DEVELOPMENT OF PATIENT MONITORING SYSTEM ARCHITECTURE BASED ON KUBERNETES CONTAINER PLATFORM

Liana Andreasyan, *Lecturer of National Polytechnic University of Armenia (NPUA), Yerevan, Armenia*
Styopa Harutyunyan, *Master's student of NPUA, Yerevan, Armenia*
Hovhannes Grigoryan, *Master's student of NPUA, Yerevan, Armenia*

DOI: https://doi.org/10.31435/rsglobal_conf/30012021/7381

Abstract. *One of the main tasks of high-tech medicine is to provide personalized medical care to patients in real-time and to develop procedures for reducing medical errors due to the increase in the number of modern diseases, associated drugs and laboratory tests, new technologies, and the processing of a large amount of information related to them. Thus, it is necessary to automate data processing, ensuring the integrity and security of medical data transmission in cloud-based solutions with different types of network architecture. This paper proposes a way to expand the availability and usability of medical data to improve patient data storing, processing, and monitoring based on Fog to Cloud (F2C) paradigm using Kubernetes (K8s) container orchestration platform that automatically manages and deploys a group of microservices.*
Keywords: *F2C, Fog node, IoMT, Fog Cluster, Minikube, Kubernetes platform, microservices.*

Introduction. The cloud solutions used to process big data flows have limitations due to the slow interaction between geographically centralized architecture, real-time users, and cloud platforms. In addition, online connection failures can disrupt the operation of software-enabled devices connected to smart devices and medical gadgets, leading to life-threatening incidents. A new paradigm has been launched called Fog computing that creates a cloud infrastructure called Fog Node. The infrastructure provides data collection and processing via wireless communication technologies [1, 2].

The use of Fog technology is an expansion of cloud computing, consisting of edge nodes that are directly connected to physical devices. It can be a three-tier or multi-level F2C collaboration [3, 4]. F2C acts as an intermediate level between the cloud and Internet of Medical Things (IoMT). The use of IoMT provides several advantages:

- Increased network efficiency;
- Fewer electricity costs;
- High degree of distribution of network nodes;
- Possibility of initial data processing resulting in fast reply-response rate;
- Compatibility with cloud solutions.

The basic architecture of fog computing consists of three main levels:

Device layer: The closest layer to the end-users/devices. It consists of distributed smart devices and sensors. These devices are sending data to the upper Fog layer.

Fog layer: The middle layer is located at the edge of the network. It contains Fog nodes or clusters that include gateways, access points, routers, base stations. Fog nodes are responsible for performing tasks such as data storing, processing, and managing.

Cloud layer: The cloud layer is responsible for reliable storage, as well as for exploratory and confirmatory analysis of the data.

In modern cloud solutions, there is a transition from monolithic systems to platforms that provide microservices. One such platform is K8s, an open-source, portable, non-monolithic platform for automating the deploying, scaling, and management of microservices. K8s is a master/slave model, the master node manages containers through multiple workers (slave or minion) nodes. These nodes can be deployed on physical servers (virtual machines) as well as in public and private clouds. The wizard is responsible for managing the API server, the service deployment schedule, and the overall cluster. The API server is implemented through the RESTful interface. This is the access point for managing the entire K8s clusters. Users can send commands to the API server with the built-in K8s through a command-line interface (CLI) known as kubectl. Kubelet runs on every node in the cluster to start or terminate the necessary containers. In K8s, microservices are often combined to form a group of containers that represent the smallest work unit called a Pod. Considering the service load and using the available resources, the master places the Pod on a specific node. Then, if necessary, the designated node loads the container image from the appropriate register and directs the Pod to perform the necessary actions. Kube-scheduler (KS) is the default scheduling component on the K8s platform that determines the location of the Pod (fig. 1).

