing. Concretely, this paper designs and implements a cloud disaster recovery service which adopts the main and standby mode to set up a production center and a disaster recovery center. By integrating the resources of data centers, based on the software warehouse, this service realizes the local and remote disaster recovery of software. When disasters occur, this service can respond in time and ensure the continuity of businesses, which improves the security of the whole cloud system. Extensive simulation experiments demonstrate the effectiveness of the cloud disaster recovery service proposed in this paper.

II. Design Ideas

Aiming at the business requirements of aerospace ground systems based on cloud computing, including intensive computing, strict timeliness, and high availability, three demands are put forward for the cloud disaster recovery service mode of cloud-based aerospace ground systems.

Firstly, the demand for application remote backup and version synchronization. Considering that the application software will continue to undergo debugging, modifications, and version update iterations during the running in the production environment, the cloud-based aerospace software disaster recovery services need to be able to perceive the version changes of the application software and automatically synchronize the latest version of the application data between the main and standby nodes.

Secondly, the demand for the efficient perceiving and accurate identification of faults. Cloud platforms need to be able to collect the information of running states, environmental information, and resource usages of business software in real-time, obtain the symptoms of software faults at the first time, accurately identify whether the application is running normally, and give timely alarms for the faulted applications.

Thirdly, the demand for automatic disaster recovery and no awareness of upstream businesses. Considering the extensive dependences and calling relationships among businesses, cloud disaster recovery services need to ensure that the access portals of business system applications that are recovered locally/remotely remain unchanged, so as to make upstream businesses be unaware of the disaster recovery actions as far as possible and avoid the chain reactions caused by specific software faults.

In view of the above demands, this paper proposes a disaster recovery service for cloud-based aerospace ground systems. This service integrates and schedules various resource management and data backup synchronization capacities of cloud platforms to realize the local and remote software disaster recovery demands. The main design ideas are shown in the following figure.

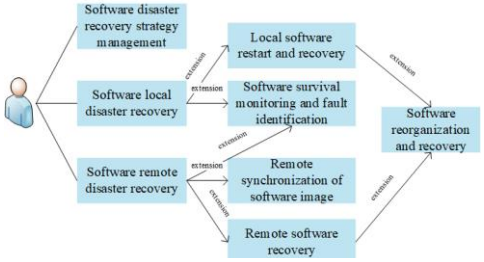

Fig. 1. The design ideas of the software disaster recovery service for cloud computing-based aerospace ground systems

As shown in Fig. 1, the software disaster recovery service for aerospace ground systems based on cloud computing needs to support the functions of disaster recovery strategy management, software local disaster recovery, and software remote disaster recovery. Among them, the software recovery strategy management function is responsible for supporting administrators to manage and configure the recovery strategies of cloud software through visual interface or OpenAPI. The software local disaster recovery function is responsible for trying to restart or reconstruct the application at the local location where business systems are deployed when software faults occur. If the application cannot be successfully recovered locally, the remote disaster recovery function is responsible for

reconstructing the application through remote backup data and software deployment packages.

To achieve the above functions, our software disaster recovery service for aerospace ground systems based on cloud computing is composed of a software image synchronization module, a fault identification module, and a reorganization recovery module. The software image synchronization module is responsible for synchronizing software images, deployment packages, and other data among cloud nodes, according to the disaster recovery strategy. The fault identification module is responsible for monitoring the software states of the cloud-based aerospace ground system in real-time, so as to timely initiate fault alarm and trigger the execution of disaster recovery strategy as required. The reorganization recovery module is responsible for recovering the specified application of the aerospace ground system at the specified location as required to realize the application recovery and business switching.

III. A Disaster Recovery Service

A. Overall Architecture

The software disaster recovery service for cloud-based aerospace ground systems proposed in this paper mainly consists of an image synchronization module, a fault identification module, and a reorganization recovery module. The overall architecture is shown as follows.

