

MINIMIZING THE RESPONSE TIME OF VIDEO STREAMING USING CLOUDLET

A DISSERTATION

*Submitted in partial fulfillment of the
requirements for the award of the degree*

of

MASTER OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

AMIT KEWAL

[14535004]



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE – 247 667 (INDIA)

MAY, 2016

DECLARATION OF AUTHORSHIP

I declare that the work presented in this dissertation with title “**Minimizing the response time of video streaming using cloudlet** ” towards the fulfillment of the requirement for the award of the degree of Master of Technology in Computer Science & Engineering submitted in the Dept. of Computer Science & Engineering, Indian Institute of Technology, Roorkee, India is authentic record of my own work carried out during the period from July 2015 to May 2016 under the supervision of Dr. Sateesh K. Peddoju, Assistant Professor, Dept. of CSE, IIT Roorkee.

The content of this dissertation has not been submitted by me for the award of any other degree of this or any other institute.

DATE:

SIGNED:

PLACE:

(AMIT KEWAL)

CERTIFICATE

This is to certify that the statement made by the candidate is correct to the best of my knowledge and belief.

SIGNED:

DATE:

(Dr. Sateesh K. Peddoju)

Place: Roorkee

Assistant Professor

Department of CSE IIT Roorkee

ACKNOWLEDGEMENTS

FIRST and foremost, I would like to express my deep sense of gratitude and regards to my research supervisor Dr. Sateesh K. Peddoju, Professor of Computer Science & Engineering, Indian Institute of Technology, Roorkee, for his trust in my work, his invaluable guidance, regular source of encouragement and assistance throughout the course of dissertation work. His wisdom, knowledge and commitment to the highest standards inspired and motivated me. He has been very generous in providing the necessary resources to carry out my research. He is an inspiring teacher, a great advisor and most importantly a nice person.

I would like to thank **Railtel Corporation of India Ltd. (Grant Code: RCI-763(3)-ECD)** for providing financial assistance for procuring various hardware to setup cloud in High Performance Computing Lab.

On a personal note, I would like to say that I am indebted to my family for everything that they have given to me. I thank my parents, my brother and my friends for providing constant support, encouragement and care.

Lastly, I thank almighty for the wisdom and perseverance that he bestowed upon me during my research.

AMIT KEWAL

ABSTRACT

Since last decade Mobile Cloud Computing has gained a great importance not only for the small, medium but also for large sized enterprises. Mobile Cloud Computing is one of the most important technology on which many IT industries are currently working on. Today is the trend of mobile devices whether it is smartphones, tablets, laptops, smart watches, all have become the necessity of our life. Currently many researches are going on mobile cloud computing which focus on energy savings of mobile devices by offloading computing-intensive jobs from the mobile devices to the distant clouds, the access latency between mobile users and the clouds usually is large and sometimes unbearable. Cloudlet as a latest technology is capable to bridge this gap, and has been demonstrated to enhance the performance of mobile devices significantly while meeting the crisp response time requirements of mobile users. In this project the work is done on using the cloudlet as a small data center in order to reduce the network cost. Cloudlet technology is used to reduce the latency by installing it to the strategic locations over the network. It's an approach of bringing the clouds near to the mobile devices.

TABLE OF CONTENT

LIST OF FIGURES	iii
LIST OF TABLES	iv
1. OVERVIEW AND BASIC CONCEPTS.....	1
1.1. INTRODUCTION	1
1.2. MOTIVATION	2
1.3. PROBLEM STATEMENT.....	3
1.4. CONTRIBUTION OF THESIS.....	3
1.5. ORGANIZATION OF THESIS.....	3
2. LITERAURE REVIEW.....	4
2.1. INTRODUCTION	4
2.2. CLOUD COMPUTING	5
2.3. MOBILE CLOUD COMPUTING.....	6
2.3.1. Application of Mobile Cloud Computation	6
2.4. CLOUDLET	7
2.4.1. Cloudlet Attributes.....	7
2.4.2. Comparison between Cloud and Cloudlet	8
2.4.3. Cloudlet Concept	9
2.5. RESEARCH MODELS	10
2.6. RESEARCH GAPS	11
3. PROPOSED FRAMEWORK.....	12
3.1. OBJECTIVES	12
3.2. POSSIBLE SOLUTIONS.....	12
3.3. ENERGY EFFICIENCY USING CLOUDLET	13
3.4. PROPOSED ARCHITECTURE.....	14
3.4.1 Reducing Network Cost.....	14
3.4.2. Component of Latency.....	15
3.4.3. Flow Chart	16
3.4.4. Reducing Energy.....	17
3.5. REPRESENTATIONAL STATE TRANSFER (REST)	18

3.5.1. Comparisons between SOAP and REST	19
3.6. PACKET CAPTURE.....	20
3.7. MODIFIED LRU	20
4. IMPLEMENTATION	22
4.1. CLOUD SET UP	22
4.1.1. OpenNebula Design Principle:	22
4.1.2. OpenNebula functionalities:	23
4.1.3. OpenNebula Architecture	23
4.2. VIDEO STORAGE.....	23
4.3. WEB SITE DESIGNING	24
4.4. REQUEST PROCESSING	25
4.5. SEARCHING USING HASH TABLE.....	25
4.6. ANALYSIS	26
4.6.1. Data Collected of the top 10 most popular video sites	26
4.6.2. Zipf's Law.....	27
4.6.3. Zipf's Distribution	27
4.7. HASH TABLE DIVISION	28
4.8. PACKET CAPTURING	29
4.9. OUTPUT	30
4.9.1. Output at the client site	30
4.9.2. Output at the cloudlet site	31
4.10. STORED DATA	32
5. RESULTS	33
5.1. OUTPUT	33
5.1.1 Output showing size of webpage	33
5.2 RESPONSE TIME OUTPUT	34
5.2.1 Output obtained without cloudlet.....	34
5.2.2 Output obtained using cloudlet.....	35
5.3 COMPARISION	36
6. CONCLUSION AND FUTURE WORK.....	38
REFERENCES.....	39