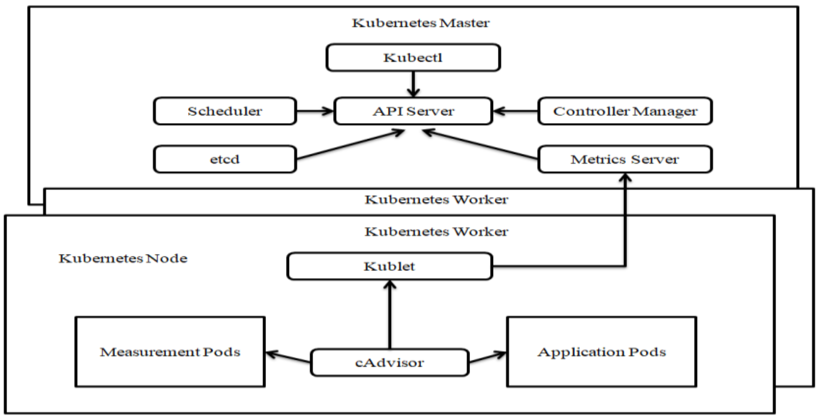

Fig. 1. Generalized architecture of the Kubernetes platform

Both single node (single-master/single-worker) and multi-node (single-master/multi-worker) clusters can be organized on the K8s platform. F2C technology allows for data transmission and processing using both clusters on the K8s platform [6].

The purpose of this article is the development of the architecture of a patient-monitoring system based on the K8s platform with F2C technology.

The main objectives of this paper are:

- To design the architecture and application model of the Fog computing-based remote monitoring network PLAN (Patient Local Area Network). PLAN has a configurable interface for processing various bio-signals and can perform real-time gathering and transmission of health data via Message Queuing Telemetry Transport (MQTT) protocol to the HLAN (Hospital Local Area Network) network;
- To develop the HLAN architecture using K8s container platform and the Fog cluster model which will provide functions needed to monitor outpatients;
- To introduce a collaboration with Fog clusters for the HLAN on single- and multi-node container platforms.

F2C based patient monitoring network. For the suggested patient monitoring system, the transmission and processing of patient data are organized through the PLAN patient network and the HLAN hospital network. Figure 2 shows a simplified illustration of the connection between a single fog node and HLAN.

There are invasive and non-invasive wearable and non-wearable sensors for measuring the patient's vital signs (VS) - heart rate (HR), respiratory rate (RR), body temperature (BT), electrocardiogram (ECG), skin temperature (ST), blood pressure (BP), blood sugar (BS), oxygen saturation (SpO_2), etc. In general, the Arduino Nano board based on the Atmega328 microcontroller is used for measuring and monitoring various parameters such as SpO_2, BT, HR, BP, etc. collected from non-wearable sensors. To date, quite a few wearable sensors allow for accurate monitoring of sleep, BT, HR, BP, and SpO_2 using BLE and Wi-Fi connectivity. This project is designed to be able to use both wearable and non-wearable sensors for collecting continuous VS.

The hardware platform implementing the PLAN network consists of wearable and non-wearable sensors, Arduino Nano board, Raspberry Pi 4 microcomputer, and Mikrotik Hap Lite router, enabling communication with clinicians using a smartphone (or tablet) and Internet. In the Fog node, Raspberry Pi is used to collect, check, and store data in a local database (in our case, the database is MariaDB on Raspbian OS), after which the data is sent by MQTT protocol to the HLAN server. The gathered data is transmitted to a getaway via wireless or wired communication protocols such as Ethernet, Bluetooth, and Wi-Fi. Mosquitto is used as an MQTT broker in Raspberry Pi as shown in fig. 3 [5].

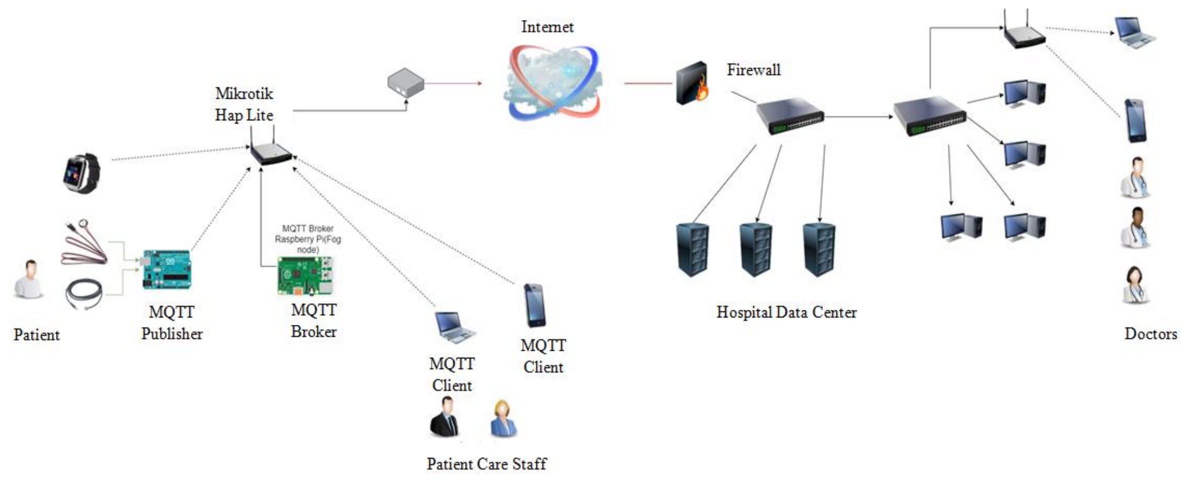

Fig. 2. General view of a remote monitoring network in the case of a single Fog node

