International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. #, No.# (Mon-20..)

Providing Fault Tolerance Mechanism in Clinical Support System Through Fog Computing as Middleware

Mohd Hariz Naim[1], Jasni Mohamad Zain [2], and Kamarularifin Abd Jalil[3]

[1] Centre for Advanced Computing Technology C-ACT, Universiti Teknikal Malaysia Melaka, Melaka, Malaysia
[2] Institute for Big Data Analytics and Artificial Intelligence, Universiti Teknologi MARA, Shah Alam, Malaysia
[3]Department of Computer Technology and Network, Universiti Teknologi MARA, Shah Alam, Malaysia

E-mail address: mohdhariz@utem.edu.my, jasni@tmsk.uitm.edu.my, kamarul@tmsk.uitm.edu.my

Received ## Mon. 20##, Revised ## Mon. 20##, Accepted ## Mon. 20##, Published ## Mon. 20##

Abstract: Fault tolerance is one of the paradigms for providing high availability in computerized system where application service is replicated to multiple nodes. The paradigm is widely used in cloud computing environment where users may benefit automatic backup when the application such as clinical support system is deployed in cloud. However, application residing in premise such as small clinics are still prone to outage and would require certain time with human involvement when performing recovery. Thus, the objective of this research is to provide a backup mechanism for health system residing in-premise of a clinic. We proposed the use of fog computing model that acts as middleware for detecting and failover solution when an outage has temporarily occurred. The middleware will perform failure detection through heartbeat and replicate the services at the same time. When an outage is detected, the middleware will take action to take over as secondary service provider to ensure applications may be used seamlessly.
.

Keywords: Fault Tolerance, Fog Computing, High Availability

1. Introduction

Computer system need to ensure three security elements consisting of Confidentiality, Integrity and Availability (CIA) [1] when providing services to end users. The term high availability is part of the Availability element where the system need to provide 99.9999% of uptime [2], [3] which is also known as the five nine. The availability of system to end users may differ due to many factors and as for this reason, it is a challenge for software and system providers to meet the Service Level Agreement (SLA) in term of availability. In critical sectors such as health care, the availability of computerized system is crucial to provide seamless information so that medical personnel could respond to patient accurately when giving treatment.

Many software providers preferably deploy the applications on the cloud as a popular solution as cloud providers could take care of the infrastructure as well as the availability. The cloud architecture however would require Internet connectivity for end users to access the applications. This however has created certain issues such as network attacks [4], performance delay [5] and expensive subscription service fee when using the cloud provision [6], [7]. Aa a result, some solution providers are looking an alternative to deploy their application closer to users such as in-premise server deployment. The term closer to users are associated with fog computing [8] where the machine or devices are located at the edge of network [9] isolated from the cloud computing architecture. Together with implementation of cloud computing, the fog computing will act as a supplementary to the computer system in the event of cloud failure, where the fog computing may continuously provide alternative services to the end users.

In ensuring seamless application services running, the cloud computing technology have implemented several measures in providing high availability. In general, there are three mechanisms as mentioned by [1]: Fault Tolerance, Protective Redundancy and Overload Protection in which will be explained in detail in the next section. Even though all the mechanisms are implemented by cloud providers, the outage would still occurs unexpectedly causing interruptions to hundreds of back-end service operations. In addition, for applications running internally within an organization would not be able to take the benefit of cloud and would need to setup their own backup and secondary server.

2. RELATED WORKS

Cloud computing is widely used nowadays due to the popularity and advancement of Internet connectivity and the wide use of computerized devices. Mobile phones, small peripherals including Internet of Things rely heavily on the Internet to connect to the cloud. For this reason, software providers prefer to deploy software services especially back-end services such as web services on the cloud servers. The term cloud computing is referred as pool of computerized system resources [10] that are used by end users regardless of different operating system and hardware devices. With cross platform and less dependency on hardware-centric, the cloud computing able to integrate and exchange communication between multiple devices.

The cloud services are also considered as distributed computing [11] which consist of categories namely as Infrastructure as a Service (IaaS), Platform as a Service (Paas) and Software as a Service (Saas) [1], [11]–[13]. These services have different characteristics in providing daily business activities and operation. As mention by [1], the SaaS cloud provides a complete and ready to use application for end users where users may buy the services base on subscription such as Dropbox, Netflix and Office 365. The PaaS on the other hand, focus on option to deploy end user's solution through semi-complete and less hectic server setup. While the IaaS provides virtual or hardware resources including network connectivity collaboration between different geographical location. With different scope of services, users of all level may flexibly customize their cloud subscriptions based on their requirements and budget.