LIST OF FIGURES

Figure 1-1: Video Traffic Analysis.....	2
Figure 2-1: Cloud Computing Architecture	5
Figure 2-2: Mobile Cloud Computing Architecture	6
Figure 2-3: Cloudlet Concept.....	7
Figure 2-4: Cloudlet Working Scenario.....	9
Figure 3-1: Hop count increases network cost.....	12
Figure 3-2: Cloudlet installed at specific location	13
Figure 3-3 Network without cloudlet and with cloudlet.....	13
Figure 3-4: Proposed Architecture.....	14
Figure 3-5: Time taken in Response	15
Figure 3-6: Flow Chart.....	16
Figure 3-7: Offloading Task to cloudlet	17
Figure 3-8: REST Architecture working.....	18
Figure 4-1: OpenNebula Features	22
Figure 4-2: OpenNebula Architecture	23
Figure 4-3: Website.....	24
Figure 4-4: Graph showing monthly visitors on websites	26
Figure 4-5: Graph showing Zipf's law on words	27
Figure 4-6: Divisions in hash table	28
Figure 4-7: Jars files included in the project.....	29
Figure 4-8: Response at client site	30
Figure 4-9: Packet capturing at cloudlet site.....	31
Figure 4-10: Captured Packets.....	32
Figure 5-1: Size of webpage	33
Figure 5-2: Response Time without using Cloudlet	34
Figure 5-3: Output obtained without using Cloudlet	34
Figure 5-4: Response Time using Cloudlet	35
Figure 5-5: Output obtained using Cloudlet	35
Figure 5-6: Comparison of Energy Consumption.....	36

LIST OF TABLES

Table 1:	Comparison between Cloud and Cloudlet.....	8
Table 2:	Comparison between SOAP and REST	17
Table 3:	Data of website’s monthly visitors.....	26
Table 4:	Results	27

1. OVERVIEW AND BASIC CONCEPTS

1.1. INTRODUCTION

As the number of mobile devices are increasing so their features are too advancing. Mobiles with HD display and high computation efficiency are emerging every day. So there is a shift of user from desktops to mobile platform [1]. In order to do so, several challenges are there including how to increase mobile device battery life and reduce network cost. In doing so Cloud computing plays an important role here. Combining Cloud computing and Mobile Computing can change the whole scenario of the computation which is done on the mobile devices. The total energy consumption is influenced by factors, such as computation tasks and communication of these devices and the transmission distances among these devices. Although many research studies are going on power aware mobile cloud computing then also here exist problem of response delay and offloading jobs to the distant cloud. For this problem, cloudlet is the technology that can fulfill the requirements. “A cloudlet is a mobility-enhanced small-scale cloud datacenter that is located at the edge of the Internet. The main purpose of the cloudlet is supporting resource-intensive and interactive mobile applications by providing powerful computing resources to mobile devices with lower latency”. In order to provide Quality of Experience (QoE) to users real time network monitoring is done. Recently a most effective model has been proposed i.e. HTTP adaptive streaming [2]. This HTTP adaptive streaming has been already deployed and being used by famous video streaming services like YouTube and Dailymotion. Seeing the current video demands this model can be quite effective in saving the battery life and reducing the networking cost. The thesis work focus on using cloudlet not only as data center but also as processing unit. This project is using the REST (Representation al State Transfer) architecture for creating the cloudlet technology.

1.2. MOTIVATION

Now a day, Mobile internet access has increased dramatically through Smartphone, tablet and laptop etc. There is a rapid increment in the mobile traffic with the increase of mobile services and networks. Now mobile devices (e.g., smartphones, tablets, laptops, smart watches) has become the necessity to our lives. The mobile device battery life and hardware technology has not improved sufficient enough to support computing intensive application. Therefore, many applications are still not suitable for mobile devices due to constrains, such limited battery life, low processing power, unpredictable network connectivity and limited memory [3]. By the year 2019, 90% of the mobile traffic will be accounted by cloud applications as compared to last year's 81%. Mobile data traffic will step up by 10-fold between 2014 and 2019. Mobile data traffic will reach 24.2 Exabyte per month by 2019 with a growth rate of a CAGR of 57 percent between 2014 and 2019[3]. According to Cisco Study by the year 2019 video traffic will dominates more than 80% of the network traffic. So there is a lot of scope of research in order to make mobile cloud computing more efficient and energy saving. The increment in network traffic can be seen in the following Fig. 1-1.