The data from the sensors are converted via MQTT Publisher to the corresponding format. Since remote monitoring devices generate heterogeneous data, it is necessary to separate this data by MQTT topics. For example, the data from the temperature sensor can be sent to /health/temperature topic, and the data from the heart rate sensor can be sent to /health/heartrate topic. Each remote monitoring device has a unique identifier and the ability to recognize and transfer data over to the network.

The general view of the PLAN network is presented in fig. 3. The information collected from the various sensors is sent to the MQTT publisher which processes the data by topics. The Raspberry Pi device is used as a Mosquitto broker. It receives the data from the MQTT publisher and sends it to the subscribers who have permission to view that topic.

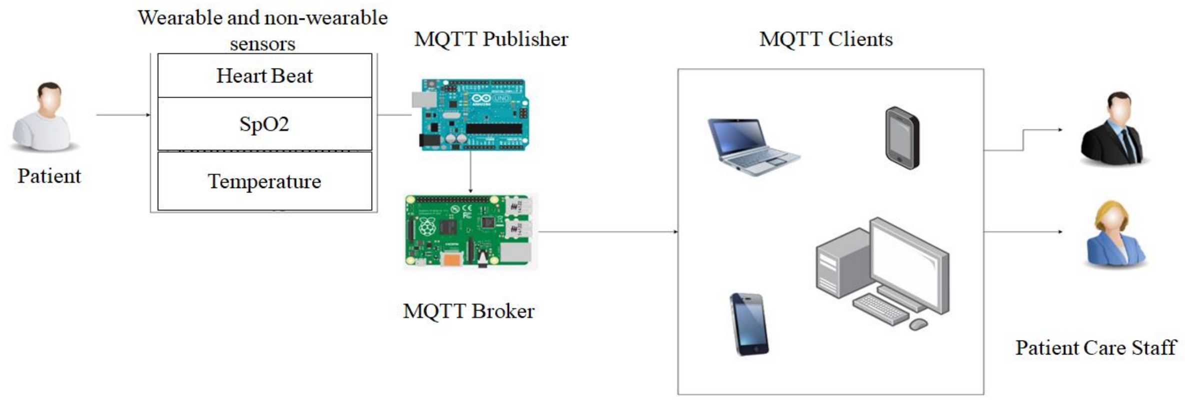

Fig. 3. General view of the PLAN architecture

The patient's health status parameters are monitored in real-time and transmitted through the Fog node to the hospital FTP server for storing and processing (fig. 4). Security in Fog computing plays an important role because data fraud can have serious health consequences for patients. Health data protection in this research is provided by SFTP protocol, Mosquitto Auth Plugin with MySQL [8, 9] which is also used for authentication and authorization. Fig. 4 shows the Fog Node workflow diagram for N=3600 seconds, where N is a configurable parameter and can be changed with the doctor's recommendations.

Application A1 works in the background to perform the appropriate tasks described in the MQTT message. Application A2 creates a copy of the database through the Mysqldump program and compresses it via gzip (we get a file that is about 40 times smaller). The created .gz file will be encrypted and sent to the HLAN via the SFTP protocol. The A1 and A2 applications are written in Python.