Despite the benefit of the cloud computing has to offer, the cloud is still prone to some issues that users may need to consider. According to research done by [5] which mentioned that the latency issues are due to the geographically distributed cloud servers whilst the users are physically located far from the servers. This means that the farther the users are located, the more time would be needed for the service to acknowledge and provide the service. Consecutively, this explains the need for the users to specify which location they would prefer to access the service especially during downloading files from server. Although the cloud architecture have implemented certain high availability mechanism, there is unexpected downtime that experience by certain cloud providers [14]. It was reported that even the renowned cloud provider such as Amazon EC2 and Google Cloud Platform also suffers from sudden outage that would impact high cost to the end users [14]. Additionally, the situations are worsen if the applications hosted on the cloud require real-time respond which need low latency and high availability of service [15] such as providing service related to financial and health care. Any latency or outage of service would have huge impact and loss to the involved users. In ensuring high availability and seamless operation of service at the application layer, some software providers make use of the middleware [16] for integrating the service among different cloud providers. However, some cloud providers do not provide platform and features for implementing the middleware, making the high availability mechanism limited to the cloud layer.

As the cloud computing progress to suit the users demand, the fog computing also emerged as a supplementary to the cloud infrastructure. The term fog is viewed as mass of devices ranging from mobile phones, small peripherals and sensors [17] connected through certain protocols. As reviewed by [18], [19], the fog computing is another paradigm of computing where the logical process such as data collection, calculation, analysis and decision making are executed by devices residing at the edge of network instead of the cloud. All the devices are capable to perform tasks starting from data collection, storing into database and other tasks with minimal dependency from the central or cloud servers. Figure 1 visualizes the role of fog computing which shows how the fog devices residing in edge of network correlating between application and cloud servers. This proves to be helpful and beneficial in term of latency, better real time response and independent from the Internet [18].

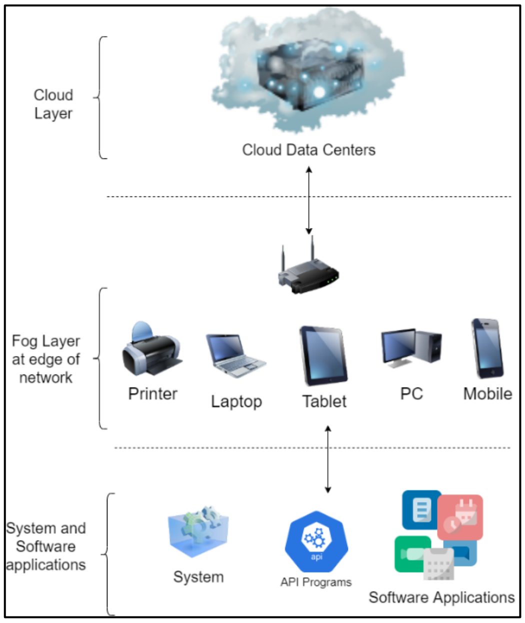

Figure 1. Role of Fog Layer

In fact, the usage of fog devices such as sensors devices for healthcare usage would bring more flexibilities and advantages such as portability and real time processing [19], [20] as they are closer and can be attached directly to patients. This is crucial for patients that require constant monitoring and elderly citizens that lives away from relatives and medical personnel. In the event of emergency occurrences, the IoT fog devices may trigger notifications and alerting emergency responses to the relatives. Additionally, fog devices may also help to observe and perform data collections among patients with better latency. In term of connectivity, the fog devices are able to perform without Internet connection and capable to work individually [8], [19] without depending on cloud centralized servers. This proves to be beneficial for remote clinics or health centres that are isolated and require offline processing rather than depending on centralized cloud servers that need stable Internet connectivity. The fog devices however, still suffers from some issues and challenges especially related to the limited power supplies and resources capabilities [17], [19] which is not at the same par with cloud resources. Moreover, deploying healthcare application at isolated clinics and health centres on single host node as a server is prone to downtime and service interruptions as fault tolerance mechanisms must be manually imposed at the surrounding environment. This eventually would make deployment expensive and hard to maintain in the future. In addition, remote clinics are usually small in size and do not have dedicated staff to manage the infrastructure.