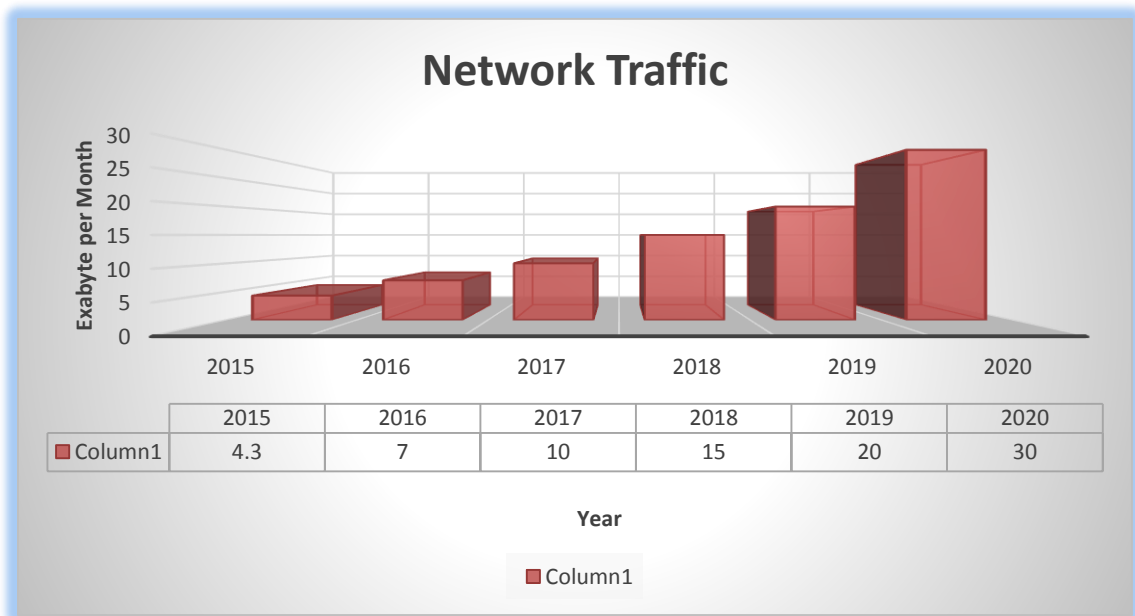


Figure 1-1: Video Traffic Analysis

1.3. PROBLEM STATEMENT

The thesis here tries to tackle the current problem we face in our mobile devices i.e. the quick battery decay while performing computing-intensive task and watching online videos. Problem statement of the thesis is described below:

“Minimizing the response time of video streaming using cloudlet”

1.4. CONTRIBUTION OF THESIS

The major contribution of this thesis are as follows:

- A working cloudlet for processing the client requests.
- An energy efficient model for mobile cloud computing.

1.5. ORGANIZATION OF THESIS

The thesis contains 7 chapters. First chapter is about brief introduction and rest of the thesis is described as follows:

Chapter 2 described of the literature reviews.

Chapter 3 discussed about proposed models.

Chapter 4 described the simulation part of the project.

Chapter 5 shows the results of the simulation and testing.

Chapter 6 discussed about conclusion and future work.

2. LITERAURE REVIEW

2.1. INTRODUCTION

In HTTP adaptive streaming model there is a web server that has all the videos encoded in different quality and each of them are divided into segment of fixed length(like 2 seconds,4 seconds or 6 seconds of playback)[3]. One of the important thing here is segment metadata that includes the information like playback length and segment quality. Here HTTP adaptive streaming model considers both the energy consumption of the mobile devices and the wireless network conditions. Here we are more concerned over Energy Efficiency in cloud computing. So here we will discuss how we can reduce the energy consumption of mobile devices by using cloud computing help. By the year 2019, cloud applications will generate about 90% of the total mobile device traffic. This project aims to provide energy efficient multimedia adaptive streaming for mobile users. Mobile user often suffer from long buffering delay and intermittent break down while streaming video via 3G/4G mobile network because of unusual behavior and limited capacity of the wireless network caused by user mobility. Dynamic Adaptive streaming over HTTP is used for adaptive bitrate streaming to provide user better quality of streaming that suits user current network condition independent of mobile devices [4]. Energy efficiency will be achieved by keeping the mobile network interface active only during the streaming session. And also during the transmission of segment for streaming, all the cloudlet in the path in between will cache the initial segment and some upcoming segment of each video efficiently so that next upcoming request can be served with the help of cache copy and simultaneously from cloudlet to actual server request can be made for upcoming segment. So, all the time streaming will be provided with the help of cache segment. Also the computing-intensive tasks which will drain battery faster should be offloaded to the cloudlet [5]. This way energy efficiency will be achieved.

2.2. CLOUD COMPUTING

According to NIST: “Cloud computing is a model for enabling available, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal management effort or service provider interaction”. Cloud Computing architecture can be viewed in the Fig. 2-1. The cloud computing has following features:

- **Access method:** User can edit, share, read files over the internet from anywhere in the world. On demand access to the resources is provided by the cloud.
- **Computing method:** Internet based computing is performed with the help of shared pools, configurable virtualized resources, including application, storage.
- **Setup:** Rapid provisioning/release, dynamic scalability, minimal management and service-provider interaction

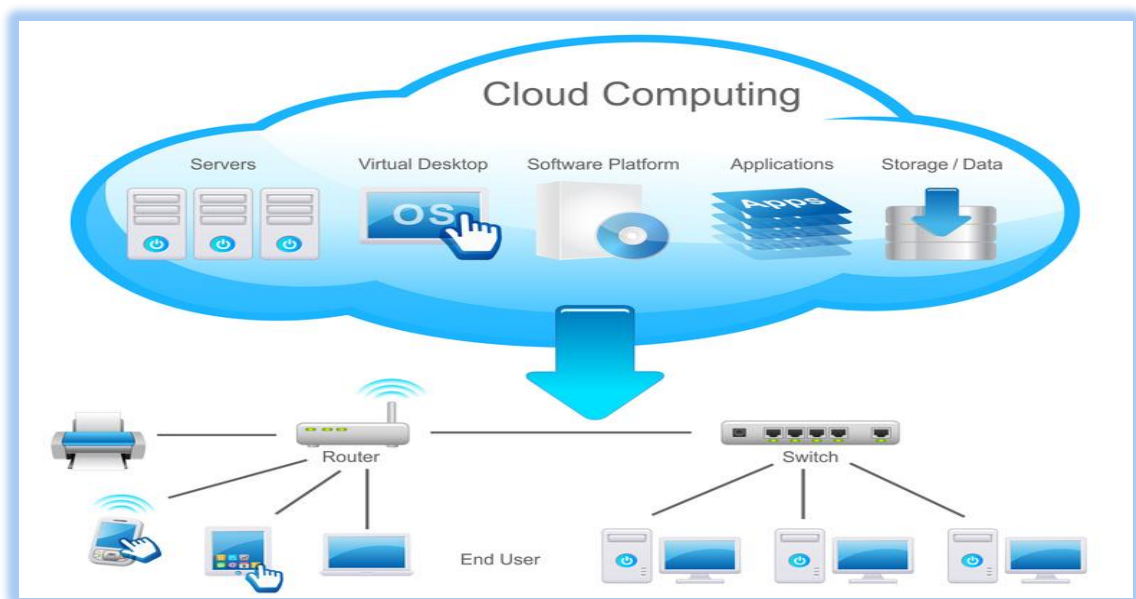


Figure 2-1: Cloud Computing Architecture [rubiconn.com]

