COMPARATIVE ANALYSIS OF CLOUD AND FOG ENVIRONMENT BASED ON NETWORK USAGE AND COST OF EXECUTION USING IFOGSIM

By:

Mohammad Saqibul Alam
21141041
Sadia Jebunnesa Jabin
21141033
Ajmain Alam
21141045

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

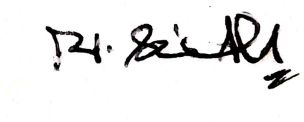
Department of Computer Science and Engineering
Brac University
June 2021

Declaration

It is hereby declared that

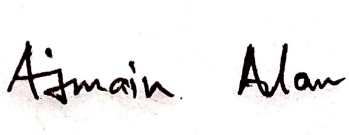
1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:



Mohammad Saqibul Alam
21141041

Sadia Jebunnesa Jabin
21141033



Ajmain Alam
21141045

Approval

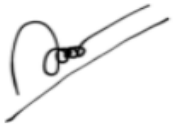
The thesis titled "Comparative analysis of Cloud and Fog environment based on network usage and cost of execution using iFogSim" submitted by

1. Mohammad Saqibul Alam (21141041)
2. Sadia Jebunnesa Jabin (21141033)
3. Ajmain Alam (21141045)

Of Spring, 2021 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of Bachelor of Science in Computer Science and Engineering on June 06, 2021.

Examining Committee:

Supervisor:
(Member)



Muhammad Iqbal Hossain, PhD
Assistant Professor
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Md. Golam Rabiul Alam, PhD
Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Ethics Statement

Whatever results we have found from this thesis is purely our original work. We have read several papers, journals and articles to help increase our knowledge on the topic but the findings and analysis are our own work. Materials from other sources have been properly cited in the thesis.

Abstract

The number of IoT devices around the world is increasing exponentially with the growth of technology. But the resources we have at hand may prove difficult to accommodate all of them. To make efficient use of all the resources, we need to ensure that we get maximum output while decreasing the network congestion. In this paper, we deal with such an IoT device: CCTV. In our digitised world, 24/7 surveillance has become prevalent through the use of CCTVs and the use of CCTVs will only keep growing. To better use the data from the CCTVs, we run it through video processing and analysis to extract the data we need. This computation is done on a local host machine which requires the machine to have high specifications to be able to perform such rigorous tasks. This means that a certain machine is dedicated to a certain number of CCTVs. In our paper, we have decided to introduce fog computing to the scenario. The CCTVs will send the footage to the Fog Servers instead of the local machine, where the necessary processing and analysis will be done and the results will be sent to the local machine. This, not only helps in reducing the cost of execution and total network usage from a traditional setup, allows multiple users to reuse the same computational resources in the fog server.

Keywords: Fog computing; Edge computing; IoT.

Acknowledgement

Firstly, all praises to Almighty Allah for blessing us with the opportunities to complete our thesis without any major obstruction in this time of a global pandemic.

Secondly, we would like to extend our heartfelt gratitude to our supervisor Dr. Muhammad Iqbal Hossain for helping us and guiding us, offering his support and advice whenever we needed it. We would also like to thank two other faculties, Jannatun Noor and Arif Shakil, from Brac University for sharing their knowledge of cloud computing.

Finally, we would like to thank our families and friends who have supported us in these trying times, offering us mental support.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Fog computing is a decentralized network structure that connects cloud computing to the Internet of Things (IoT). It agrees with the concept that almost all IoT devices that are used by people on a regular basis, will be interconnected with each other. Iot devices include smartphones, smart home appliances, smart watches, wearable health monitoring system, biometric cybersecurity scanners, fire detection and alarm system, smart door lock and burglar alarm system, lighting management system, smart bicycle, fitness trackers etc. But the sensors in IoT devices lack storage capability, resources to carry out computational workload and analyze data. With the Cloud storage in a distant geolocation, it is difficult to do the required task in a reasonable timeframe. Fog computing puts one and one together. It allows data transmission between IoT devices and cloud services to be processed faster by bringing them closer to one another. Concurrently, it determines whether the information is to be stored in the cloud or in the local hosts. Any device with computing and storage capability along with network connectivity can act as a fog node.

In our research, we hope to offload the computation needed to process and analyse the video footage from multiple CCTVs to a Fog Server instead of processing them locally. This will, hopefully, lead to a decrease in the cost of execution compared to a traditional Cloud setup.

1.1 Background

As the world is being digitised, the prevalence and necessity of CCTVs has become an important factor in ensuring constant surveillance to the public eye. But with the increasing number of CCTVs, there is also a demand for video processing and analysing. Normally, said processing would be done on a local machine (a high-specification computer). We are to see if offloading the computation to a Fog server would help the situation in any way. As per our research, we have not found any papers comparing the differences between a Fog setup and a Cloud setup, so we hope to fill in that gap.

1.2 Problem Statement

While using Cloud has provided us with innumerable facilities, with the ever increasing number of users and IoT devices, it gives rise to a variety of issues which include latency and bandwidth management, power management etc. For our paper, we are focusing on lowering the network usage and cost of execution in a CCTV footage analysis architecture. In a Cloud setup, the analysis of video footage from CCTV has to be done on a local machine(s). To meet the demands necessary for video processing and analysing, the local machine has to have high specifications. Imagine there is a vendor who uses 2 CCTVs for his store and another vendor who uses a single CCTV. In the traditional approach, each of the vendors would require two separate high-specification PCs to carry out any video analysing and processing, if they chose to. This introduces a lot of wasted resources that could rather have been reused by multiple users.

1.3 Research Objective

The purpose of our research is to decrease the overall cost of execution and the total network usage in a Cloud setup by implementing a Fog setup. This would allow users to offload the necessary video processing and analysis to the Fog servers and reduce the need to have a host machine with high specifications. This would also let multiple clusters of CCTVs reuse the resources in a single Fog Server at different time intervals. Carrying out the analysis practically would require a lot of resources that we do not have access to. So, for this research, we are working within the boundaries of a simulating tool iFogSim. We hope this will also benefit other areas where offloading computational tasks from end devices to Fog servers is possible.

1.4 Thesis Structure

Chapter 1 consists of 4 parts : Introduction, Problem Statement, Research Objectives and Thesis Structure.

Chapter 2 is the Literature Review. In this chapter, we have summarised the papers that are related to our work and have helped us with our analysis.

Chapter 3 is the Simulation Tool part where we talk about our approach to solving the problem at hand. We are using a simulation tool since we do not have access to the practical hardware required to carry out the research. In this chapter, we go into depths about how the simulation environment is set up and the specifications of the hardwares used as well as their physical and logical topologies.

Chapter 4 is the Results Analysis. Here, we talk about the results that we achieved from our simulation in Chapter 3.

We end the paper with Chapter 5 which is the Conclusion And Future Work. Here, we discuss some of the constraints we had to face and how this line of work could be updated in the future.

Chapter 2

Literature Review

Sheltami, Shahra and Shakshuki in their paper [1], talk about the characteristics of cloud computing and the architecture of fog computing and show some results to show as to why it is better than Cloud computing. They start off by defining the architecture of the fog paradigm and state some of the devices used in the fog layer. Some examples include routers, switches, servers, base stations(BS) etc. The authors define the process of how a mobile user communicates with the fog server and how requests are handled with respect to the end user, fog node and the cloud server. They finish the paper by showing results of the system response time, latency of response services, number of requests sent by users, power consumption and duty cycle. In terms of response time, they found out that to bring a service from the cloud servers takes the longest time (0.000016s), while bringing a service from a neighbouring node takes 0.000008s. And if the service already exists in the fog server, without any need to go to any other node, it takes 0.000002s, the shortest. The latency increases with increasing numbers of users as there is an increased demand for bandwidth. The number of requests sent by the users decreases. They show the power consumption in three different modes- Low Power Mode (LPM), Transmission mode (Tx) and Receiving mode (Rx).

Luan, Gao, Li, Xiang, We and Sun [2] talk about one the most prominent advantages of fog computing over cloud computing: location-awareness. They describe fog as a medium to bring services and applications faster to users, as mobile users communicate with the fog servers over a single-hop connection. With the concept of location-awareness, people's need for services can be predetermined, to some extent. So what they suggest is pre-catching location-relevant information in the fog servers, for example, flight information may be relevant to people in the airport whereas it would not make sense to store such data in a supermarket. With extra compute and storage nodes in the fog server, most of the computational load of a mobile-user is relieved to be done on the fog nodes. Scalability, extra management and maintenance cost, according to them, are some of the challenges of fog computing deployment.

Puliafito, Mingozzi and Anastasi in their paper [3] mentions the issue of mobility support on mobile nodes in a fog environment. Implementing mobility support results in service availability, efficient bandwidth consumption, lower latency and higher responsiveness as all the resource-intensive computing is done on the fog

server and IoT devices only send and receive data to and from the server. For flexible mobility, migration must be done on the topologically closest fog nodes considering migration cost and existing workload. CPU and bandwidth consumption should be limited and downtime should be minimized. To achieve service migration, techniques, such as virtualization and migration, methods to find a set of possible target fog nodes and to be able to adapt with the change of IP addresses when the end users are mobile and moving, are needed. In a stateless fog service, migrate the targeted fog nodes and redirect to the mobile nodes. In [4] and [5], the authors introduced the idea of FMC and FME which helps with user mobility but only in cellular networks. In [6], FMC is refined to help users connect to different networks but increases the threat of service as IP keeps changing when the users change location. In [7], the writers model and predict the user mobility through a 1D mobility pattern and model the service migration process as a distance-based MDP. In [8], Wang. et al. introduced a newer algorithm to reduce the complexity from O(N3) to O(N2) by changing to a 2D random walk mobility model. In [9], authors present SENGUE that performs migration by collecting key parameters, analysing and predicting QoS violations and determining the optimal node executing MDP.

Shanhe Yi, Zhengrui Qin, and Qun Li [10] talk about the security issues of the fog environment and the mitigation techniques. There are several issues with rogue nodes which can be mitigated [11] through a measurement-based method which enables a client to avoid connecting rogue access points. For various kinds of network security attacks, employment of SDN was proposed heavily which can help with IDS, and can influence OpenFlow. SDN can provide big differences in traffic isolation and prioritization, mainly so that the network does not get overflowed or congested. For secure data storage they talked about the public auditing for data that is already stored in the cloud which is done by a third-party auditor. Pinocchio, built for clients, so that they can verify general communications done by a server and they only have to count on cryptography. Data search can also be adopted to protect sensitive and subtle data files. [12] They have proposed to use attribute-based encryption so that they can use a good data access control scheme. They also talked about where IDS can be deployed in fog networks to detect malicious attacks like insider attack, flooding attack, port scanning etc.

Wang, Ning and Wang, in their paper [13], talk about transferring the computational workload of real-time traffic management systems to the fog to lower the average response time needed for vehicles to report events. The main issue they found with reducing the response time of traffic management servers is that the servers are centralized; it holds a lot of unnecessary information. For example, information about a traffic jam is relevant to the restricted area for a short period of time. So, the proposal of a decentralized traffic management system was put forth. The main challenges they faced in constructing such a system are that traditional traffic management schemes are centralized, the installing and placement of fog nodes requires extra cost and offloading network traffic optimally with limited computational resources. They proposed dividing a city into different regions and managing the traffic region-wise. They established a three-layer model, with the cloud layer, cloudlet layer and the fog layer. The cloudlet and few fog nodes coexist in each region, providing computational resources without needing additional

networking cost. Parked and moving vehicles are modelled as fog nodes according to queuing theory and treat the moving vehicles as a processing server based on an M/M/1 queue. Their proposed solution, with a time complexity of O(M4), divides the problem into two sub-problems and solves them iteratively. Their results show an improved average system response time and lower computational resources usage.