In order to ensure the high availability of services, many IT providers would implement certain mechanisms. According to [1], the availability mechanisms namely fault tolerance, protective redundancy and overload protection are widely implemented in cloud environment. These mechanisms are very suitable for clouds as the resources and hardware capabilities are significantly high end [21] where the storage is massive while the processing power are scalable. Cloud providers such as Google, Microsoft Azure and Amazon AWS would implement all the availability mechanisms within their data centres that are distributed around the world which mean that they have abundant of resources to spare promising close-to-zero downtime. Despite the commitment to fulfilling the Service Level Agreement (SLA), the outage of service could still happen as reported by [22] where unexpected service downtime by certain cloud providers has exceeded beyond the accepted downtime. The fault tolerance is defined as the capability of a system to continuously operate even when a fault has occurred where it includes avoiding a failure or failure detection before able to recover [1]. Generally, the fault tolerance mechanisms are categorized into post active [11] and proactive [13], [23] technique. The proactive would mean the mechanism will take precaution measurement before the failure occurs while the post active or reactive method would need to take recovery action after the failure has occurred.

In order to execute the proactive measurement, the information regarding the resources are needed such as resource consumption and utilization from previous history where the record is modelled [11], [24] to predict when will the next downtime is expected to occur. Besides prediction, the proactive also involve taking preventative actions such as rebooting and restarting operating system and application services [25] to prevent system aging which can lead to application failure when the resources are not freed up. These proactive measurements however still suffer from downtime as rebooting the operating system and services would take some time [26] to get back up and running especially for big data centres where rebooting the entire systems at the same time is not an effective method. Additionally, modelling and prediction also prone to false positive detection and inaccurate estimation [11] which require deep and more knowledge about the node task activities and resource utilization.

On the other hand, the post active measurement would need to continuously monitor the remote node for any failure by implementing certain such as heartbeat and watch dog timer [24]. Through this technique, a signal is continuously sent to the monitored node which then listened for response from the monitored node. Once the monitored node is not responding to the signal, a failure is indicated and may trigger to perform recovery actions. Since it requires seamless monitoring, this is very costly in term of resources as additional computational overhead is incurred [11]. The recovery actions would need to redirect or provide secondary service provider which is also known as switch over or protective redundancy where another node will act as a backup [21], [27] to take over from the failed node. For application-based fault tolerance, in case of failure occurrence the software provider could employ Roll-Back and Roll-Forward mechanism [1] as part of post active measurement. These would involve in saving the state of application to a save point [23] especially related to data that is stored in database. Whenever the data is corrupted or missing, the application can change back to the saved point to recover the application from failure.

Under the protective redundancy mechanism, there are several models that can be implemented namely 2N, N+M, N-Way and N-Way Active [1]. Here the models would refer to the number of replications that will be created to serve as backup node where the backup would be scattered around either within same or different geographical place. The N indicates the redundant element or replicated node that acts as standby while the M is the active node providing the primary services. According to [1], the 2N model will be having two nodes or elements where one will provide main services while the other element will act as standby node, this way when the main node has failed, the standby node will take over to provide the required services. On the other hand, the N+M model will create standby node depending on the number of main nodes and the standby node must be more than the main node as the main idea is to provide multiple geographically located standby nodes [11]. In the event of natural disaster such as earthquake and flood, the services can be redirected to another location. Contrarily, the N-Way model would also require the redundant node to provide some services instead of just standing by awaiting to take over the main node. This eventually, will optimize the resources and able to balance the task load between main node and replicated nodes. Similarly, the N-Way Active model also make all the available node as the main node which protect each other and provide all the required services. In other words, there is no standby node as all the nodes are actively providing services to the end client.

Based on the mechanisms that have been discussed, it was found that the most suitable technique that is implemented in this research is the post active with the combination of fault tolerance and failure detection by using heartbeat signal listener. This is because for clinical support system, the end users are residing at the edge of network which is prone to connectivity failure and unable to access to the required services due to application failure when performing daily tasks. For this reason, the aim of the research is to provide an alternative high availability services by utilising fog devices residing on edge of network. Further discussion on the implementation will be explained in the next section.