2.3. MOBILE CLOUD COMPUTING

In a simple term mobile cloud computing refers to an infrastructure that combines the functionalities of both cloud computing and mobile devices. In this infrastructure the computing intensive processing and the data storage is done outside the mobile device. Mobile cloud application moves the computing-intensive application and the data storage away from the mobile devices into the cloud [5]. This helps in extending battery life, enhance CPU performance, and reduce power consumption of the mobile devices. Mobile Cloud Computing architecture can be easy understood from the Fig.2-2.

2.3.1. Application of Mobile Cloud Computing

- Offloading Computation
- Processing Speed and Data Storage
- Increase battery life
- Improving Reliability
- Security

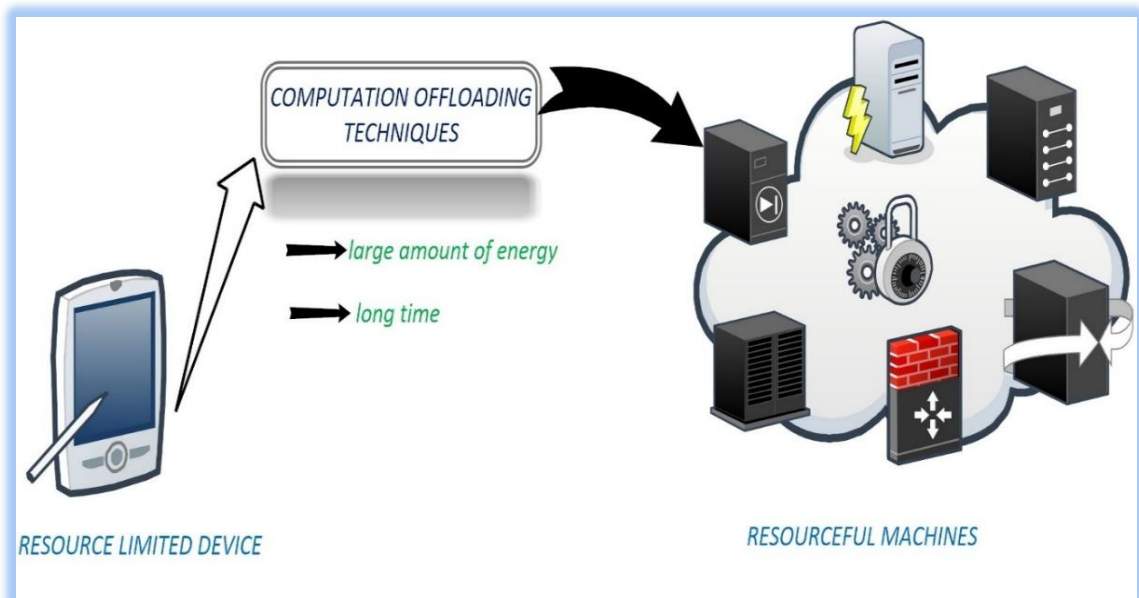


Figure 2-2: Mobile Cloud Computing Architecture

2.4. CLOUDLET

A cloudlet is a new architectural model in the current deployed architecture over the network. One can view the cloudlet as a small cloud present closer to the mobile devices which mimic the behavior of cloud. It is viewed as a “data center in a box”. Cloudlet is installed between the cloud and mobile device. A simple concept of cloudlet is shown in Fig. 2-3.