Ni, Zhan, Lin and Shen in their paper [14] talks about the vulnerabilities of fog assisted IoT devices and the potential challenges to secure fog computing. With the increasing number of real time applications of IoT, implementation of fog can increase responsiveness by 50% compared to the traditional cloud. However, due to inadequate security, IoT devices are more prone to numerous threats like forgery, sybil, DOS, MITM, and collusion. Several challenges are identified in securing the fog environment and some promising solutions are introduced in terms of real-time services, transient storage, data dissemination and decentralized computation. A variety of schemes have been proposed to solve the issues including lightweight cryptographic schemes, HIDS, NIDS, DIDS, homomorphic encryption: Paillier encryption and BGN encryption, differential privacy, atomic proxy cryptography, ABE: CP-ABE and KP-ABE, broadcast encryption, server-aided schemes etc. Yet, more issues like location privacy preservation, detection of rogue fog nodes and IoT devices, privacy exposure in data combination and decentralized and scalable secure infrastructure still remain for further research.

Kenitar, Arioua, Younes, Radi and Salhaoui [15] propose using the network coordinate system NC in order to imitate the highly latency networks. Real-time bandwidth vulnerable applications in denely distributed IoT devices. By considering the energy utilization equation as a vector 1D and bandwidth optimization equation as a sum of 1D vectors, 2 pairs of equations were developed for both fog and cloud. Energy transmission of the unit packet between end nodes (,) and total data size (DN, DC) for a request (k) were taken as parameters for the energy consumption equation of fog and cloud respectively. f(t) = .(DN(k) DC(k)) and cld(t) = (+).(DN(k)). The delays in unit byte data transmission (,) from end nodes to the fog and cloud, storage (s) and processing (p) were taken into consideration for the latency equation for fog and cloud respectively : f(t) = .((DNs(k) + DNp(k))(DCs(k)+ DCp(k))) and cld(t) = (+).((DNs(k) + DNp(k). The proposed concept model contains physical, middleware and cloud packaging. All the nodes are presumed active and data is sent to a fog enabled device. However, the bandwidth is lower than traditional cloud architecture. The developed algorithm works by importing data, computing the bandwidth transmission of data by per unit and computing the data that needs to be forwarded. In the same manner, the amount of energy consumption is determined. Size of the data and the number of nodes were adapted as variables to check the effectiveness in respect of bandwidth and the energy utilization in fog and cloud architecture. The results indicate a decrease in transmission of latency and energy optimization scenarios where fog architecture was used.

Caiza, Saeteros, Oñate and Garcia [16] performed a systematic review of fog computing being considered as a new component of industrial 4.0. The authors researched the characteristics and challenges that must be considered for fog implementation by

analyzing the architecture, bandwidth utilization, power optimization and security. The architecture highlights 3 layers which allow the fog to collaborate with cloud enabled services. They propose implementation of a load balancing algorithm in order to minimize processing time of different priority data by collaborating nodes. Moreover, with the help of redundant systems, fog servers can also perform tasks. In terms of security, new measures should be considered for access control and encryption. Security needs to be implemented directly to the fog nodes as it is decentralized and more vulnerable to attacks. Latency should be given the highest attention while implementing fog at the industrial level. However, optimization and learning algorithms developed for the IOT environment to improve latency cannot be considered at industrial level with different scenarios. Battery restriction of fog nodes is one of the highlighted drawbacks of implementing fog in IIoT architecture. For optimized energy consumption of fog nodes, WSN configured topology has been proposed for distribution. Yet scope of development in research and application of fog at an industrial level remains.

Memon, Maheswaran, in their paper [17] , talked around utilizing machine learning models to optimize the transmission capacity and information transmission between cloud and haze hubs. They utilized JAMScript, which could be a mist computing middleware that permits one to count the real-world enhancements in transfer speed, inactivity and vitality viability as well as the gage overheads incited by the framework. In this report they utilized the KERAS API with a Tensorflow backend to contraption the outfit of LSTMS. For relapse, they moreover utilized Python Scikit-learn. Each base learner's structure was changed with a different one. The distinctive number of units and diverse dropouts in each stack (a three information stream stack) estimating for regularization. Each show has been prepared for 500 ages, no overfitting has been found over this skyline of planning. Preparing ended since there was no discernibility. They moreover found a decrease within the botch. The result was with an LSTM based outfit. It surpassed the neural network and ARIMA based representations by noteworthy margins.

Li, Zhang, Jin, Yang, Yuan, Gao [18] communicated approximately the idleness estimation for mist based IoT. They utilized the NCS calculation which surveys idleness based on recognized arrangements in computer-generated geometric vectors. They utilized Point of interest based facilitated frameworks which were GNP and NPS, they moreover utilized Dispersed Arrange Facilitate Framework which were Vivaldi and Pharos to urge an adequate result. Numerous NCS calculations accomplished more than 90 percent precision of idleness estimation on the web. They moreover utilized records which accomplished around 90 percent precision with 13 or more points of interest, whereas 16 points of interest are required to attain comparable exactness in GNP

F.H. Rahman et al. [19] calculated the cost of execution for scaling out and scaling up methods using iFogSim. They created a scenario with Cloud, Proxy, Fog and any IOT device (camera in this use case) and calculated the variables. The cloud execution cost was significantly reduced by around 4.6635 % for the scaling out method and decreased around 5.4197 % while they were taking the scaling up approach. While scaling up means, for the end user's obtainability must be assured.

They also talked about the security for a heterogeneous fog network by means of the scaling out methodology and how cyber security threats like network intrusion, information leakage and DOS would ascend because the fog environment is being shared by many manipulators.

Valentin et al. [20] talked about the FIWARE platform which will act as a middleware stand. According to the European Union, it was made with the idea for expansion and worldwide arrangement. They also discussed a prototype which is executed on FIWARE. In that, they showed how the preprocessing can be done with cameras and computers. They showed relevant information about video processing errands and for posterior study.

Shukla et al. [21] proposed that we could use cloud computing and IoT for the healthcare industry which will boost the medical technology industry. The doctors will be able to connect with the patients sufficiently and more adequately. The huge amount of medical data that is being generated all over the world, could be easily stored and processed through Cloud. The data needs to travel distances away from the end-users or IoT and if it's saved on a cloud platform, it will be much faster. They projected an analytical model description and tried to reduce the latency of the whole system. They suggested a layered approach which can benefit both the doctor and the patient for swift communication. They simulated using SPARK which is an open-source software used for managing big data streams in a real-time situation. They found that using fog decreases the latency compared to the cloud environment.

Chapter 3

Simulation Tool

We are comparing the network usage in the topologies of a Cloud vs Fog. In the Cloud topology, we assume that the video processing and analysis is done on a local machine and data is sent to the Cloud purely for storage purposes. In the Fog setup, however, we offload the computational video processing to the fog servers and only the result data is sent to the local host. We have decided to use iFogSim to achieve the simulation.

3.1 iFogSim

3.1.1 Introduction to iFogSim

iFogSim is a Java-based simulation toolkit used to understand latency in an end to end connection, congestion in a network, power consumption and other factors in a fog environment. In identifying such factors beforehand, it helps to build the proper setup in the future. According to the authors [22], iFogSim has 3 components:

- Physical components: Fog devices are an example of physical components,where the lower level fog devices are connected to sensors and actuators. Compared to a cloud setup, Fog devices play the same role as data centers where they offer computational resources, network and memory

- Logical components: These include the AppModules(Application modules) and AppEdges. AppEdges define the dependency between two modules. Going back to a cloud setup, the AppModules are mapped with VMs and AppEdges dictate the logical flow of data between VMs.

- Management components: The Controller and Module objects are the management components in iFogSim. The accessible resources of a Fog Device are recognized by the Module Mapping object as per the specifications of the AppModules. If a Fog Device is not able to meet the demands of a module, the module is forwarded to a higher level Fog Device. The Controller object initiates the AppModules on their designated Fog Devices after being placed by the Module Mapping object. After the simulation is done, the Controller object collects the results which contain the consumption of energy, usage of network and cost during the simulation.

3.1.2 Topologies of Fog and Cloud

In the Cloud paradigm, users connect to a data server over the internet, where information is stored and accessed anytime and from anywhere with an internet connection. In the Cloud architecture, there are mainly two parts: the Front End, where there is the Client's machine, and the Back End, which consists of the different components that make up a Cloud environment like Storage, Servers, Networking etc. The communication between the two parts is done over the internet.

The Fog paradigm works hand in hand with the Cloud, where the devices are arranged in a three-step hierarchy. End devices at the lower level have IoT actuators and IoT sensors connected to them. The gateway devices are the fog servers that are placed between the cloud and the end user. To make matters easy, we have considered the fog devices of the same level to be homogeneous and have maintained the same sensing frequency for all the sensors.

Figure 3.1 illustrates the physical topology of a fog setup, where the IoT devices communicate with the Cloud through the Fog servers.

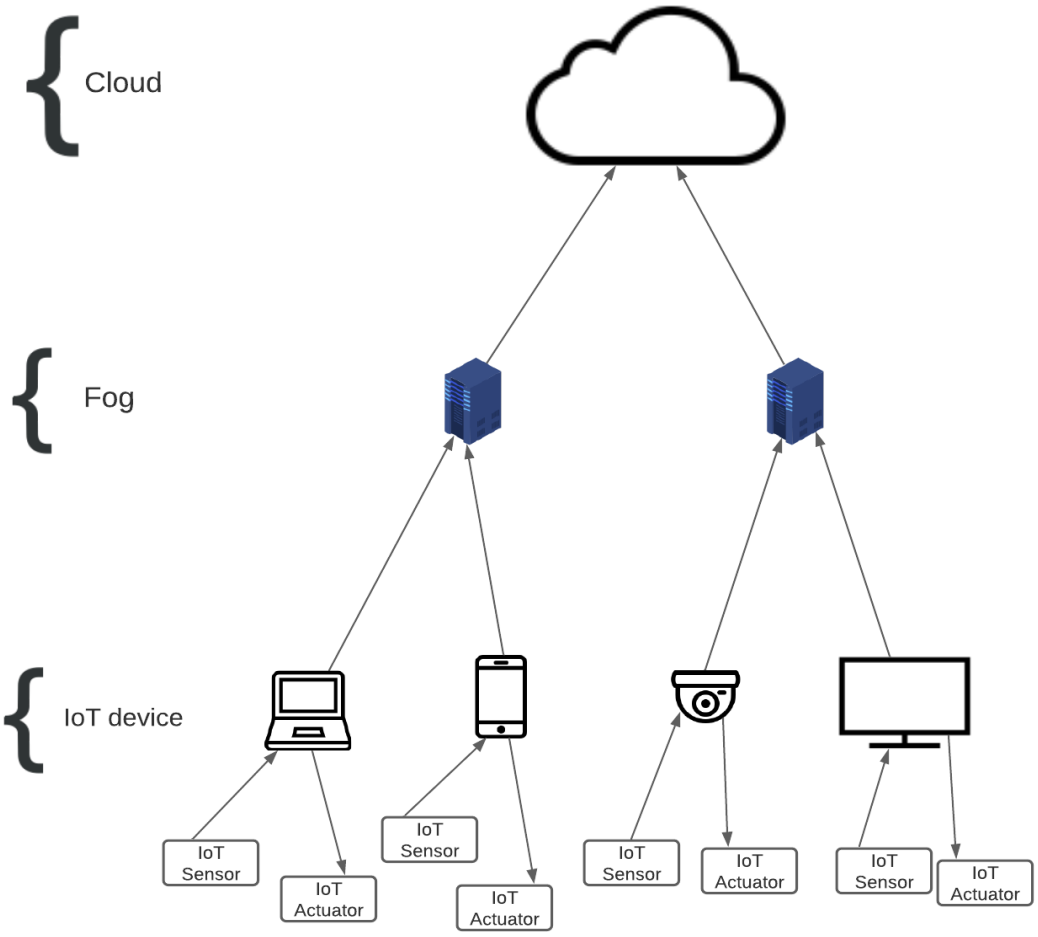


Figure 3.1: Physical topology of a fog setup

Figure 3.2 depicts the logical topology of a general fog setup. It is assumed that the ClientModule in Figure 3.2 is set in the end devices and the StorageModule in the cloud. The MainModule is placed in the fog server and needs a set level of computational resources to start. If other end devices require extra resources to perform tasks in a given time, they do so by requesting adjacent fog nodes.