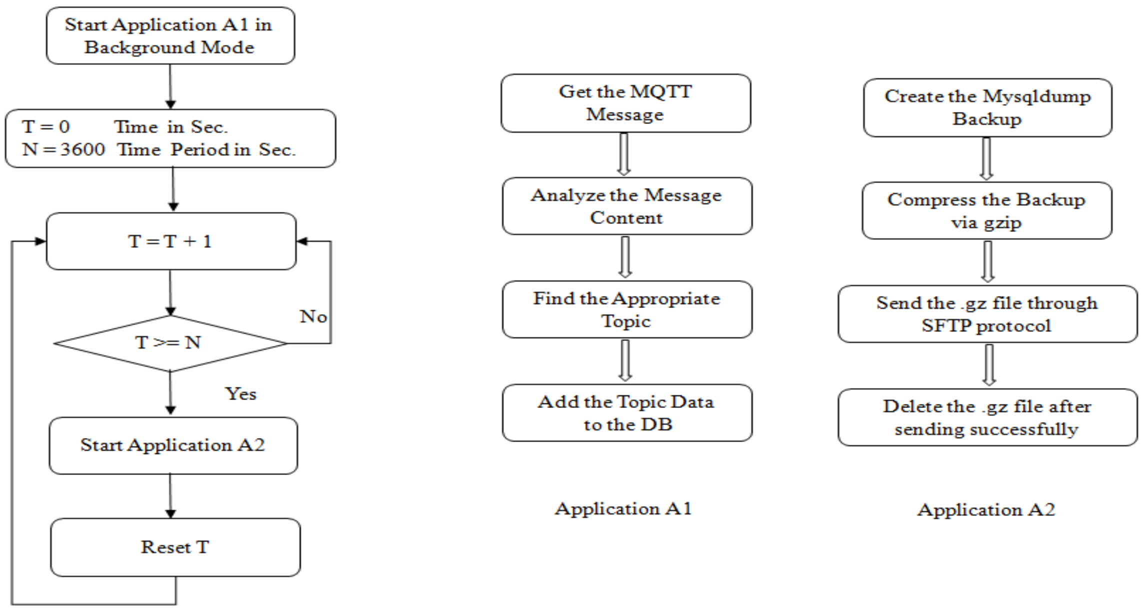
Start Application A1 in Background Mode
T = 0 Time in Sec.
N = 3600 Time Period in Sec.
T = T + 1
T >= N
No
Yes
Start Application A2
Reset T
Get the MQTT Message
Analyze the Message Content
Find the Appropriate Topic
Add the Topic Data to the DB
Application A1
Create the Mysqldump Backup
Compress the Backup via gzip
Send the .gz file through SFTP protocol
Delete the .gz file after sending successfully
Application A2

Fig. 4. Fog Node workflow diagram

Data from all Fog nodes are collected in the HLAN network and stored for further analysis to make predictions of disease complications based on disease history and laboratory tests. The main functions performed in the HLAN network are:

- The storing of data from doctors, caregivers, patient registration, laboratory test results, and disease history records in the hospital database;
- The analysis of data by machine learning methods and the implementation of predictions;
- The transfer of test results to the doctor and, if necessary, to the patient.

Single node Fog cluster architecture in HLAN network. The Fog cluster organized via Minikube includes the master node control resource that transmits the execution of commands via kubectl to the API server which is then redirected to the designated worker nodes (fig. 5).

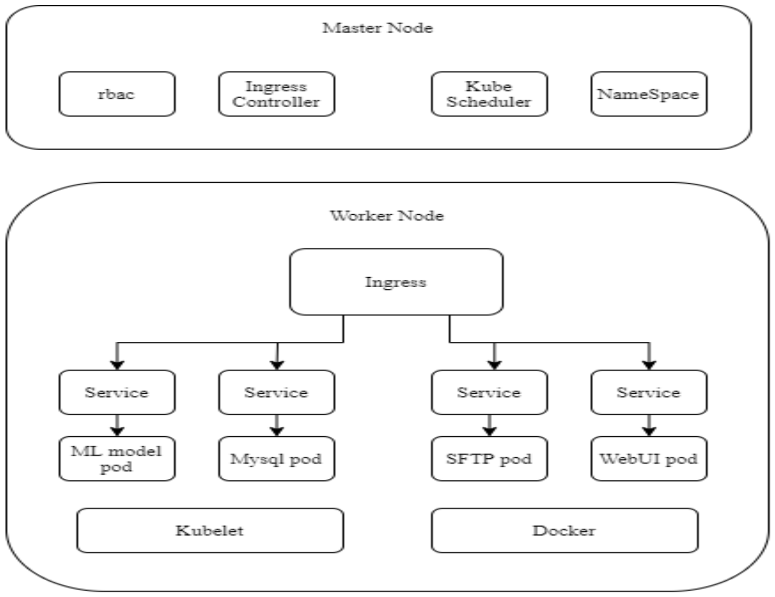

Fig. 5. Minikube Single node K8s components for HLAN network

The K8s worker node operates SFTP traffic processing, MySQL database operations, decision-making ML model, Web interface system micro-services. Each service is implemented as a container APIs and deployed in the Pod. The MySQL database stores the information transmitted through the SFTP protocol from the Fog node (PLAN network). The web interface provides information communication between the patient and his/her caregivers.