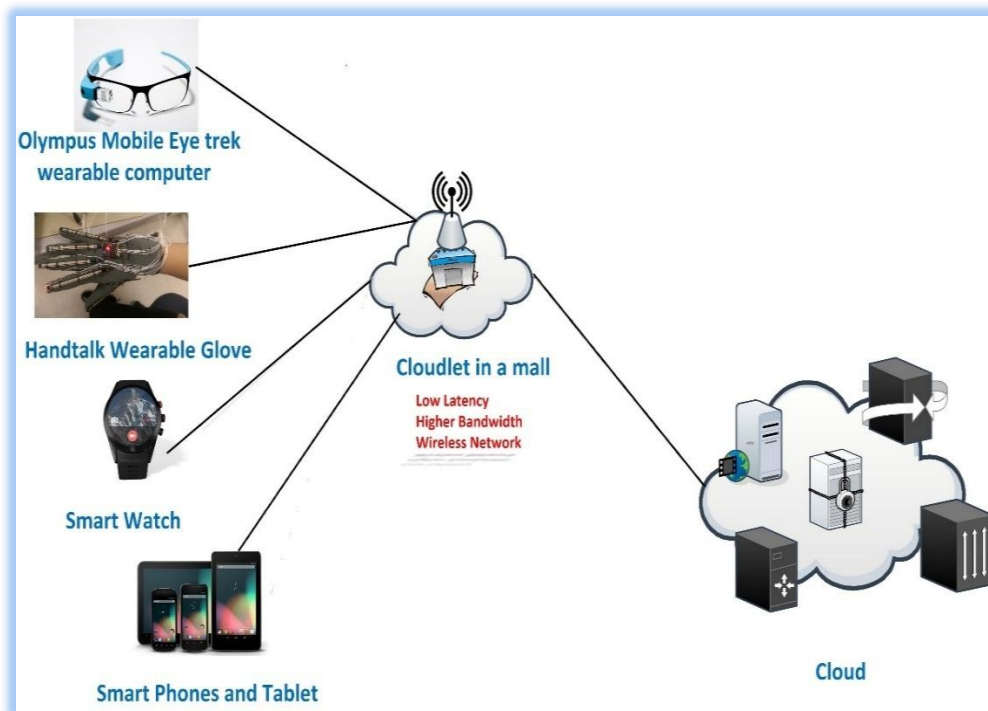


Figure 2-3: Cloudlet Concept

2.4.1. Cloudlet Attributes

- **Cloudlet consist of only soft state:** No hard state. Cloudlet can be used to cache/store the data which it receive from the cloud. Also can be used to store the data of the mobile devices. No need to monitor cloudlet like we need to monitor cloud.

- **Highly efficient system:** Cloudlet consist of high processing power systems (CPU, RAMS etc.). Here is no limitation of the power consumption. Cloudlet has an excellent connectivity with the cloud.
- **Lower Latency:** Since cloudlet is installed closer to the mobile user. The time needed to connect cloudlet is very low thus provide a much lower latency with a higher bandwidth.
- **Fulfill the demand of mobile devices:** Whenever a job is offloaded to the cloudlet, cloudlet do the processing in the virtual machines (VMs) and then returns the results to mobile devices. Generally computing-intensive tasks are offloaded to the cloudlet.

2.4.2. Comparison between Cloud and Cloudlet

Table 1: Comparison between Cloud and Cloudlet

	CLOUDLET	CLOUD
State	Only soft sate	Hard and Soft state
Management	Self-managed	Professionally managed
Environment	“Data Center in a box” at business premises	Machine room with power conditioning and cooling
Ownership	Decentralized ownership by local business	Centralized ownership by Amazon, Yahoo etc.
Network	LAN latency/bandwidth	Internet latency/bandwidth
Sharing	Few users at a time	100s-1000s of users at a time

2.4.3. Cloudlet Concept

In Fig. 2-4 we can see that mobile at node B is present in traffic and in order to communicate with other cell phones it can only use the 3G (GPRS) connection. Therefore its battery gets drain out very quickly. All of its offload activity has to pass through the 3G connection. Now the cell phone located at node C, for the communication purpose uses the Wi-Fi present in coffee shop and there cloudlet is also present. Mobile phone at node C when performs the calculation or computation tasks then it offload some of the computation task to the cloudlet which effectively helps in saving the battery life as we can see in the figure. Now the phone at last node D, communicates with others using Wi-Fi but can offload its computation to the cloudlet situated at far [4]. If this cell phone will offload its computation task to a distant cloudlet then it will take a longer time in computation thus energy saving will not be the optimal one. This shows that network connection also plays important role in saving energy of the mobile devices.