3. METHODOLOGY

In this section, the detailed experimentation and proposed solution will be explained. In this research, there are three proposed modules namely, the failure detection, service replication and task offloading to secondary backup node as depicted in Figure 2. The first action before applying failover solution is to correctly identify failure occurrences with minimal false positive. At the same time, another dedicated machine or node is configured to become the redundant elements which is called the cloned fog device. The last module is the switchover or offloading the services to the cloned fog devices which resides closer to end users located in a clinic. Here, the proposed components are installed to a dedicated fog device which is a Raspberry Pi 4 residing at the same network within a health centre so that whenever a network failure is detected, the fog device will swap its address that mimicking the central main server. The fog device has resource specification of 2 gigabyte of memory and extended storage of 128 gigabyte hard disk storage which is enough for storing cloned data for small application such as clinical support system.

A. Failure Detection Module

In this module, the failure detection is implemented based on the heartbeat technique [23] where the module will keep on listening to the main cloud server by continuously sending periodic signals and await for respond. The module is written in Python script which points to the main cloud server address. Here the heartbeat listener will keep on running and trigger to the signal for every 5 seconds executed by Cron script that is available by Raspbian operating system of the Raspberry Pi.

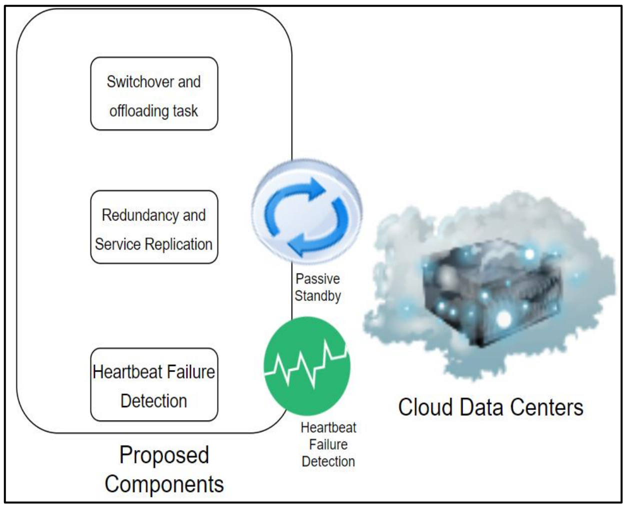
Switchover and offloading task
Redundancy and Service Replication
Heartbeat Failure Detection
Proposed Components
Passive Standby
Heartbeat Failure Detection
Cloud Data Centers

Figure 2. The proposed components

B. Redundancy and Replication of Services

The second module is the replication of services that mirrored to the main cloud server. However, due to constraint of storage and memory resources by the Raspberry Pi device, only important services are replicated in the device. Here, the services that are replicated are the Apache Tomcat service for running back-end application, the MySQL service for providing database and the Clinical Support System which is a Java Swing application that run on the Apache Tomcat. The installed service act as a cold standby node which means that the end users do not access the fog device as the primary source, instead it would only be accessed when an outage has occurred at the main central server. In order to ensure that the data is up to date with the centralized server, the MySQL database is synchronized by enabling data mirroring feature.

C. Switchover and Offloading Task

Since the Raspberry Pi is a cold standby node, it also known as slave node that await to take over from the central main server that is the master node. Here, we proposed that both the master and the slave to have its own dedicated static IP address rather than dynamic address so that it would be easier to switch the address in the event of failure occurrences. When the heartbeat has triggered the failure connectivity, the IP address of the slave would switch to be the same as the master's IP address. This way, end users would not notice service interruption and able to access to the clinical support system seamlessly as the desktop application has resolved the inaccessible address.

4. Simulation and Result

From the proposed methodology, the environment take place in a private health centre where it simulates the edge of network that has separated network architecture compared to central cloud server. The fog device is placed within the health centre which belong to the same network segment with other devices that are attached to a single switch ad depicts in Figure 3.

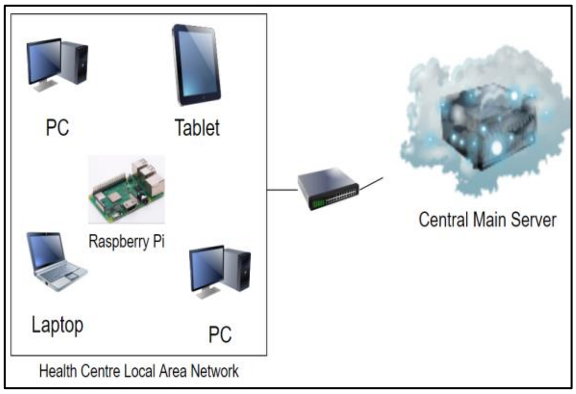

Figure 3. All Devices connected within same local area network.