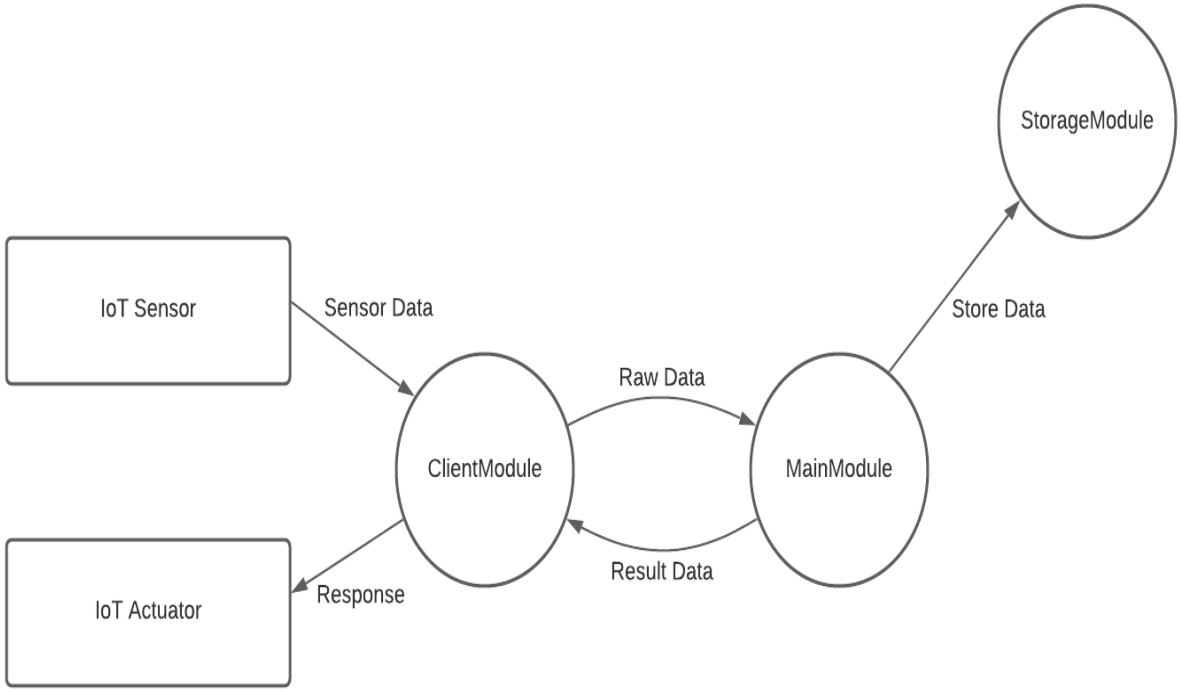


Figure 3.2: Logical topology of a fog setup

MainModules are placed in the gateway Fog Device where the deadlines of different connected end devices and the resource availability of the Fog Device are identified. This flow of tasks is shown in Figure 3.3. We start by placing the MainModule in a gateway Fog Device and determining all the adjacent Fog Devices. Then the deadline based QoS and the resource requirement of the Fog Devices are implemented. Then, we determine if the current Fog Device has the resources available to meet the earliest QoS deadline. If it does, then we place the module in the Fog Device and keep repeating it if there are other modules left to be placed. If, at any point, the Fog Device does not have the necessary resources available, the modules are sent to the Cloud to be processed.

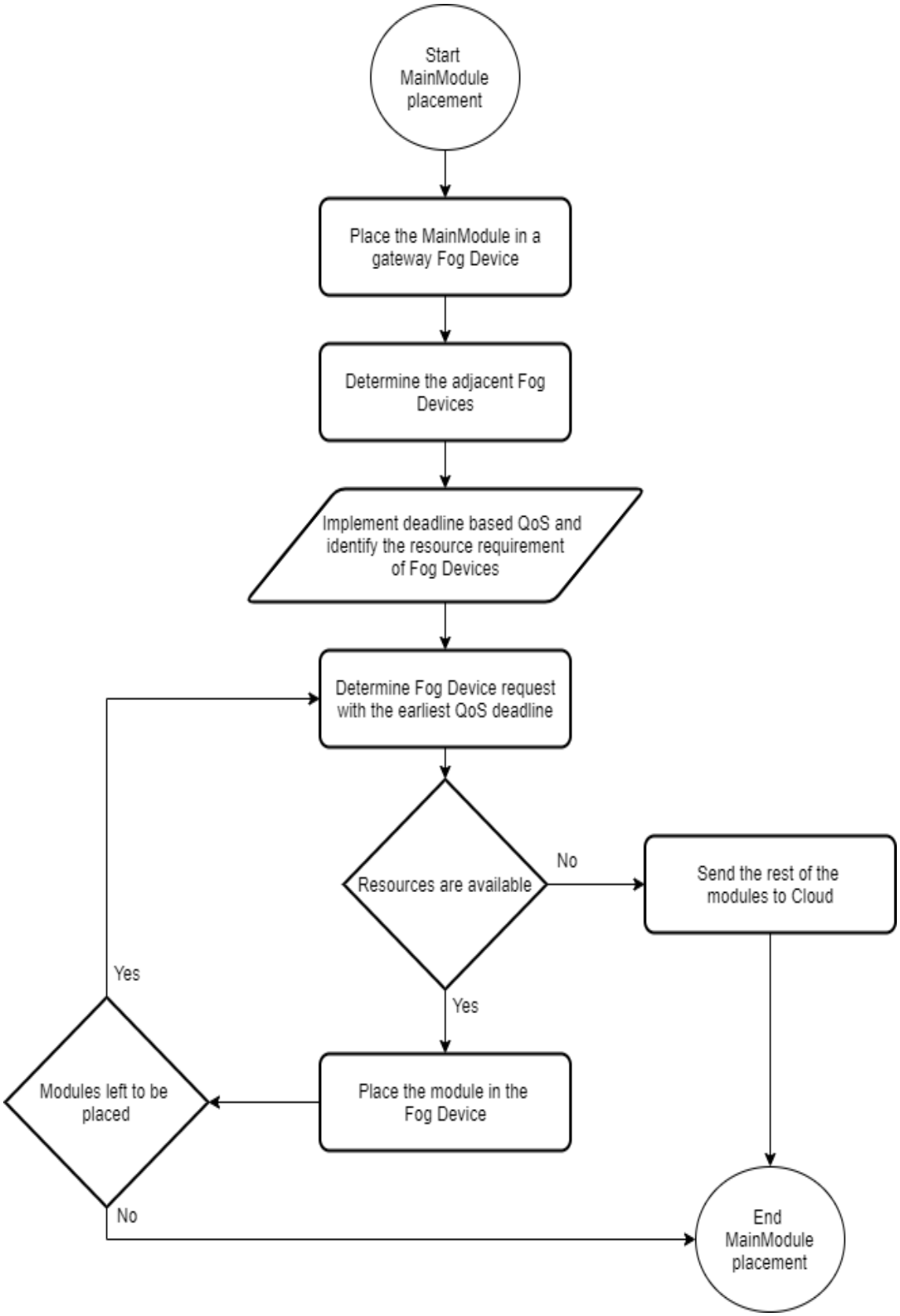


Figure 3.3: Flowchart of the module placement in a fog environment

3.1.3 Application in a real life scenario

To compare the results of the two approaches, we have decided to study a CCTV system, where footage from the IoT sensor is sent to the camera which is further processed either locally or on a fog server.

For the Cloud setup, as shown in Figure 3.4, we have considered that all the CCTVs are connected to a local host in the LAN, say an admin's PC. The footage is then analysed for facial detection or whatever the user might intend to do, in the Admin PC. After the processing is done, the result is shown on an IoT actuator (the user's monitor) and data is sent to the Cloud just for storage. We assume that the Admin PC acts as a gateway as the data to be stored in the Cloud is sent from here.

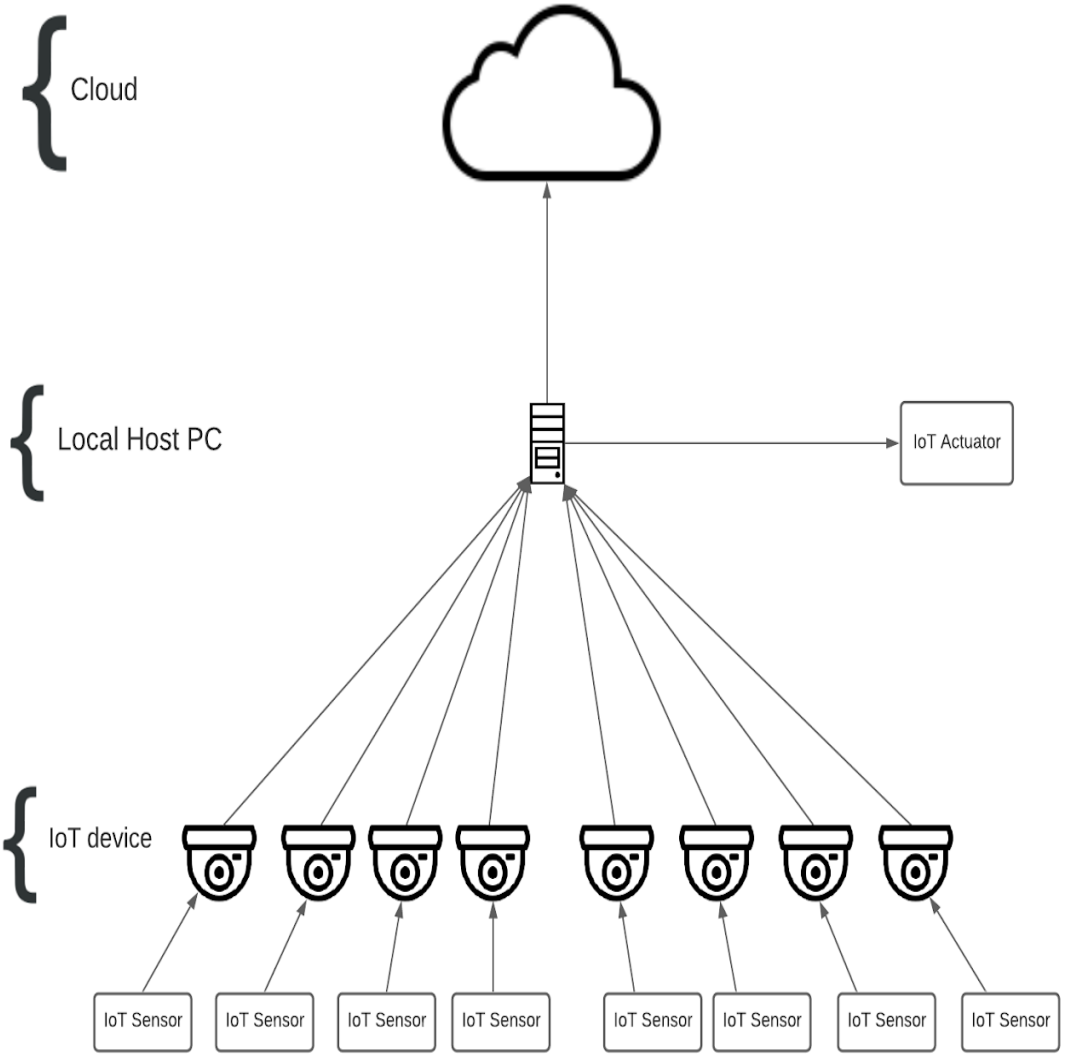


Figure 3.4: Physical topology of the Cloud CCTV setup

Table 3.1: Specifications of hardware in the Cloud CCTV topology

Parameter	Cloud	Admin PC	CCTV
Level	0	1	2
CPU length (MIPS)	44800	3500	1500
Rate per MIPS	0.01	0	0
RAM(MB)	40000	8000	1000
Downlink Bandwidth	10000	10000	-
Uplink Bandwidth	100	10000	10000
Idle Power	16*83.25	83.4333	82.44
Busy Power	16*103	107.339	87.53