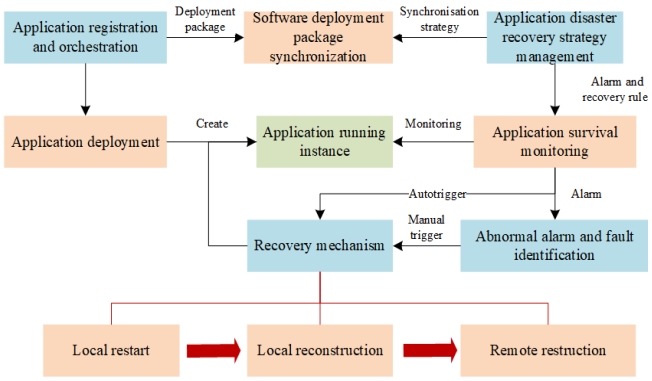

Fig. 2. The overall architecture design of the software disaster recovery service for aerospace ground systems

As shown in the Fig. 2, the software image synchronization module is responsible for synchronizing the software deployment packages created in the application registration and orchestration phases between the main and standby nodes. The application instances deployed by the main node will be monitored and identified in real-time by the application survival monitoring submodule. The software applications which are identified as faulted will automatically or manually trigger the alarm and recovery processes. The system will try to recover the business systems according to multiple steps such as local restart, local reconstruction, and remote reorganization.

B. Software Image Synchronization Module

The software image synchronization module is responsible for synchronizing the software images, deployment packages, and other data among cloud nodes, according to the disaster recovery strategy. This module also supports automatic synchronization of the latest files to the disaster recovery location when the software versions are updated.

According to the difference of software deployment modes and runtime environments, the categories of software images and related data that need to be synchronized are also different, as shown in Fig. 3. According to the running environments, software can be divided into virtual host deployment, bare metal instance deployment, and container deployment. According to the deployment forms, software can be divided into deployment package (or script) deployment and image deployment.

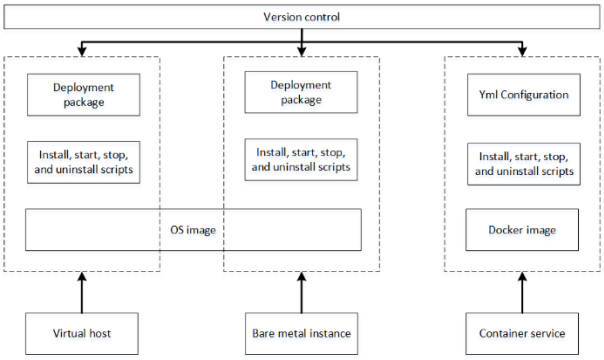

Fig. 3. The software deployment forms of aerospace ground systems based on cloud computing

Considering the data synchronization and version control requirements of each running environment and deployment form, the main object for synchronization in the software image synchronization module is images. An image is designed as an interface, which is the abstraction of the application and script configuration set required by a software deployment unit to perform deployment tasks. According to the runtime environments and deployment forms, there are many different implementations of images, such as OS image files commonly used by virtual host or bare metal instance deployment, docker image files commonly used by container service deployment, or sets of application deployment packages and deployment scripts. Based on the software disaster recovery strategy configuration, the disaster recovery service for cloud-based aerospace ground systems will automatically generate several software deployment unit synchronization tasks, while each software deployment unit synchronization task is composed of one or more image synchronization tasks. At the same time, each image synchronization task consists of an image interface instance, a source, a destination, and a synchronization mode. The image synchronization tasks are the main task objects for synchronization.

C. Fault Identification Module

The fault identification module is responsible for the real-time monitoring of software's running and survival states, so as to timely identify faults, initiate alarms, and trigger the execution of subsequent processing on demand. Concretely, the fault identification module collects the health information of software applications and identifies whether there are software faults. When a fault is found, this module gives an alarm in time and triggers the subsequent processing of disaster recovery.

As shown in the Fig. 4, the software fault identification module mainly consists of three submodules:

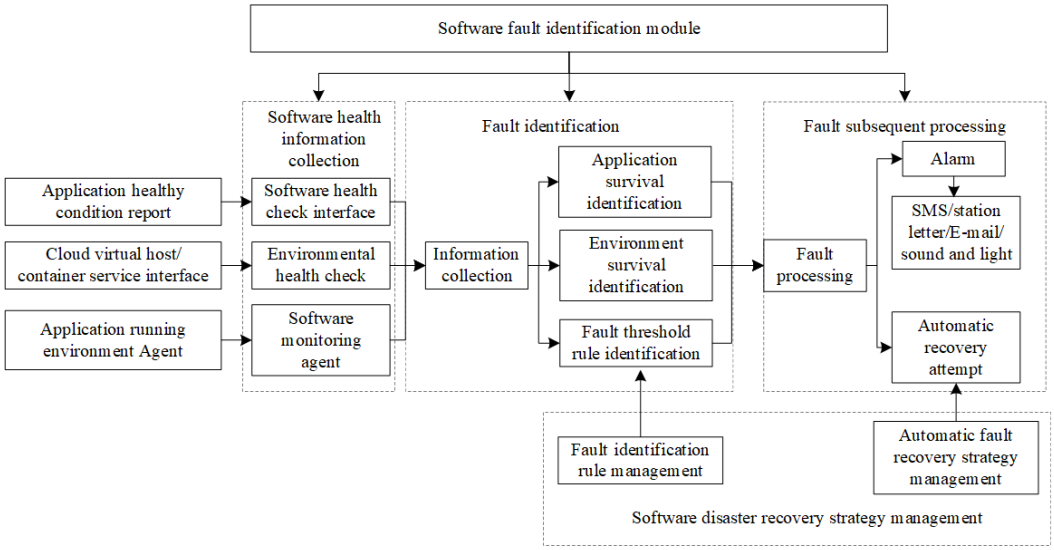

Fig. 4. The software deployment forms of aerospace ground systems based on cloud computing

1) The software health information collection submodule is responsible for collecting, summarizing and reporting the software monitoring information from multiple different ways.

2) The fault identification submodule is responsible for intelligently identifying software's health state, according to the software health information and the fault identification rules configured in the software disaster recovery strategy. Alarms and other subsequent processes are triggered if faults are found.

3) The fault subsequent processing submodule is responsible for receiving the fault information reported by the fault identification submodule, triggering a variety of fault alarms according to the configuration, and triggering the execution of the subsequent automatic fault recovery and reconstruction.

In terms of fault information collection, the fault identification module needs to support the application health check, environmental health check, and monitoring agent service, so as to summarize multi-source monitoring data and make comprehensive health monitoring identifications. In terms of fault identification, the module should support the configuration of a variety of fault identification rules, including timeout time, retry times, CPU/memory threshold, exception level, number of exceptions, etc. In terms of faults' subsequent processing, the module should support the trigger of alarm and automatic recovery attempts. The alarm methods include in-station letters, short messages, e-mail, as well as sound and light, etc.

D. Reorganization Recovery Module

The reorganization recovery module is responsible for recovering the specified version of the specified cloud software at the specified location as required and redirecting the upstream data flows and data access portals, so as to realize the application system recovery and businesses' seamless switching. This module needs to support the automatic deployment and startup of software applications, and support the configuration of cloud product service information such as domain name server (DNS), server load balancer (SLB), and elastic IP (EIP) after deployments, so as to realize the redirection of upstream access portals. Then the upstream businesses will continually access without awareness after business reorganization recovery.

In terms of recovery methods, the reorganization recovery module needs to support multi-type multi-level recovery mechanisms such as local restart, local reconstruction, and remote reconstruction. In terms of recovering business access, the reorganization recovery module needs to support redirecting upstream request portals through DNS, SLB, EIP, and other forms, so as to continue the business process.

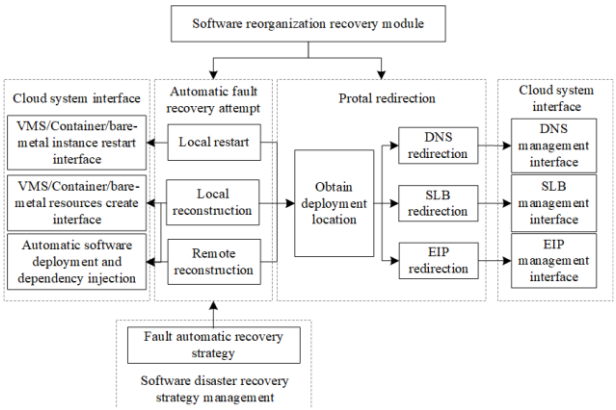

Fig. 5. The functional composition of the reorganization recovery module

As shown in Fig. 5, the software reorganization recovery module mainly consists of two submodules: the automatic fault recovery submodule and the access portal redirection submodule.

The automatic fault recovery submodule is responsible for trying several recovery methods in order according to the disaster recovery strategy. The recovery methods include local restart, local reconstruction, and remote reconstruction. After the automatic software fault recovery, the access portal redirection submodule is responsible for redirecting the

upstream business requests to the recovered application, so as to ensure that upstream callers can continuously access the application software without any changes. This submodule can ensure the continuity of businesses. Concrete redirection methods include DNS, SLB, EIP, etc.