Here, other devices such as desktops and tablets would connect to the same switch and when network outage has occurred, other devices are isolated but still connected as single local area network (LAN) which mean all the devices are able to communicate with each other. The simulation of failure is done by disconnecting the switch connection towards the central main server which is similar to the situation where the network has failed or in the event that the Internet connectivity is not available.

Based on the process flow shown in Figure 4, the heartbeat signals are sent to the central main server for every 5 seconds and wait for respond. At the same time if the signal is acknowledged, the fog device will perform data synchronization to update its local database. The data in central server is massive which has more than 10 gigabytes of data, thus in order to avoid unnecessary synchronization and avoiding network congestion, the synchronization only executed when the binary log is different between the master and slave. A difference in binary log would indicate that the data has changed, and only certain data is passed to the slave rather than passing the whole database.

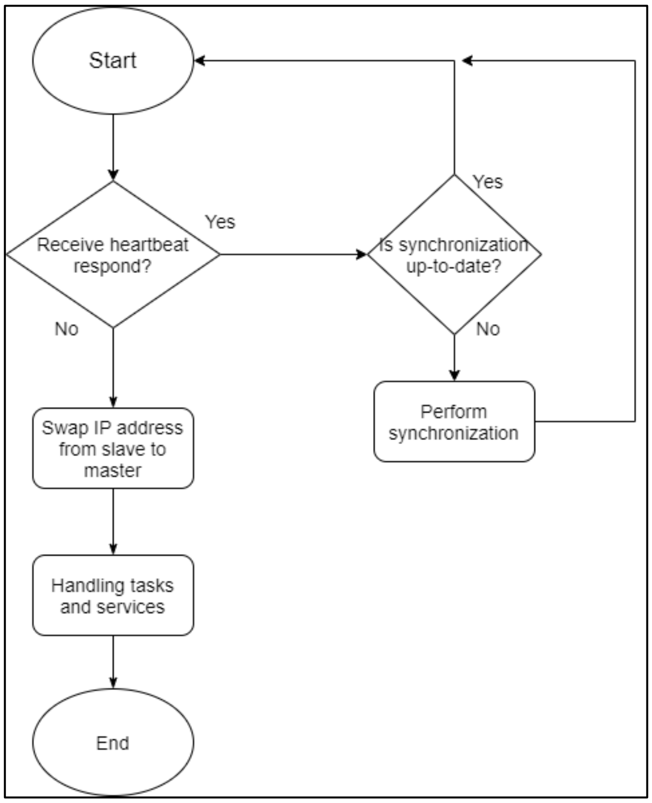
Start
Receive heartbeat respond?
Yes
No
Is synchronization up-to-date?
Yes
No
Perform synchronization
Swap IP address from slave to master
Handling tasks and services
End

Figure 4. Proposed Process Flow

On the other hand, if the heartbeat is not getting any respond from the central server, the fog device that act as slave will change its IP address to become similar with the central server. In this simulation, initially the central server has address of 10.102.100.100 while the fog device has address of 10.102.100.102 as depicts in Figure 5. When an outage occurred, the fog device address will change to 10.102.100.100 as visualized in Figure 6. By swapping the address, the clinical support system desktop application will automatically refer the tasks and services to the fog devices without having to manually redirect or change service provider for every device in the same network segment.