In the Fog setup, as shown in Figure 3.5, a fog server is dedicated for every 4 CCTVs. CCTVs are connected to the fog server where there is a higher computational capability than in the local host. Data is processed in the server and the result is sent to the local host (Admin Pc) and shown on an IoT actuator. Data is sent to the Cloud for storage. This allows for the Admin PC to not have high-end specifications which the data analysis and processing might require, lowering the cost in that sense.

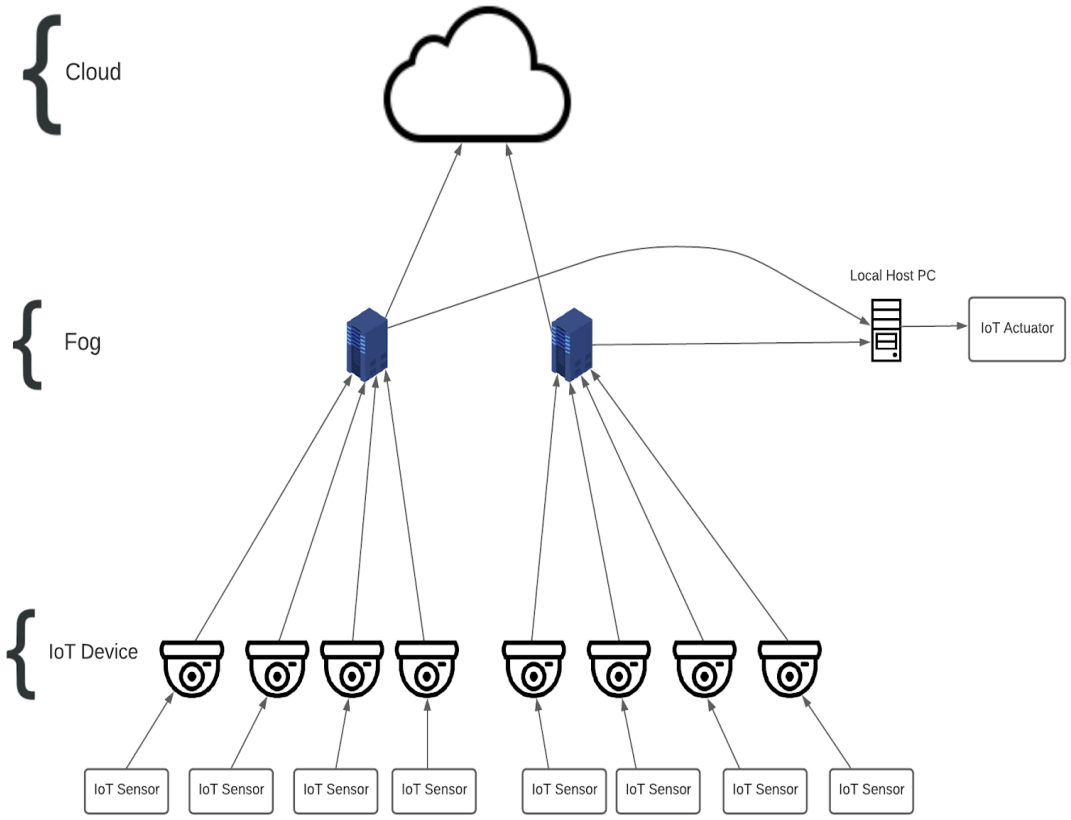


Figure 3.5: Physical topology of the Fog CCTV setup

Table 3.2: Specifications of hardware in the Fog CCTV topology

Parameter	Cloud	Fog Server	CCTV
Level	0	1	2
CPU length (MIPS)	44800	2800	1500
Rate per MIPS	0.01	0	0
RAM(MB)	40000	4000	1000
Downlink Bandwidth	10000	10000	-
Uplink Bandwidth	100	10000	10000
Idle Power	16*83.25	83.4333	82.44
Busy Power	16*103	107.339	87.53

In the logical topology of the Fog setup Figure 3.6, the IoT sensor in the CCTV sends the SensorData to the CCTV module. The CCTV sends the RawData to the Fog Server module for processing. After it is done, the Fog Server sends StoreData to the Cloud (for storage of data) and ResultData to the Admin PC. The Response is then shown on the IoT Actuator of the Admin PC.

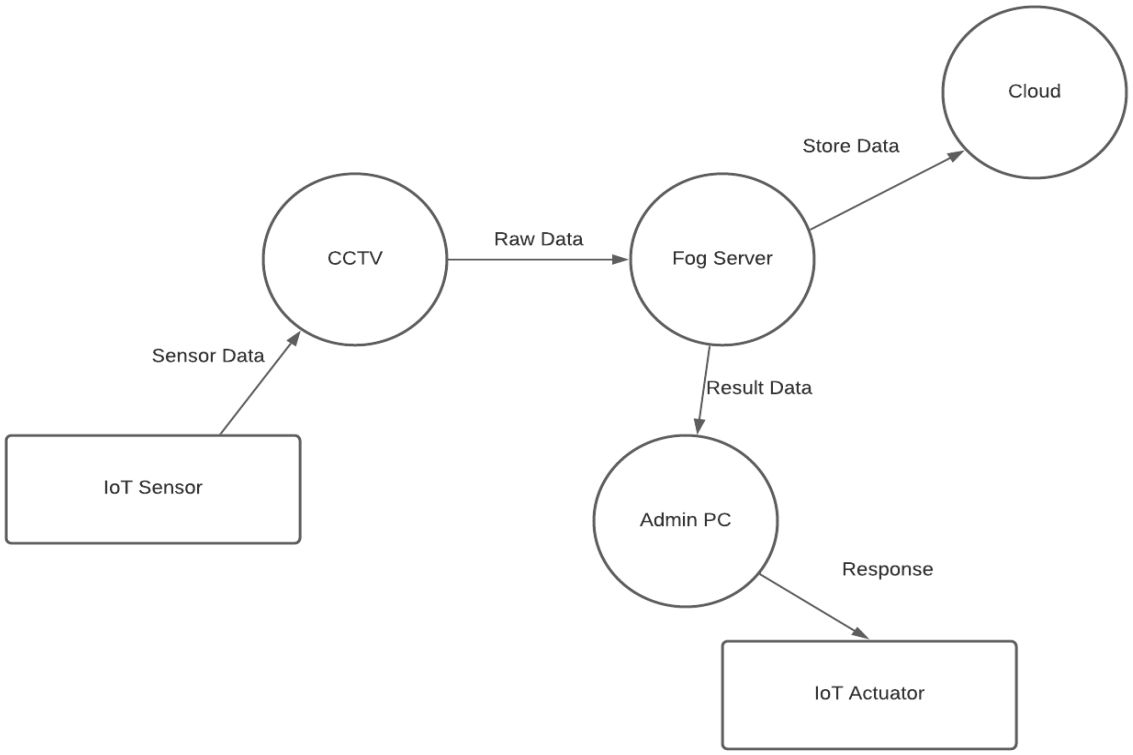


Figure 3.6: Logical topology of the Fog CCTV setup

In the logical topology of the Cloud setup Figure 3.7, The IoT sensor in the CCTV sends SensorData to the CCTV module. The CCTV module then sends the Raw-Data to the Admin PC module where the data processing and analysing is done in the same module. After computing the ResultData, it sends StoreData to the Cloud and the Response is shown on the IoT Actuator of the Admin Pc.

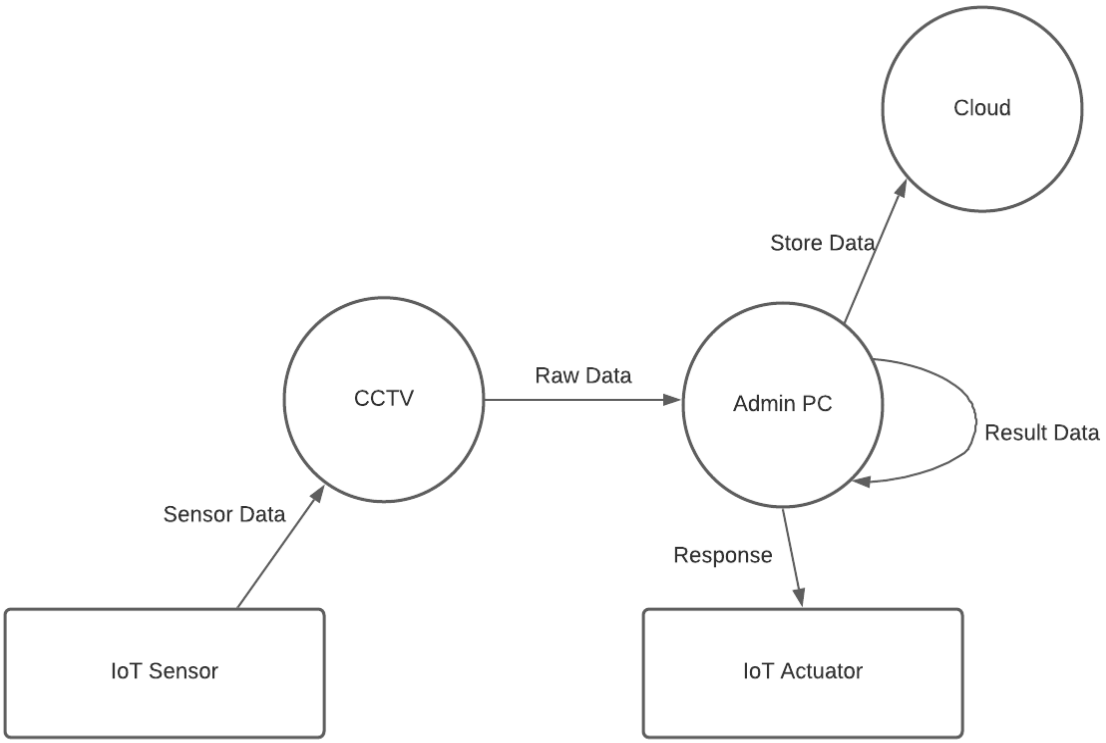


Figure 3.7: Logical topology of the Cloud CCTV setup

Chapter 4

Results Analysis

We wrote the code in iFogSim with the specifications given in Table 3.1 for the Cloud topology and Table 3.2 for the Fog topology. The code was run for 4, 8, 12, 16, 20 and 24 CCTVs in both the environments.