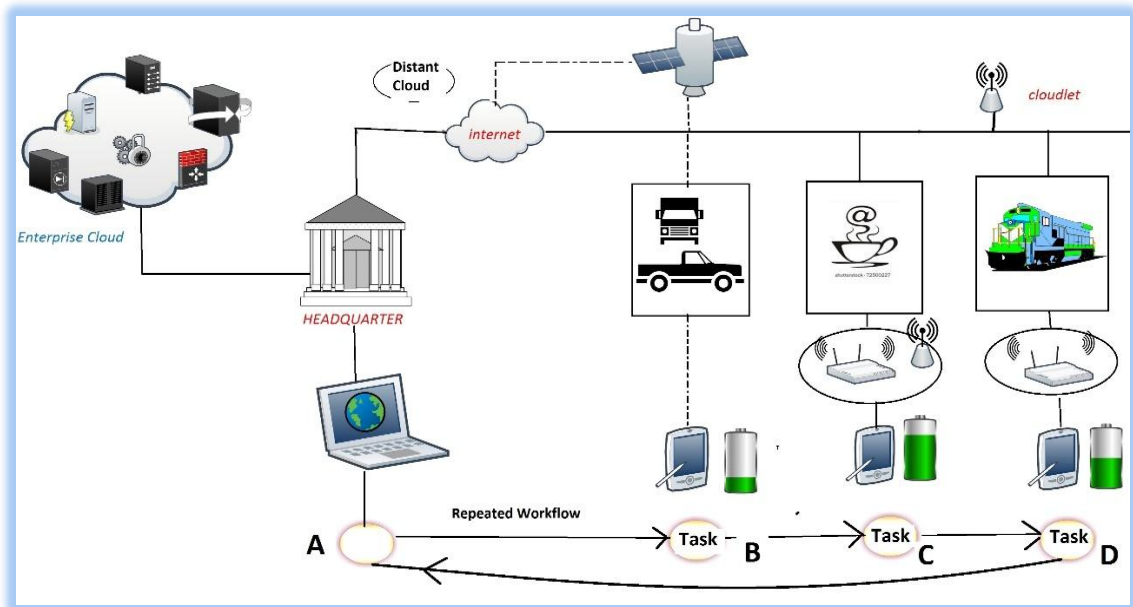


Figure 2-4: Cloudlet Working Scenario

2.5. RESEARCH MODELS

Various model has been proposed like by Gao et al. [5] proposed the energy-aware offloading model for workflows. This model described which tasks are likely to be offloaded to cloud and which are not, like task involving input/output access must be executed locally not on the cloudlet. Also discussed about selecting the best cloudlet available. Liu et al. [6] proposed a model in which a part of the total task which will consume a lot of energy is only offloaded not the whole task. This model doesn't show the diverse nature since it was only for a single mobile device. Wei et al. [4] proposes model of Mobile Cloud with Smart Offloading System where different offloading scenario are discussed. This model discussed how the mobile devices treated as the resource poor master node. So the service broker will map this master node to the slave cloud having high computation ability. Yunmin et al. [2] proposed the model which uses various available wireless networks for seamless adaptive high quality video service which not only reduces the networking cost but also saves the battery life of the mobile devices. Imam et al. [11] proposed the model in which energy is saved by doing the virtual machine migration among the servers. This model shows when all the load is transferred to minimum number of servers then the rest of the servers can be put in shutdown or sleep mode and in this manner energy can be saved. Hoang et al. [4] described in his survey paper about the mobile cloud computing and its application. Satyanarayanan et al. [12] presented a model for a cloudlet system that can accept seamless request from various mobile device. Li et al. [13] in his research described that by offloading a part of gaming code to the cloudlet can make user play the game for longer time. Kumar and Lu [14] demonstrated in his paper that by offloading the computing-intensive task to cloudlet can reduce the energy consumption of mobile devices. Kumar and Lu [14] also observed that it is not sure that every time offloading the computing-task to cloudlet will save energy when there is intensive communication. .

2.6. RESEARCH GAPS

Cisco Visual Network Index shows that mobile network over the world carried 18 Exabyte in 2013 and by 2018 the traffic will get doubled. Among the traffic generated by global mobile devices nearly 69% of traffic is due to video demand by the users [4]. So there is the need to look forward for energy efficiency technique which can be deployed in the current scenario of the mobile cloud computing comprising both network cost and mobile device battery. Previous models were not totally dedicated to the energy efficiency in mobile devices. So here in the proposed HTTP adaptive streaming model, a high-quality seamless video service can be achieved. Offloading the task to an optimal cloudlet among the various heterogeneous devices is the major concern [3]. Different cloudlet will have different resources and computability so selecting best of them is our task. However each device can process the computation task locally or may offload the data for computation on the cloudlet. Various factor that influences the energy consumption are data size, network bandwidth, computation power, transmission distance. Bringing cloud nearer to the users is quite tough job. We need to bring cloud closer to user because when we transfer heavy computational task to the cloud then the network cost increases. So this is the major concern for researcher. Various studies are going on for this field. One solution to this we have obtained is the introduction of cloudlet in our network architecture. A lot of scope is there to research in this area.

3. PROPOSED FRAMEWORK

3.1. OBJECTIVES

- Reducing Network Cost by placing Cloudlet nearer to the mobile device.
- Bringing cloud closer to the end user.

3.2. POSSIBLE SOLUTIONS

The network cost mainly depends upon the packet delay. The delay in packets occurs when there are large number of hop counts from client to server and server to client. So when the hop counts increases the latency time also increases.

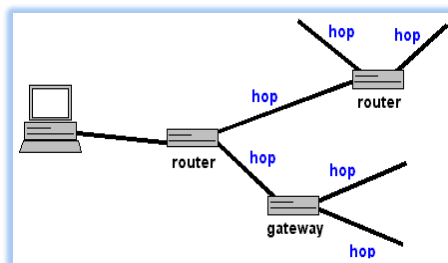


Figure 3-1: Hop count increases network cost

One of the possible solution is to have the control over the packets. This can be achieved when we will be able to bring the cloud closer to the clients. If we go for the datacenter installation over the specific locations then there installation will be expensive because installing 20-30 servers will need some investment. This can only be achieved by installing the cloudlet over the network as a specific locations. Cloudlet will also reduce the task of router.

Videos form various streaming website like YouTube, Netflix, hot star etc. can be cached in the cloudlet. "Customers are certainly concerned about latency, and often they don't know what to expect," said Alex Watson-Jackson, IT Infrastructure and Services Solutions marketing manager at pan-European network operator Colt Technology Services.

Studies has shown that by splitting tcp connection the latency can be reduced by ~30 msec. Also it is observed that the response time of search query gets reduced by ~25-35%.

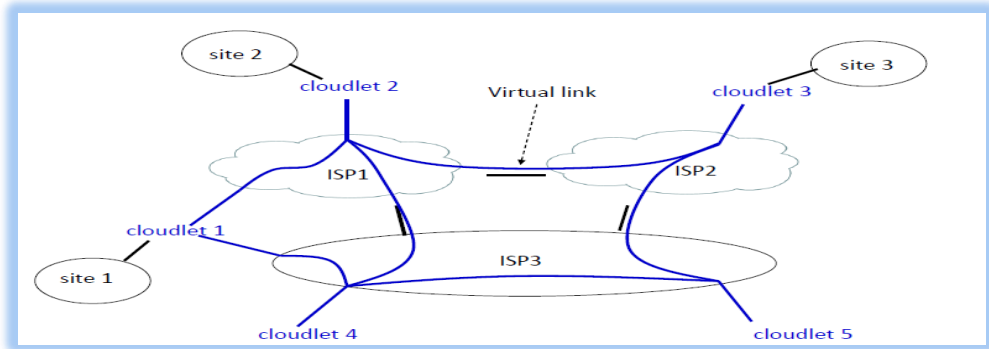


Figure 3-2: Cloudlet installed at specific location

3.3. ENERGY EFFICIENCY USING CLOUDLET

In the research it has been shown that with the increase in the network latencies the battery consumption also gets increased. So when a mobile device is connected to a LTE network it consumes more than 1.5 W. This can be seen in the following Fig 3-3 that the chip of the device remains on active state for some amount of time but not in the case when cloudlet is introduced. With the introduction of cloudlet power consumption value remain same but for lesser amount of time. The tail time also get reduced which also helps in saving energy. This type of result can be seen in the result chapter.