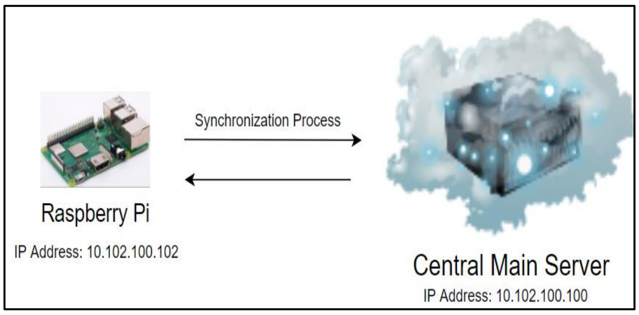

Figure 5. IP address configuration during uptime

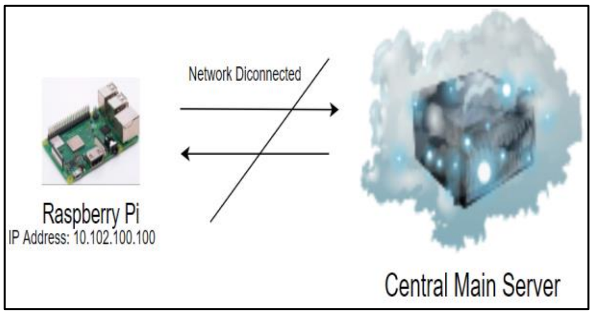

Figure 6. IP address configuration during network failure

During an outage, the fog device will keep on sending heartbeat signal to the central main server and when the network has re-established, the fog device will quickly swap back to its original IP address to avoid address conflict. All the devices will now point to the central main server and the fog device will become the slave node performing cold standby which continue to monitor central main server for any outage by sending heartbeat signal.

A. Result

In this subsection, the result of the simulation will be presented where failure simulation is executed and presented and discussed in detail. The result of the simulation is based on time to recover from failure upon disconnection of switch from the Internet source which is the main connection to the main central server. Here we record the downtime and the time taken to recover up to 3 trials to observe the recovery consistency. The recovery time taken is recorded by the fog device which is based on the connection to the neighbouring devices when failure has occurred. Another measurement metric that is taken into consideration is the time taken by the application users to gain access upon a network failure where the time is recorded by the Clinical Support System when it has detected inaccessible back-end service to the main cloud server.

TABLE 1. SIMULATION RESULT BASED ON FAILURE CONDITION

No of Trial	Recovery Time Taken (seconds)	Time Taken for Application to Gain Access (seconds)
1st	5.1	4.3
2nd	5.5	6.1
3rd	5.3	4.8

It is observed that the time taken for the application to recover is slightly different from the time taken to recover by the fog device. This might be due to the back-end application programming interface (API) called by the Clinical Support System that is restricted and random by the Java library. In addition, optimizing the application for faster service recovery is beyond our research scope.

5. CONCLUSION AND FUTURE WORK

In this research, we have proposed the utilization of Raspberry Pi as fog devices to provide fault tolerance mechanism in a health centre environment where the device acts as a cold standby node as part of redundancy element. From the simulation, the proposed method has proven that the fog device is capable of providing backup service whenever a network segment is experiencing connectivity downtime. Additionally, the proposed method would be beneficial to be implemented in the edge of network where Internet connectivity is an issue.

Despite the advantages that the proposed method has to offer, there are still some issues that need to be resolved. It is known that fog devices are limited in term of resources which is not at the same par with cloud server resources [28]–[30]. As for this reason, the proposed method would need only certain services to be installed and running on the fog device. The limitation of storage also hinders the fog device from storing gigantic database and would need data optimization or meticulous data selection before storing into the fog device. In term of fault tolerance, the proposed method is limited to provide recovery for desktop application and does not cater the domain name as well as dynamic address for more flexible IP address assignation. For future work, we aim to provide more supple and wider application that can be protected from failure.

ACKNOWLEDGMENT

The author would like to express gratitude Center of Advanced Computing Technology (C-ACT) Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka and Center of Computer Technology and Network (CTN) Faculty of Computer Science and Mathematics Universiti Teknologi Mara, Malaysia for offering facilities in term of research accommodations.

REFERENCES

[1] M. Nabi, M. Toeroe, F. Khendek, M. Nabi, M. Toeroe, and F. Khendek, "Availability in the Cloud : State of the Art," *J. Netw. Comput. Appl.*, no. 60, pp. 54–67, 2016, doi: 10.1016/j.jnca.2015.11.014.

[2] R. Moreno-Vozmediano, R. S. Montero, E. Huedo, and I. M. Llorente, "Orchestrating the deployment of high availability services on multi-zone and multi-cloud scenarios," *J. Grid Comput.*, vol. 16, no. 1, pp. 39–53, 2017, doi: 10.1007/s10723-017-9417-z.

[3] M. R. Mesbahi, A. M. Rahmani, and M. Hosseinzadeh, "Highly reliable architecture using the 80/20 rule in cloud computing datacenters," *Futur. Gener. Comput. Syst.*, vol. 77, pp. 77–86, 2017, doi: 10.1016/j.future.2017.06.011.