In the Fog environment, each fog server is allocated with 4 CCTVs. So, for 4 CCTVs, there was 1 fog server. For 8 CCTVs, there were 2 fog servers and so on.

For the Cloud environment, however, all the computational work is done on the local host PC. So, 4,8 or 24 CCTVs, they are connected to the 1 local PC.

After running the code in iFogSim, the results tell us the execution time, the Application Loop Delays, the Tuple CPU Execution Delay and other variables. An example of how the results look like are given below:

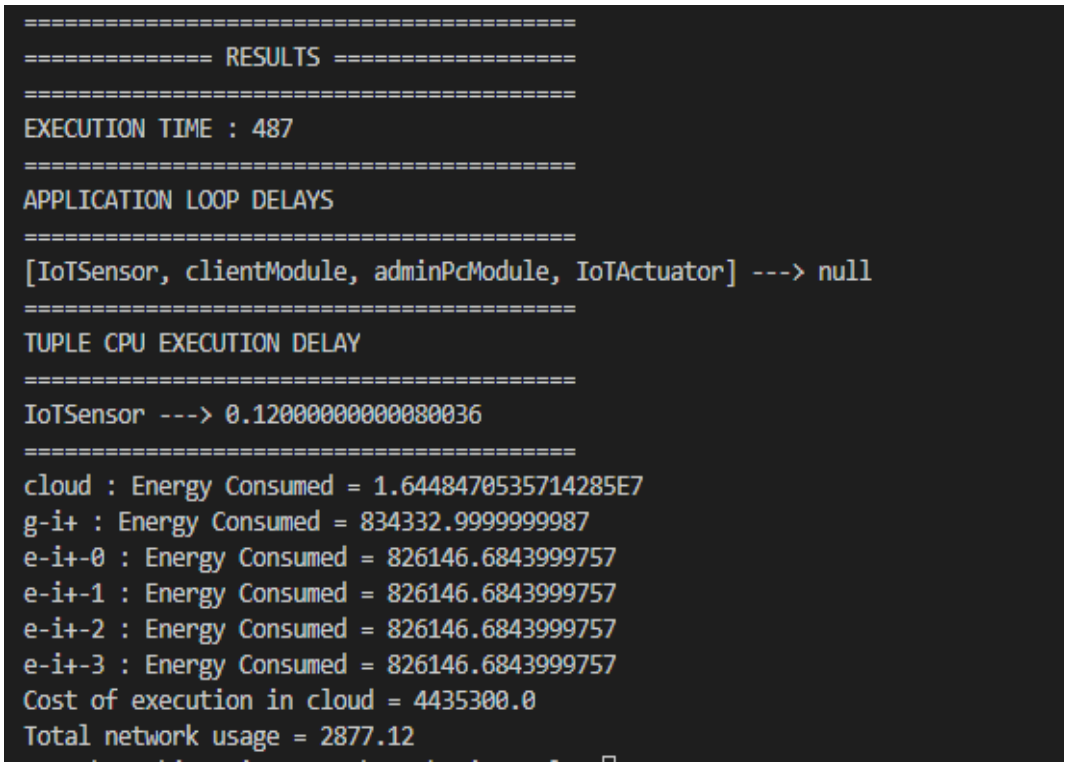


Figure 4.1: Results of running 4 CCTVs in a Cloud topology

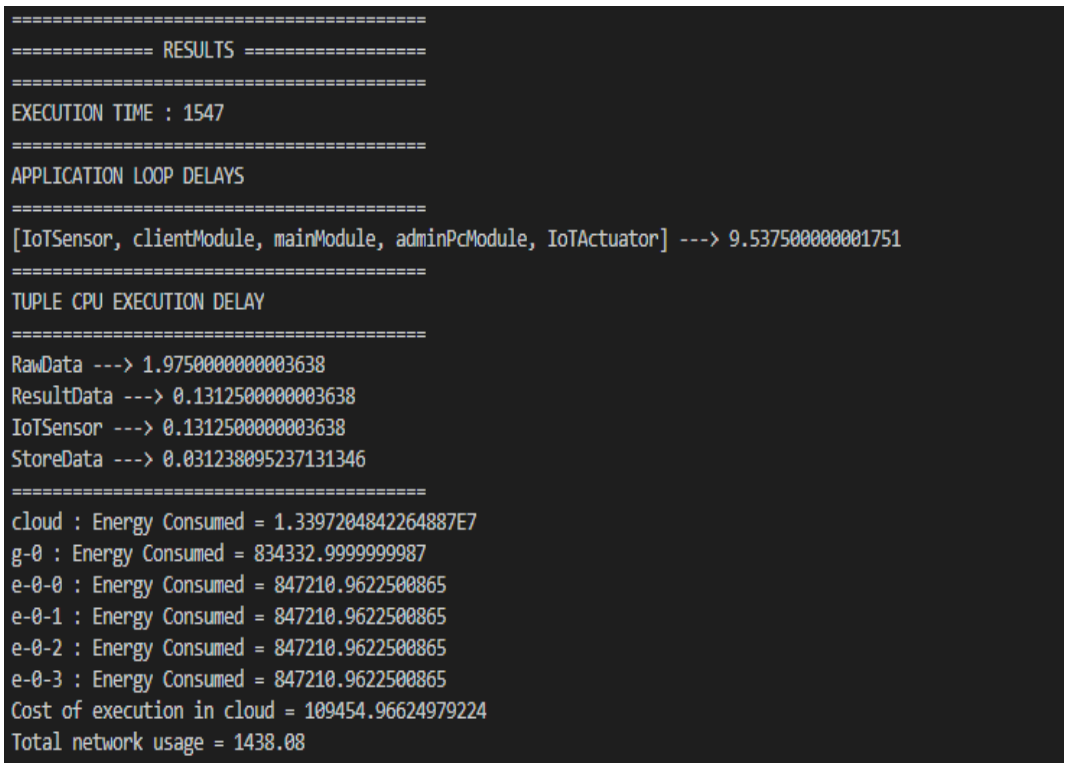


Figure 4.2: Results of running 4 CCTVs in a Fog topology

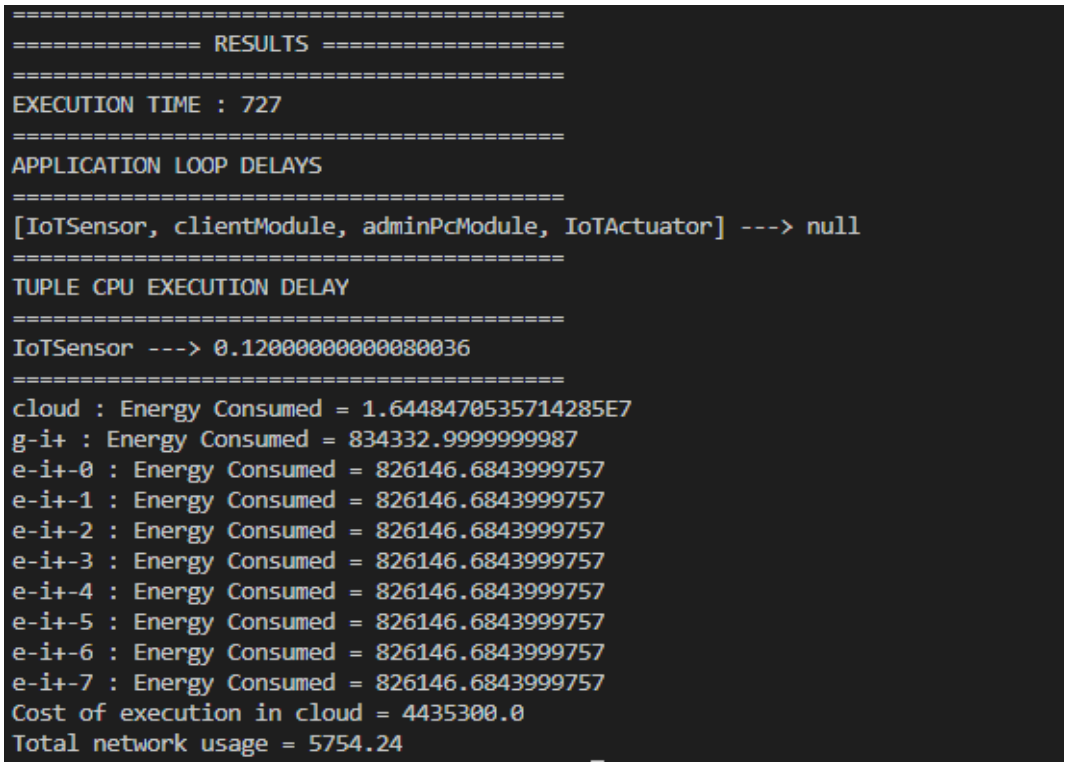


Figure 4.3: Results of running 8 CCTVs in a Cloud topology

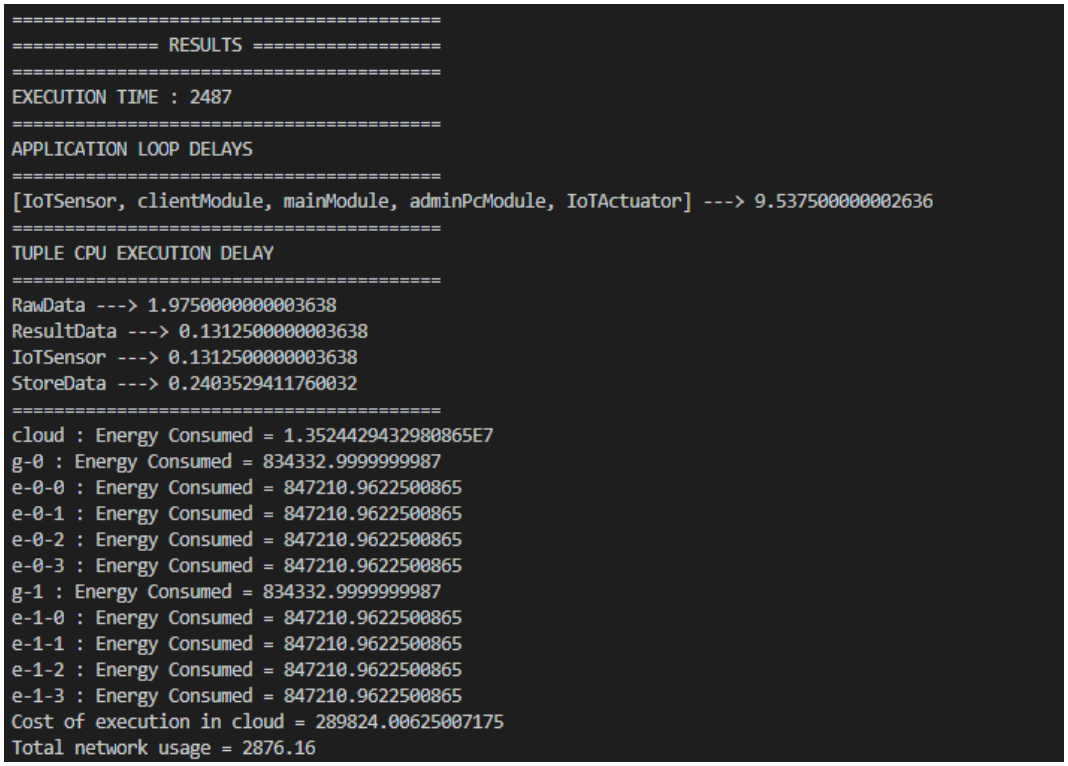


Figure 4.4: Results of running 8 CCTVs in a Fog topology

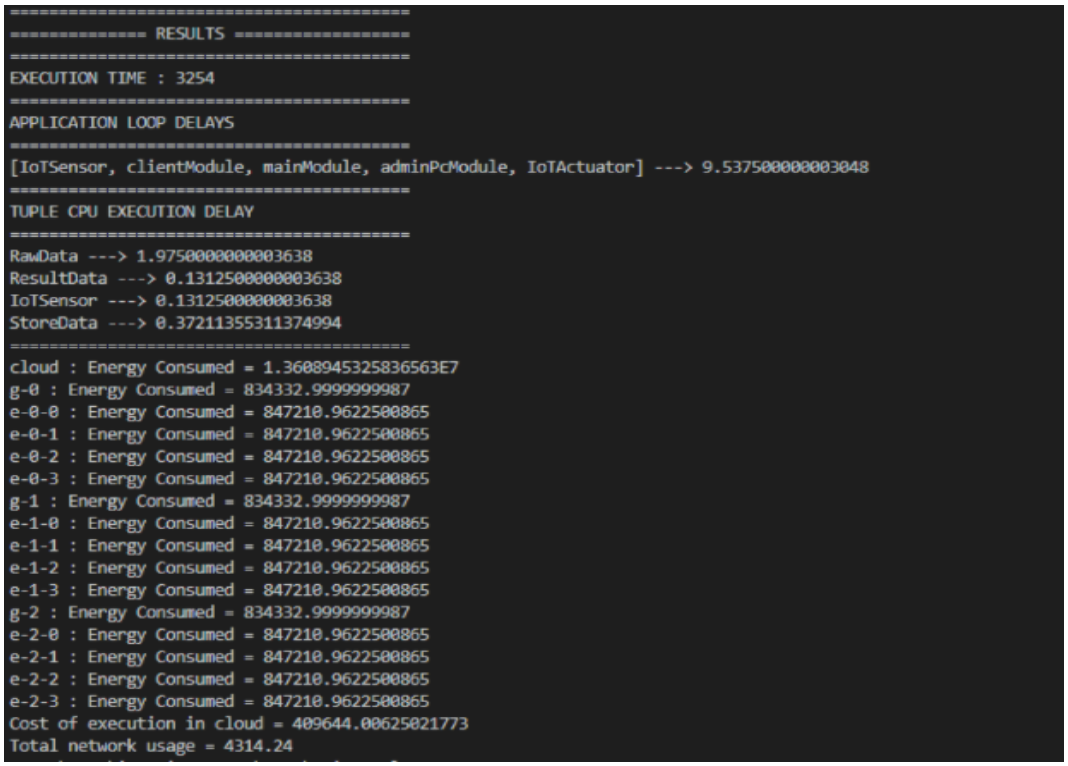


Figure 4.5: Results of running 12 CCTVs in a Fog topology

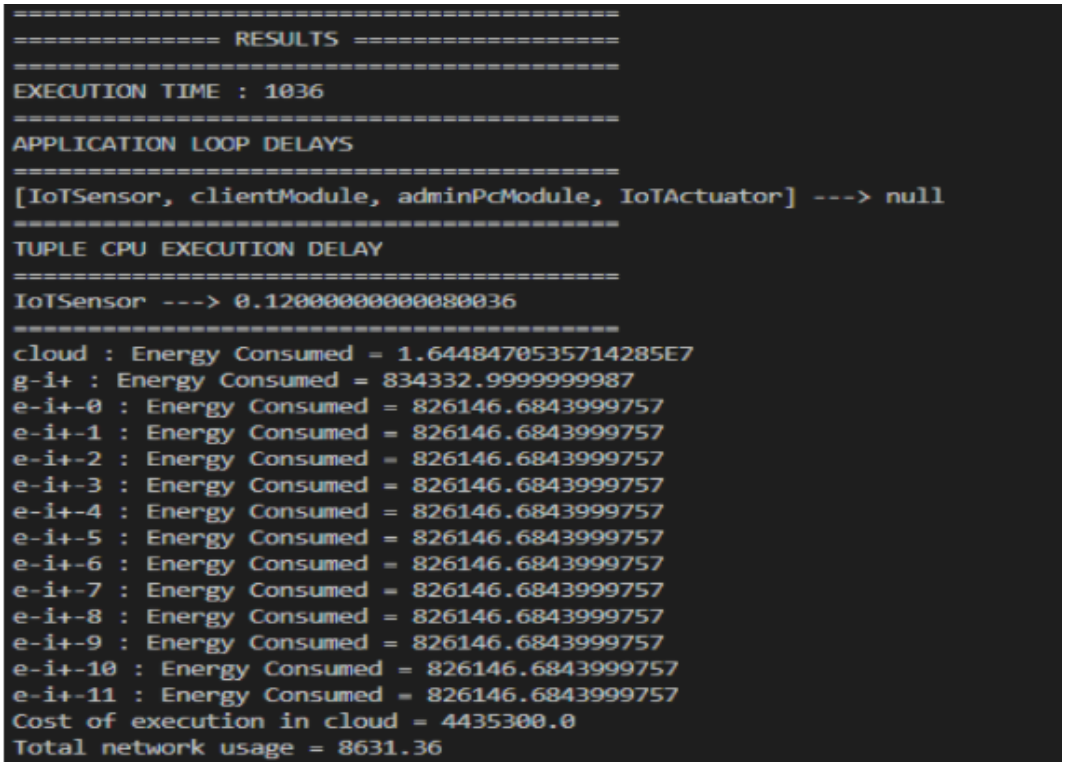


Figure 4.6: Results of running 12 CCTVs in a Cloud topology