The single-node Fog cluster can be converted to a multi-node cluster if necessary.

Multi-node cluster architecture in HLAN network. Figure 6 shows the collaboration between Fog clusters and the main cluster in the multi-node Kubernetes ecosystem for the HLAN network. The Fog cluster has the microservices of the fig. 5 model. The ecosystem is highly secure as the following mechanisms are used: Kubernetes API access control, Transport Layer Security (TLS) for all API traffic, API authentication and authorization. [7].

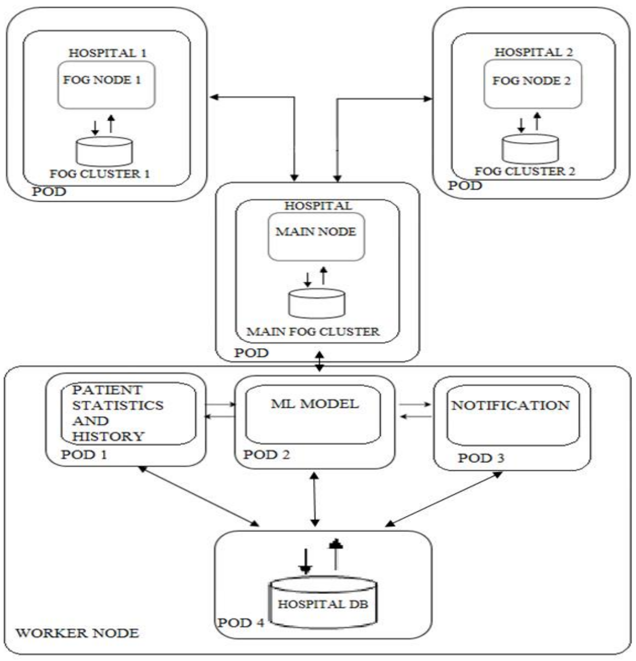

Fig. 6. Fog cluster architecture in the Kubernetes ecosystem

Conclusions. The paper shows the usability of Fog-based K8s container monitoring network architecture for personalized healthcare. Low power consumption, quick response rate, reduced network exploitation costs, and data privacy are the significant advantages of the proposed system.

The monitoring provides the following features:

- Patient monitoring and data collection from various sensors through the Fog node in the PLAN network;
- Data storing, processing, flow load management, creation of additional Pods with the same parameters, and the provision of high security and reliability in the HLAN network with Fog clusters through the K8s container platform.

REFERENCES

1. Alsaffar, A. A., et al.: An architecture of iot service delegation and resource allocation based on collaboration between fog and cloud computing. Mobile Information Systems, Article ID 6123234, 15 (2016).
2. Saeed, Ullah, et al.: Big Data in Cloud Computing: A Resource Management Perspective Scientific Programming, Article ID 5418679, 17 (2018).
3. X. Masip-Bruin, E. Marín-Tordera, G. Tashakor, A. Jukan and G. J. Ren, “Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems,” in IEEE Wireless Communications, vol. 23, no. 5, pp. 120-128, October 2016. doi:10.1109/MWC.2016.7721750.
4. X. Masip-Bruin, E. Marín-Tordera, A. Alonso, J. Garcia: Fog-to-Cloud Computing (F2C): The key technology enabler for dependable e-health services deployment. Future Technologies Conference (FTC) 2017. doi: 10.1109/MedHocNet.2016.7528425.
5. Mosquitto - An Open Source MQTT Broker. Retrieved from: http://mosquitto.org/
6. J. Santos, T. Wauters, B. Volckaert, F. De Turck, “Towards Network-Aware Resource Provisioning in Kubernetes for Fog Computing applications”, 2019, pp. 351-359
7. Retrieved from: https://kubernetes.io/docs/tasks/administer-cluster/securing-a-cluster/
8. Mosquitto Auth Plugin. Retrieved from: https://github.com/jpmens/mosquitto-auth-plug.git
9. Marco Calabretta, Riccardo Pecori, Massimo Vecchio, and Luca Veltri. MQTT-Auth: a Token-based Solution to Endow MQTT with Authentication and Authorization Capabilities. Journal of Communications Software and Systems, Vol. 14, No. 4, December 2018.