[4] N. Baharudin, F. H. M. Ali, M. Y. Darus, and N. Awang, "Wireless intruder detection system (WIDS) in detecting de-

authentication and disassociation attacks in IEEE 802.11," *2015 5th Int. Conf. IT Converg. Secur. ICITCS 2015 - Proc.*, pp. 1–5, 2015, doi: 10.1109/ICITCS.2015.7293037.

[5] Y. Aldwyan and R. O. Sinnott, "Latency-aware failover strategies for containerized web applications in distributed clouds Cloud Failover Techniques :," *Futur. Gener. Comput. Syst.*, vol. 101, pp. 1081–1095, 2019, doi: 10.1016/j.future.2019.07.032.

[6] E. F. Coutinho, F. R. de Carvalho Sousa, P. A. L. Rego, D. G. Gomes, and J. N. de Souza, "Elasticity in cloud computing: a survey," *Ann. des Telecommun. Telecommun.*, vol. 70, no. 7–8, pp. 289–309, 2015, doi: 10.1007/s12243-014-0450-7.

[7] P. Alves Lima, A. Sá Barreto Neto, and P. Romero Martins MacIel, "Data Centers Service Restoration Based on Distributed Agents Decision," *Proc. - 2018 IEEE Int. Conf. Syst. Man, Cybern. SMC 2018*, pp. 1611–1616, 2019, doi: 10.1109/SMC.2018.00279.

[8] E. Ahmed and M. H. Rehmani, "Mobile Edge Computing: Opportunities, solutions, and challenges," *Futur. Gener. Comput. Syst.*, vol. 70, pp. 59–63, 2017, doi: 10.1016/j.future.2016.09.015.

[9] P. Hu, S. Dhelim, H. Ning, and T. Qiu, "Survey on fog computing: architecture, key technologies, applications and open issues," *J. Netw. Comput. Appl.*, vol. 98, no. September, pp. 27–42, 2017, doi: 10.1016/j.jnca.2017.09.002.

[10] K. Syed and K. Vijaya, "Cloud Computing: Review on Recent Research Progress and Issues," *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 8, no. 3, pp. 959–962, 2019, doi: 10.30534/ijatcse/2019/96832019.

[11] A. S. Ebenezer, E. B. Rajsingh, and B. Kaliaperumal, "A novel proactive Health Aware Fault Tolerant (HAFT) scheduler for computational grid based on resource failure data analytics A novel proactive Health Aware Fault Tolerant (HAFT) scheduler for computational grid based on resource failure data analyti," *Int. J. Comput. Appl.*, vol. 7074, pp. 1–11, 2018, doi: 10.1080/1206212X.2018.1440339.

[12] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile cloud computing: A survey," *Futur. Gener. Comput. Syst.*, vol. 29, no. 1, pp. 84–106, 2013, doi: 10.1016/j.future.2012.05.023.

[13] B. Snyder, J. Ringenberg, R. Green, V. Devabhaktuni, and M. Alam, "Evaluation and design of highly reliable and highly utilized cloud computing systems," *J. Cloud Comput.*, vol. 4, no. 1, 2015, doi: 10.1186/s13677-015-0036-6.

[14] L. Acquaviva, I. Informatica, S. Monti, and I. Informatica, "NoMISHAP : A Novel Middleware Support for High Availability in Multicloud PaaS," *IEEE Cloud Comput.*, vol. 4, no. 4, pp. 60–72, 2017.

[15] K. An, S. Shekhar, F. Caglar, A. Gokhale, and S. Sastry, "A cloud middleware for assuring performance and high availability of soft real-time applications q," *J. Syst. Archit.*, vol. 60, no. 9, pp. 757–769, 2014, doi: 10.1016/j.sysarc.2014.01.009.

[16] W. Li, A. Kanso, and A. Gherbi, "Leveraging Linux Containers to Achieve High Availability for Cloud Services," pp. 76–83, 2015, doi: 10.1109/IC2E.2015.17.

[17] S. Svorobej *et al.*, "Simulating fog and edge computing scenarios: An overview and research challenges," *Futur. Internet*, vol. 11, no. 3, pp. 1–15, 2019, doi: 10.3390/fi11030055.

[18] I. Sittón-Candanedo, R. S. Alonso, J. M. Corchado, S. Rodríguez-González, and R. Casado-Vara, "A review of edge computing reference architectures and a new global edge proposal," *Futur. Gener. Comput. Syst.*, vol. 99, no. 2019, pp. 278–294, 2019, doi: 10.1016/j.future.2019.04.016.