IV. EXPERIMENTS

In order to evaluate the fault recovery capability and recovery time objective (RTO) under various cloud software deployment forms and disaster recovery modes, we design a group of experiments to simulate software faults in various scenarios and test the business recovery capability and recovery time. These experiments are used to evaluate the fault discovery, recovery capability and RTO of the software disaster recovery service designed for the aerospace ground system based on cloud computing.

A. Experimental Environment

The experiments are built based on the ground system's project demonstration and verification environment, which is deployed on the proprietary cloud of Alibaba cloud platform. The proprietary cloud governs 175 servers and 35 network switching devices. It provides computing resource services (such as virtual machine, container, and bare metal), storage resource services (such as block storage, object storage, and file storage), network resource services (such as virtual private network, load balancing, and domain name resolution services), as well as comprehensive cloud products (such as database, big data, and middleware). The software warehouse and software disaster recovery service itself occupy 3 VMs (4 core CPU and 16g memory) for deploying relevant interfaces, applications, and meta databases.

B. Experimental Process and Results

Taking the professional data processing software of ground systems as an example, the experiments fully verify its ability to access the software warehouse, automatically synchronize data between the main and standby nodes, monitor and identify software faults in time, as well as automatically reorganize and recover. The experimental process is shown in Fig. 6.

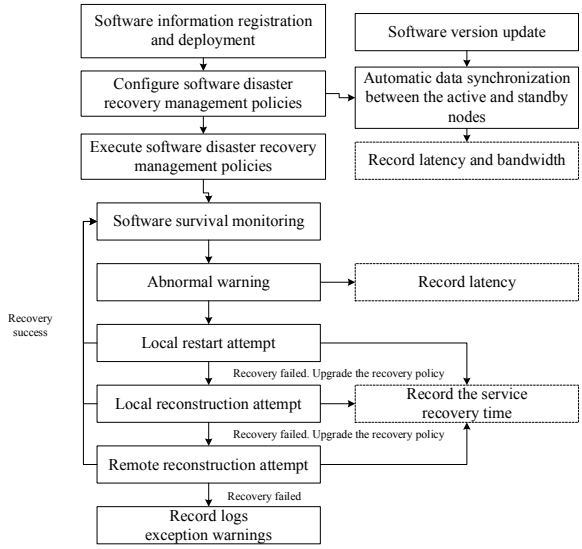

Fig. 6. Experimental process

1) Register the professional data processing software to the software warehouse of the cloud platform. Specify node A as the backup main center and node B as the remote disaster recovery center.

2) Upload the software deployment packages, scripts, and configuration files of the professional data processing software to the software warehouse. Observe the synchronization of software deployment files between nodes A and B, and record the synchronization bandwidth and time delay.

3) Update the professional data processing software from version 1.0 to version 2.0, and upload the updated deployment files. Observe the synchronization of software deployment files between nodes A and B, and record the synchronization bandwidth and time delay.

4) Configure the disaster recovery strategy of the professional data processing software, including the local restart, local reconstruction, and remote reconstruction), etc.

5) Make faults, i.e., simulate the faults of professional data processing software on node A. Observe the survival monitoring and alarm of the software warehouse, and record the alarm delay.

6) Observe the execution of the software local restart and record the recovery time.

7) Make faults to simulate the scenario of local restart failure. Observe the execution of the software local reconstruction and record the recovery time.

8) Make faults to simulate the scenario of local reconstruction failure. Observe the execution of the software remote reconstruction and record the recovery time.

9) Repeat the above test steps for cloud business application software deployed on virtual machines, bare metal instances, and containers respectively, and record the results.

According to the above experimental process, we test the professional data processing software deployed on virtual machines, bare metal instances, and containers for many times and record the average results, as shown in Table I.