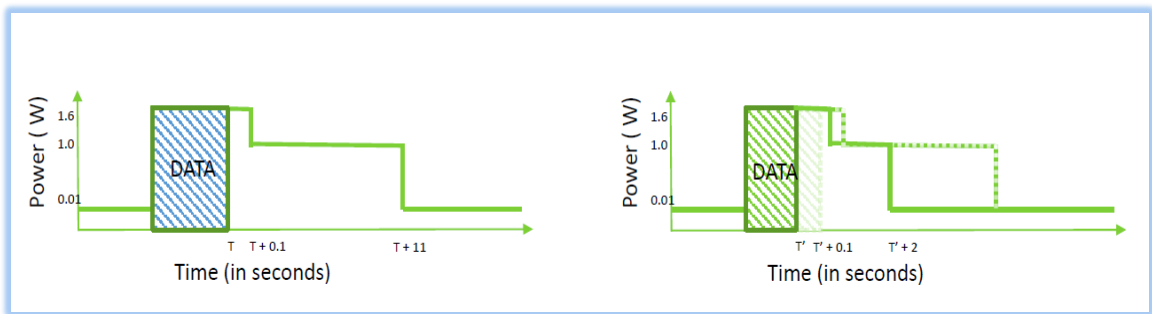


Figure 3-3 Network without cloudlet and with cloudlet [Microsoft]

Since, $Power = \frac{Energy}{Time}$ so in this case the power value remain same and time gets reduced.

This results in saving of energy.

3.4. PROPOSED ARCHITECTURE

3.4.1 Reducing Network Cost

All the packets coming to the system Ethernet are being captured. Then filtering it according to the protocols and then saving the packets in the pcap format. Then extracting the data and storing it in the cloudlet in a particular format so that if again the same request is arrived then directly through cloudlet the response will be sent back to the mobile devices. The idea is to reduce the response *time* (latency) and *network* cost. In the following, Fig 3-2 the working of the proposed architecture is show.

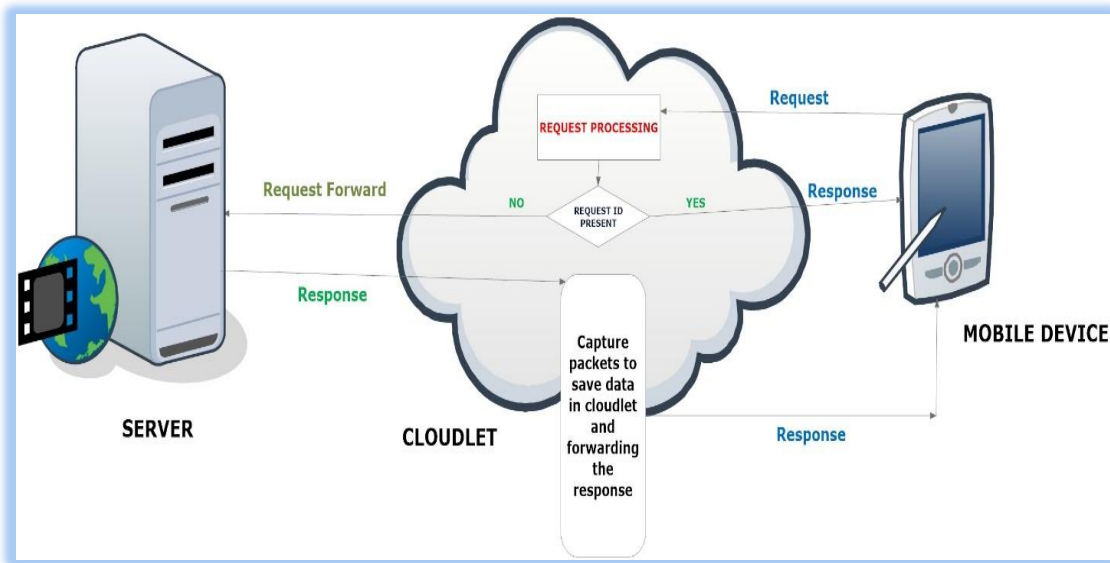


Figure 3-4: Proposed Architecture

In order to calculate the response time of the web application the formula used is described below.

$$R(\text{Response Time}) = \left(\frac{P}{D}\right) + RTT + \left(\frac{N}{C}\right) + \text{Client CT} + \text{Server CT} \quad 3.1$$

Here description of every parameters are described below:

P : It refers to payload means size of the webpage and its resource file in bytes.

D : It refers to the bandwidth of the network.

RTT : Round Trip Time is the amount of time from client to server and back to client again in milliseconds.

N : It refers to number of resources (images, CSS, script, etc.) to provide service.

C : It refers to number of requests which will receive the resources.

Client CT : It refers to the client response time.

Server CT : It refers to the server response time.

In the project **RTT** value is also reduced with the introduction of the cloudlet. Only the first time the **RTT** value will be actual one after that it will reduce drastically because next time always the reply will always be send by cloudlet.

3.4.2. Component of Latency

- Client
- Network
- Datacenter

The below Fig 3-5 shows the overall time taken for getting a response from the server to the client.