[19] K. Bilal, O. Khalid, A. Erbad, and S. U. Khan, "Potentials, trends, and prospects in edge technologies: Fog, cloudlet, mobile edge, and micro data centers," *Comput. Networks*, vol. 130, no. 2018, pp. 94–120, 2018, doi: 10.1016/j.comnet.2017.10.002.

[20] F. A. Kraemer, A. E. Braten, N. Tamkittikhun, and D. Palma, "Fog Computing in Healthcare-A Review and Discussion," *IEEE Access*, vol. 5, no. 2169, pp. 9206–9222, 2017, doi: 10.1109/ACCESS.2017.2704100.

[21] G. Estácio *et al.*, "Resource allocation based on redundancy models for high availability cloud," *Computing*, vol. 102, no. 1, pp. 43–63, 2020, doi: 10.1007/s00607-019-00728-1.

[22] M. R. Mesbahi, A. M. Rahmani, and M. Hosseinzadeh, "Reliability and high availability in cloud computing environments: a reference roadmap," *Human-centric Comput. Inf. Sci.*, vol. 8, no. 1, 2018, doi: 10.1186/s13673-018-0143-8.

[23] S. Prakash, V. Vyas, and A. Bhola, "Proactive Fault Tolerance using Heartbeat Strategy for Fault Detection," no. 1, pp. 4927–4932, 2019, doi: 10.35940/ijeat.A2079.019119.

[24] S. J. Ellis *et al.*, "Runtime Fault Detection in Programmed Molecular Systems," vol. 28, no. 2, 2019.

[25] T. Azman, N. Pa, R. Nor, and Y. Y. Jusoh, "Constructing a model of risk mitigation for anti software ageing during software maintenance," *Int. J. Adv. Sci. Technol.*, vol. 28, no. 2, pp. 341–347, 2019.

[26] G. Levitin, L. Xing, and Y. Xiang, "Cost minimization of real-time mission for software systems with rejuvenation," *Reliab. Eng. Syst. Saf.*, vol. 193, no. March 2019, p. 106593, 2020, doi: 10.1016/j.ress.2019.106593.

[27] P. T. Endo, M. Rodrigues, G. E. Gonçalves, J. Kelner, D. H. Sadok, and C. Curescu, "High availability in clouds: systematic review and research challenges," *J. Cloud Comput.*, vol. 5, no. 1, 2016, doi: 10.1186/s13677-016-0066-8.

[28] A. Yousefpour *et al.*, "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *J. Syst. Archit.*, vol. 98, no. February, pp. 289–330, 2019, doi: 10.1016/j.sysarc.2019.02.009.

[29] M. Etemadi, M. Ghobaei-Arani, and A. Shahidinejad, "Resource provisioning for IoT services in the fog computing environment: An autonomic approach," *Comput. Commun.*, vol. 161, no. July, pp. 109–131, 2020, doi: 10.1016/j.comcom.2020.07.028.

[30] D. R. Sangolli, N. M. Ravindrarao, P. C. Patil, T. Palissery, and K. Liu, "Enabling high availability edge computing platform," *Proc. - 2019 7th IEEE Int. Conf. Mob. Cloud Comput. Serv. Eng. MobileCloud 2019*, pp. 85–92, 2019, doi: 10.1109/MobileCloud.2019.00019.

Mohd Hariz bin Naim is a lecturer at Universiti Teknikal Malaysia Melaka(UTeM) in the department of software engineering faculty of information and communication technology (FICT). He is currently pursuing his PhD in field of Software Development and IT Operation (DevOps) related to high availability and fault tolerant. He is also interested in the field of mobile application development.

Jasni Mohamad Zain is a professor at the Institute of Big Data Analytics and Artificial Intelligence, Universiti Teknologi MARA (UiTM) Shah Alam Malaysia. She obtained her PhD from Brunel University West London, UK. Her research interests are related to digital watermarking, image processing, data security and data mining.

Kamarularifin Abd Jalil is an associate professor at the department of computer technology and networking, Faculty of Computer and Mathematical Science Universiti Teknologi MARA, Shah alam, Malaysia. He obtained his PhD at University of Strathclyde, UK. His research interests are mobile communications, mobile networks & protocols, wireless networks, digital forensics and information security