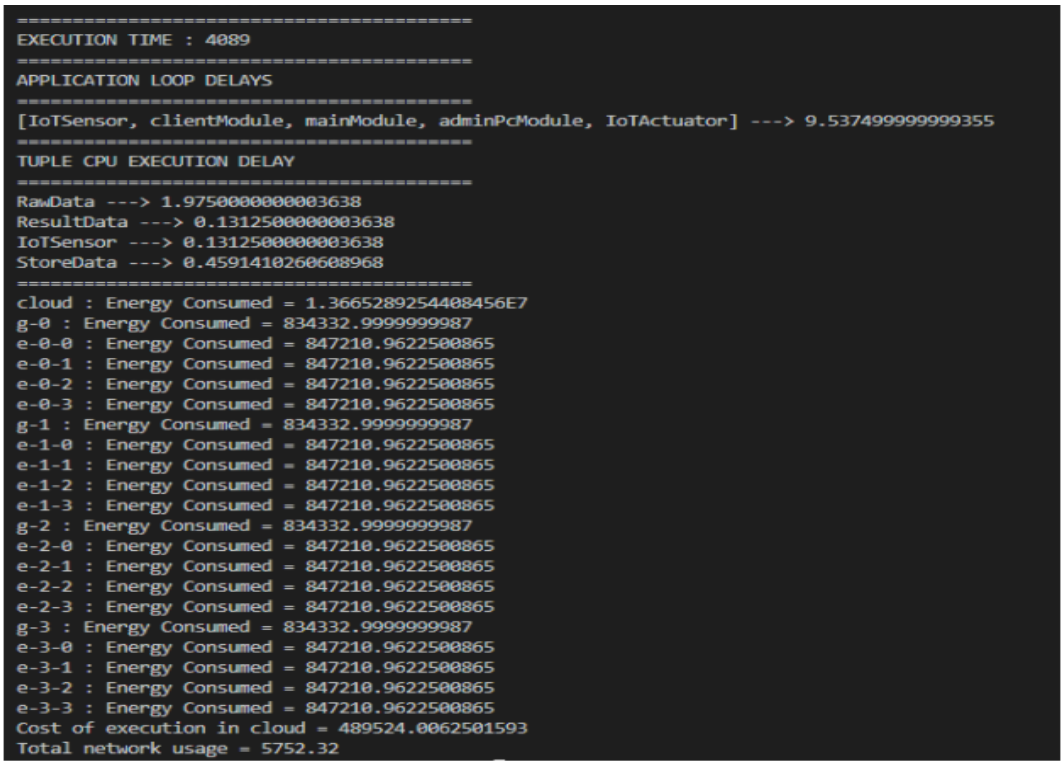


Figure 4.7: Results of running 16 CCTVs in a Fog topology

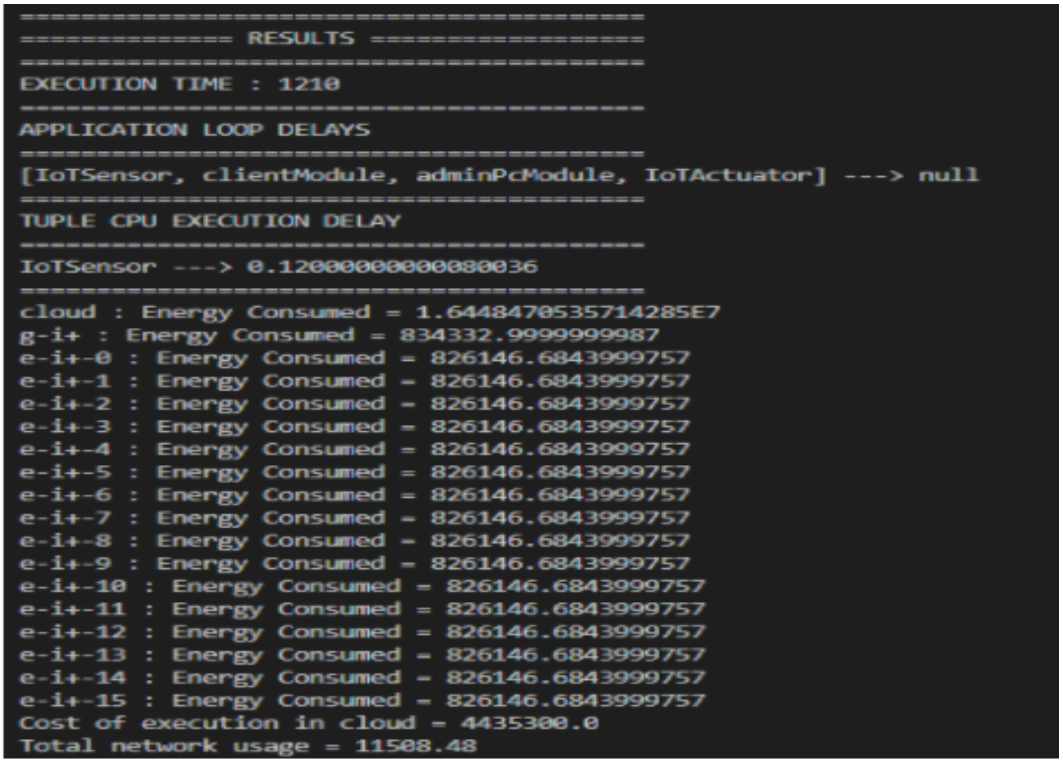


Figure 4.8: Results of running 16 CCTVs in a Cloud topology

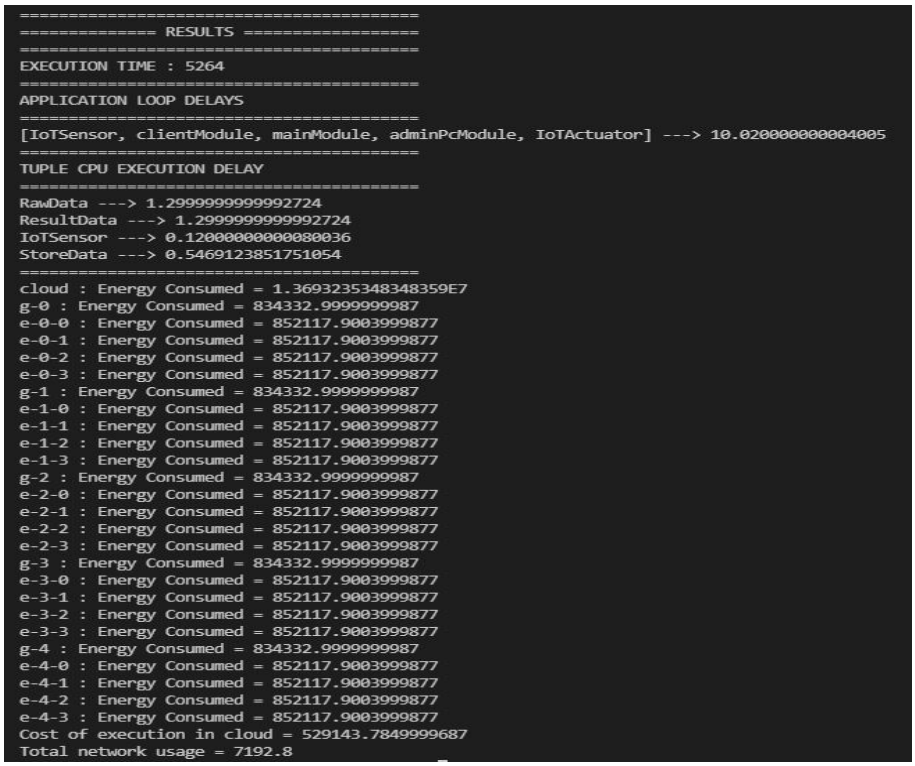


Figure 4.9: Results of running 20 CCTVs in a Fog topology

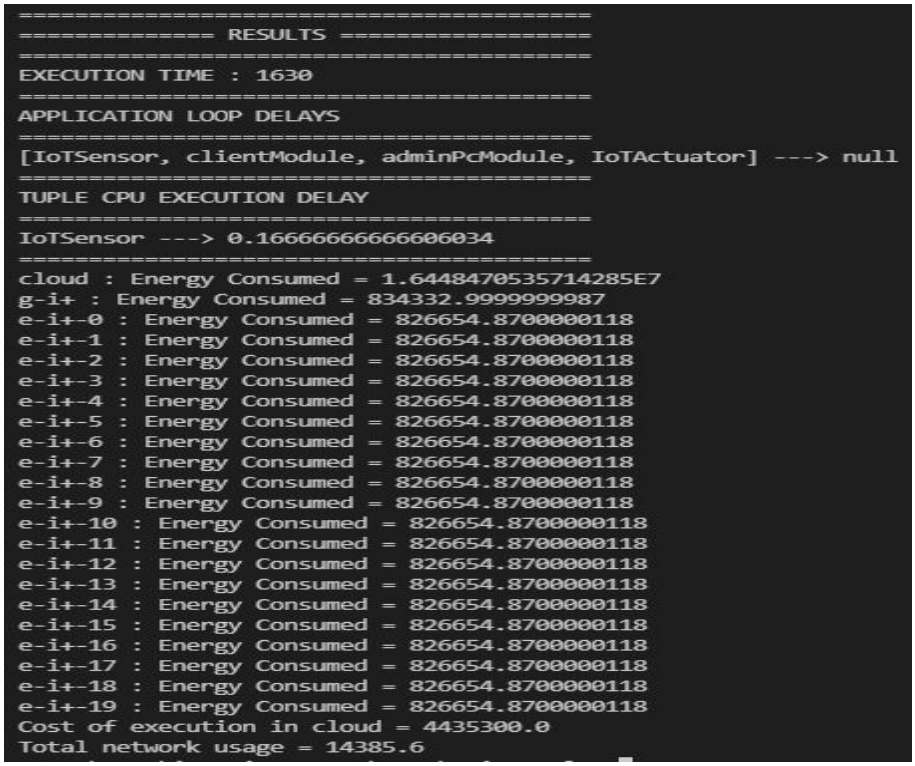


Figure 4.10: Results of running 20 CCTVs in a Cloud topology

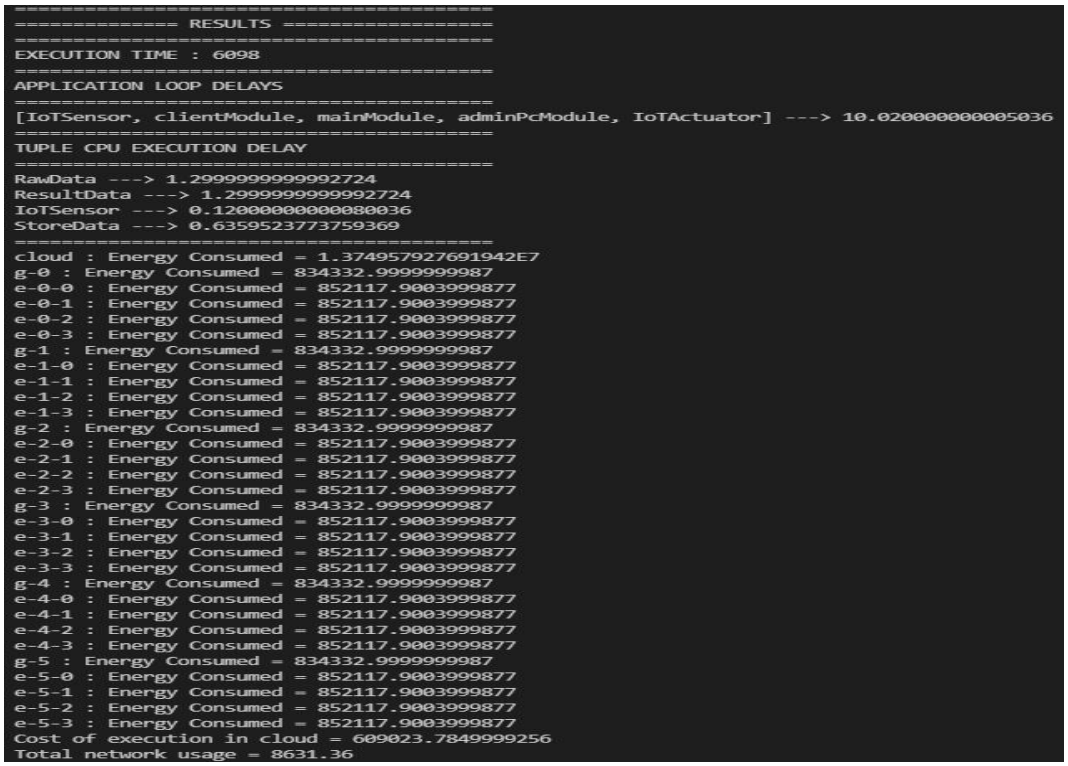


Figure 4.11: Results of running 24 CCTVs in a Fog topology

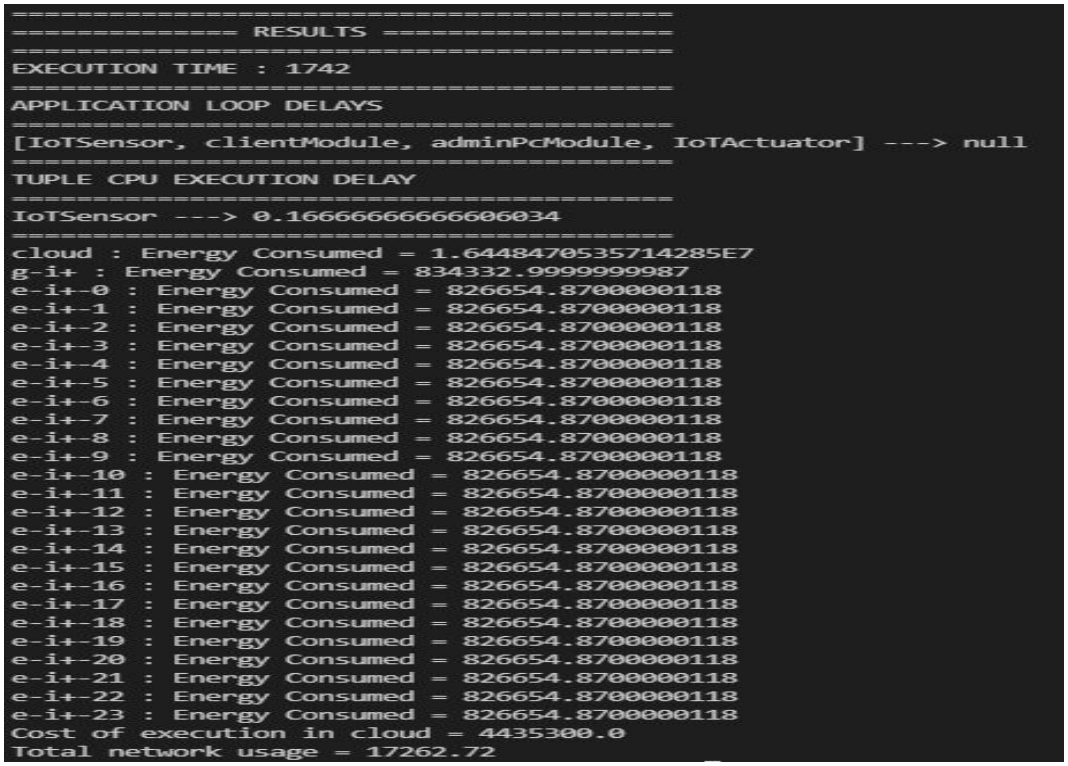


Figure 4.12: Results of running 24 CCTVs in a Cloud topology

For our paper, we are looking at the cost of execution and the total network usage, shown in the bottom two lines of the results from running the code, and how they differ from a Fog topology to a Cloud topology.

In [23], the writers describe the total execution cost as

$$\mathrm{E}= \mathrm{T_c}+ (\mathrm{C_i} \times L_{\mathrm{time}} \times \ R_{\mathrm{MIPS}} \times \ L_{\mathrm{u}} \times \ T_{\mathrm{MIPS}})$$

where $\mathrm{T_c}$ = execution cost, $\mathrm{C_i}$ = CloudSim clock, $\mathrm{L_{time}}$ = last utilization update time, $\mathrm{R_{MIPS}}$ = rate per MIPS, $\mathrm{L_u}$ = last utilization and $\mathrm{T_{MIPS}}$ = total MIPS of the host.

The results of running the simulation for CCTVs numbering from 4,8,12,16,20 and 24 in a Cloud environment are given below. The execution time and network usage increases with the increase in the number of CCTVs. The application loop delay stays null throughout all the simulations and the cost of execution stays constant at 4435300.

Table 4.1: Result Analysis of the Cloud Environment

Number of CCTVs	Execution time	Application loop delay	Cost of execution	Network usage
4	487	-	4435300.0	2877.12
8	727	-	4435300.0	5754.24
12	1036	-	4435300.0	8631.36
16	1210	-	4435300.0	11508.48
20	1630	-	4435300.0	14385.6
24	1742	-	4435300.0	17262.72

The results of running the simulation for CCTVs numbering from 4,8,12,16,20 and 24 in a Fog environment are given below. The execution time, cost of execution and the network usage increases with the increase in the number of CCTVs. The application loop delay stays constant at ≈ 10.