TABLE I. EXPERIMENTAL RESULTS

	Virtual machine	**Bare metal instances**	**Container service**
scale of deployment unit	15	15	15
time delay of fault monitoring	0.47ms	0.32ms	0.35ms
bandwidth of data synchronization	113MB/s	110MB/s	82MB/s
recovery time of local restart	37s	112s	1s
recovery time of local reconstruction	68s	207s	4s
recovery time of remote reconstruction	66s	211s	5s

C. Analysis of Experimental Results

According to the experimental results, the software disaster recovery service designed in this paper can effectively identify software faults. The fault discovery time of the application deployed on virtual machines, bare metal instances, and

containers is sub-second level. The recovery time of local restart is basically close to the OS restart time of the application running environment. According to different configurations, the recovery time of the software deployed on the container is the shortest, followed by the virtual machine, and the bare metal instance is the longest. Since the data update and synchronization between the main and standby nodes are completed in advance, the time of local reconstruction and remote reconstruction is basically the same, which is similar to the corresponding resource distribution time. Therefore, it can be considered that the software disaster recovery service does not cause obvious time overhead except resource distribution.

Through the experiments we can get the following conclusions: 1) the professional data processing software can be registered to the software warehouse successfully; 2) the disaster recovery service can automatically synchronize the deployment packages and updated versions between the main and standby nodes according to the disaster recovery strategy; 3) the disaster recovery service can also timely find the faults and damages of software, and try to automatically recover according to the disaster recovery strategy. Therefore, the disaster recovery service proposed in this paper can effectively improve the stability and high availability of aerospace business software.

V. CONCLUSIONS

With the development of cloud computing and related technologies, aerospace ground systems based on cloud computing will become an important method for aerospace businesses to make full use of and share manufacturing resources and enhance comprehensive competitiveness. In order to promote the cloud application services of aerospace systems, this paper establishes a disaster recovery service center for cloud-based aerospace ground systems, expounds three aerospace-oriented cloud disaster recovery service modes, and analyzes the development and implementation of application experiments for the cloud disaster recovery service in detail. Through the diffusion, penetration and integration of cloud ideas in traditional aerospace systems, as well as the virtualization, aggregation and integration of various resources, the integration and scheduling of various resource management and data synchronous backup capabilities of aerospace ground system cloud platforms, as well as software's local and remote disaster recovery requirements are gradually realized. What's more, the core competitiveness and sustainable development capacity of aerospace engineering are also enhanced. Thus, cloud computing has broad application prospects in aerospace systems.

REFERENCES

[1] P. E. Hayes, A. Hammons. "Disaster recovery project management," Industry Applications Conference, 2000. Conference Record of the 2000 IEEE. IEEE, 2000.

[2] T. Zhu, Y. Xie, Y. Song, W. Zhang, K. Zhang and F. Gao, "IT Disaster Tolerance and Application Classification for Data Centers," International Congress of Information and Communication Technology (ICICT 2017). pp. 341 – 346, 2017.

[3] W. Li, Y. Yun, and Y. Dong. "Ensuring Cloud Data Reliability with Minimum Replication by Proactive Replica Checking," IEEE Transactions on Computers. Vol. 65, no. 5, pp. 1494-1506, 2016.

[4] M. Alshammari, A. Alwan, A. Nordin and A. Abualkishik, "Disaster Recovery with Minimum Replica Plan for Reliability Checking in Multi-Cloud," the 9th International Conference on Ambient Systems, Networks and Technologies (ANT). pp. 247–254, 2018.

[5] S. Hamadah and D. Aqel, "A Proposed Virtual Private Cloud-Based Disaster Recovery Strategy," IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), pp. 469-473, 2019.

[6] S. Challagidad, S. Dalawai and N. Birje, "Efficient and Reliable Data Recovery Technique in Cloud Computing," Internet of Things and Cloud Computing, vol. 5, no. 1, pp. 13–18, 2017.

[7] J. Yu and L. Yang, "The Cloud Technology Double Live Data Center Information System Research and Design based on Disaster Recovery Platform," Procedia Engineering. vol. 174, pp. 1356-1370, 2016.

[8] S. Nadgowda, P. Jayachandran, and A. Verma, "I2MAP: Cloud Disaster Recovery Based on Image-Instance Mapping," Middleware 2013. pp. 204-225, 2013.

[9] L. Wang, R. E. Harper, R. Mahindru and H. V. Ramasamy, "Disaster Recovery for Cloud-Hosted Enterprise Applications," the 9th IEEE International Conference on Cloud Computing (CLOUD), pp. 432-439, 2016.

[10] X. Wu, F. Zou, G. Fu, "Cloud Application Disaster Recovery Method, System and Device," patent No. EP2879060, 2017.