Table 4.2: Result Analysis of the Fog Environment

Number of CCTVs	Execution time	Application loop delay	Cost of execution	Network usage
4	1547	9.5375	109454.9662	1438.08
8	2487	9.5375	289824.00625	2876.16
12	3254	9.5375	409644.00625	4314.24
16	4089	9.5375	489524.00625	5752.32
20	5264	10.02	529143.7849	7192.8
24	6098	10.02	609023.7849	8631.36

From tables 4.1 and 4.2, we can compare the different metrics between the simulations in the two environments. For our thesis, we are focusing on the Cost of execution and the total network usage.

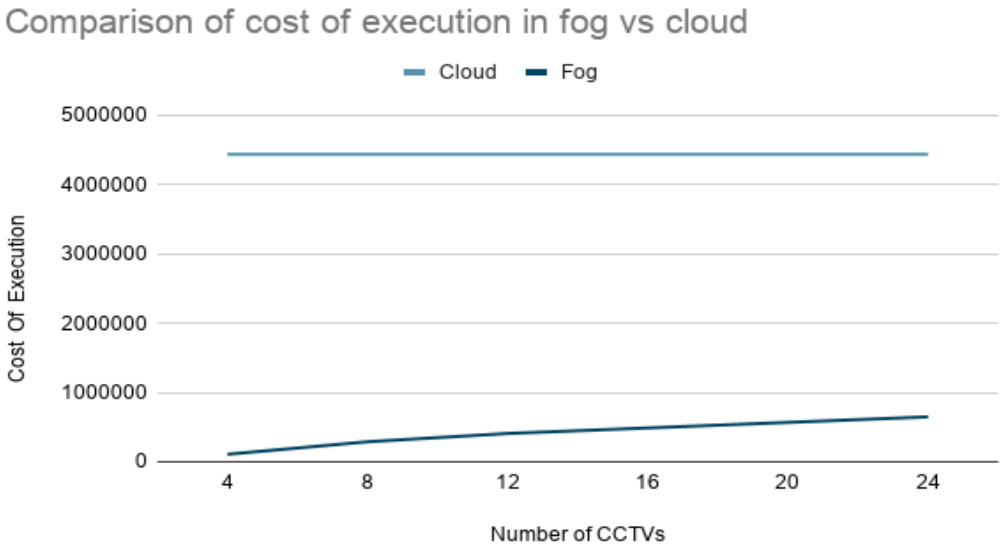


Figure 4.13: Comparison of cost of execution in Fog vs Cloud

The above graph Figure 4.13 shows the comparison between the cost of execution in the Fog topology and the Cloud topology. The cost in the Cloud topology is constant at 4435300, while the cost in the fog setup is considerably lower at the beginning, increasing with the introduction of each Fog server, even so, very slightly.

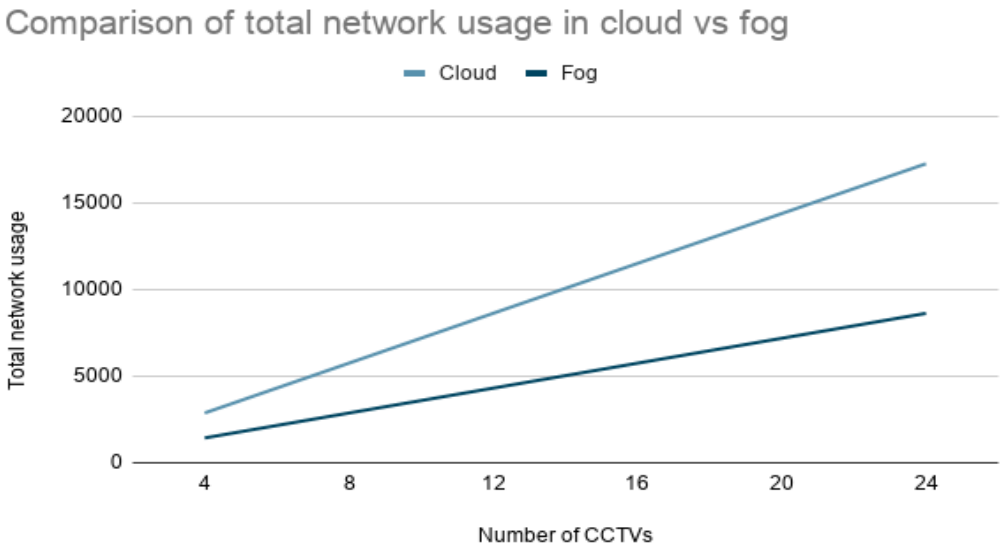


Figure 4.14: Comparison of total network usage in Cloud vs Fog

The above graph Figure 4.14 shows the comparison between the total network usage between the Cloud CCTV topology and the Fog CCTV topology. With an increasing number of CCTVs, both the graphs increase but the Cloud setup has a much steeper

increase than the Fog setup, meaning the total network usage per number of CCTVs in a Fog topology increases at a much lower rate than in a Cloud topology.

Chapter 5

Conclusion and Future Work

The implications of Cloud computing in our everyday life is rising by the day. But it is starting to face problems in terms of bandwidth and latency with the increase in the number of people and IoT devices. In our paper, we have shown that offloading the computation of CCTV footage for video processing and analysis to a Fog Server, rather than doing it in a local machine, can help in decreasing the total network usage and the cost of execution in a network. We have managed to simulate this with the help of the Java-based simulation kit iFogSim, where we modelled the logical topologies of Fog and Cloud and compared the results of the two environments. From our findings, we have managed to show that implementing a Fog topology can, in fact, help in decreasing the network usage and cost of execution in a network when compared to a Cloud topology. Offloading the computational work to a fog server also introduces the concept of allowing multiple users to reuse the same fog server to carry out different complex jobs.
There were multiple challenges that we faced during our research. One of the major challenges we faced was the lack of previous work done on this topic. Most of the papers written on the topic of fog computing were theoretical, talking about the architecture of fog computing, rather than about the application of it. Another issue which affected our work was the lack of resources, especially with the ongoing pandemic. Initially, we were hoping to allocate more than 4 CCTVs to a single Fog Server but were unable to, as our PCs simply could not run the simulation. Also due to the limitations of time and iFogSim, we were unable to showcase other performance metrics such as latency and throughput. We plan to improve on these shortcomings of our thesis and also hope our findings will help and benefit areas of research where our approach of offloading computational work to a fog server can be implemented.

Bibliography

[1] T. R. Sheltami, E. Q. Shahra, and E. M. Shakshuki, "Fog computing: Data streaming services for mobile end-users," *Procedia computer science*, vol. 134, pp. 289–296, 2018.

[2] T. H. Luan, L. Gao, Z. Li, Y. Xiang, G. Wei, and L. Sun, "Fog computing: Focusing on mobile users at the edge," *arXiv preprint arXiv:1502.01815*, 2015.

[3] C. Puliafito, E. Mingozzi, and G. Anastasi, "Fog computing for the internet of mobile things: issues and challenges," in *2017 IEEE International Conference on Smart Computing (SMARTCOMP)*, pp. 1–6, IEEE, 2017.

[4] T. Taleb, S. Dutta, A. Ksentini, M. Iqbal, and H. Flinck, "Mobile edge computing potential in making cities smarter," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 38–43, 2017.

[5] T. Taleb and A. Ksentini, "Follow me cloud: interworking federated clouds and distributed mobile networks," *IEEE Network*, vol. 27, no. 5, pp. 12–19, 2013.

[6] A. Ksentini, T. Taleb, and F. Messaoudi, "A lisp-based implementation of follow me cloud," *IEEE Access*, vol. 2, pp. 1340–1347, 2014.

[7] A. Ksentini, T. Taleb, and M. Chen, "A markov decision process-based service migration procedure for follow me cloud," in *2014 IEEE International Conference on Communications (ICC)*, pp. 1350–1354, IEEE, 2014.

[8] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, "Dynamic service migration in mobile edge-clouds," in *2015 IFIP Networking Conference (IFIP Networking)*, pp. 1–9, IEEE, 2015.

[9] W. Zhang, Y. Hu, Y. Zhang, and D. Raychaudhuri, "Segue: Quality of service aware edge cloud service migration," in *2016 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, pp. 344–351, IEEE, 2016.

[10] S. Yi, Z. Qin, and Q. Li, "Security and privacy issues of fog computing: A survey," in *International conference on wireless algorithms, systems, and applications*, pp. 685–695, Springer, 2015.

[11] H. Han, B. Sheng, C. C. Tan, Q. Li, and S. Lu, "A measurement based rogue ap detection scheme," in *IEEE INFOCOM 2009*, pp. 1593–1601, IEEE, 2009.

[12] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *2010 Proceedings IEEE INFOCOM*, pp. 1–9, Ieee, 2010.

[13] X. Wang, Z. Ning, and L. Wang, "Offloading in internet of vehicles: A fog-enabled real-time traffic management system," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4568–4578, 2018.

[14] J. Ni, K. Zhang, X. Lin, and X. Shen, "Securing fog computing for internet of things applications: Challenges and solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1, pp. 601–628, 2017.

[15] S. B. Kenitar, M. Arioua, A. Younes, M. Radi, and M. Salhaoui, "Comparative analysis of energy efficiency and latency of fog and cloud architectures," in *2019 International Conference on Sensing and Instrumentation in IoT Era (ISSI)*, pp. 1–5, IEEE, 2019.

[16] G. Caiza, M. Saeteros, W. Oñate, and M. V. Garcia, "Fog computing at industrial level, architecture, latency, energy, and security: A review," *Heliyon*, vol. 6, no. 4, p. e03706, 2020.

[17] S. Memon and M. Maheswaran, "Optimizing data transfers for bandwidth usage and end-to-end latency between fogs and cloud," in *2019 IEEE International Conference on Fog Computing (ICFC)*, pp. 107–114, IEEE, 2019.

[18] J. Li, T. Zhang, J. Jin, Y. Yang, D. Yuan, and L. Gao, "Latency estimation for fog-based internet of things," in *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*, pp. 1–6, IEEE, 2017.

[19] F. H. Rahman, T. W. Au, S. S. Newaz, and W. S. H. Suhaili, "A performance study of high-end fog and fog cluster in ifogsim," in *International Conference on Computational Intelligence in Information System*, pp. 87–96, Springer, 2018.

[20] L. Valentín, S. A. Serrano, R. O. Garcia, A. Andrade, M. A. Palacios-Alonso, and L. E. Sucar, "A cloud-based architecture for smart video surveillance," *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, p. 99, 2017.

[21] S. Shukla, M. F. Hassan, L. T. Jung, A. Awang, and M. K. Khan, "A 3-tier architecture for network latency reduction in healthcare internet-of-things using fog computing and machine learning," in *Proceedings of the 2019 8th International Conference on Software and Computer Applications*, pp. 522–528, 2019.

[22] R. Mahmud and R. Buyya, "Modelling and simulation of fog and edge computing environments using ifogsim toolkit," *Fog and edge computing: Principles and paradigms*, pp. 1–35, 2019.

[23] S. R. Hassan, I. Ahmad, S. Ahmad, A. Alfaify, and M. Shafiq, "Remote pain monitoring using fog computing for e-healthcare: An efficient architecture," *Sensors*, vol. 20, no. 22, p. 6574, 2020.