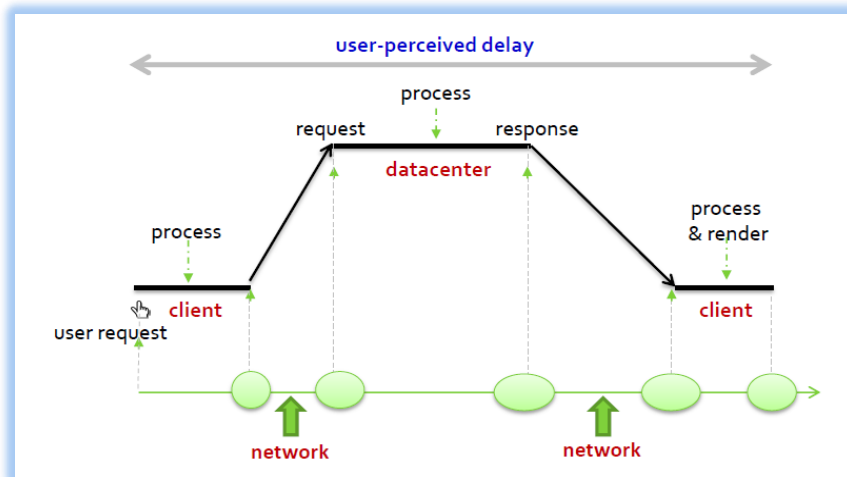


Figure 3-5: Time taken in Response

3.4.3. Flow Chart

The below flow chart briefly described the working of the proposed model. The URL “http://localhost:8080/MCC/rest/hello/id?id=https://www.youtube.com/watch?v=EJylz_9KYf8” format is shown below and after processing the request ID becomes “https://www.youtube.com/watch?v=EJylz_9KYf8”. And further working is shown in following flow chart.

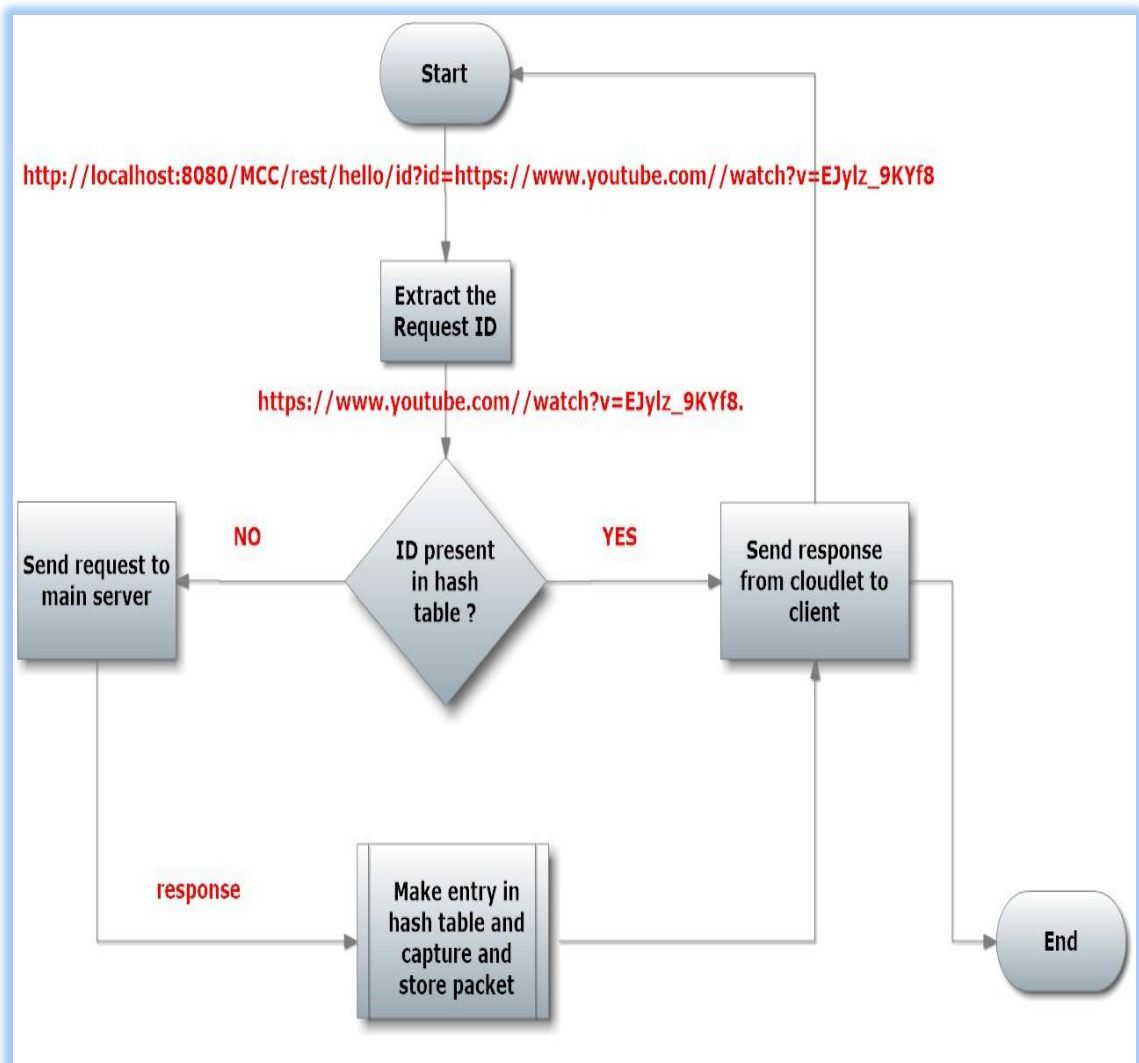


Figure 3-6: Flow Chart

3.4.4. Reducing Energy

In the job offloading technique, fetching of the job queue is performed from the mobile device. Then recognize the computing-intensive tasks. After that these tasks are offloaded to the nearby cloudlet. In cloudlet the processing occurs and the results are sent back to the mobile devices. The idea is to save the energy and thus the power consumption too. The mobile device has the architecture as shown below. It has a processor to execute all the tasks while memory is there to store the data. New jobs are inserted to the queue. Now the algorithm is to be deployed on the client device to monitor the jobs whether it is beneficial to offload the job or not. It is not always guaranteed that the energy will be saved by offloading task because other factor like network cost may affect the result. The jobs like transactions and input/output tasks cannot be offloaded to the cloudlet because of the security issues. Cloudlet can be installed anywhere in an organizations like in offices, colleges, libraries, metro stations, trains, coffee shops etc. which will help a lot in saving energy. Fig 3-7 shows the mobile device architecture and the cloudlet concept.

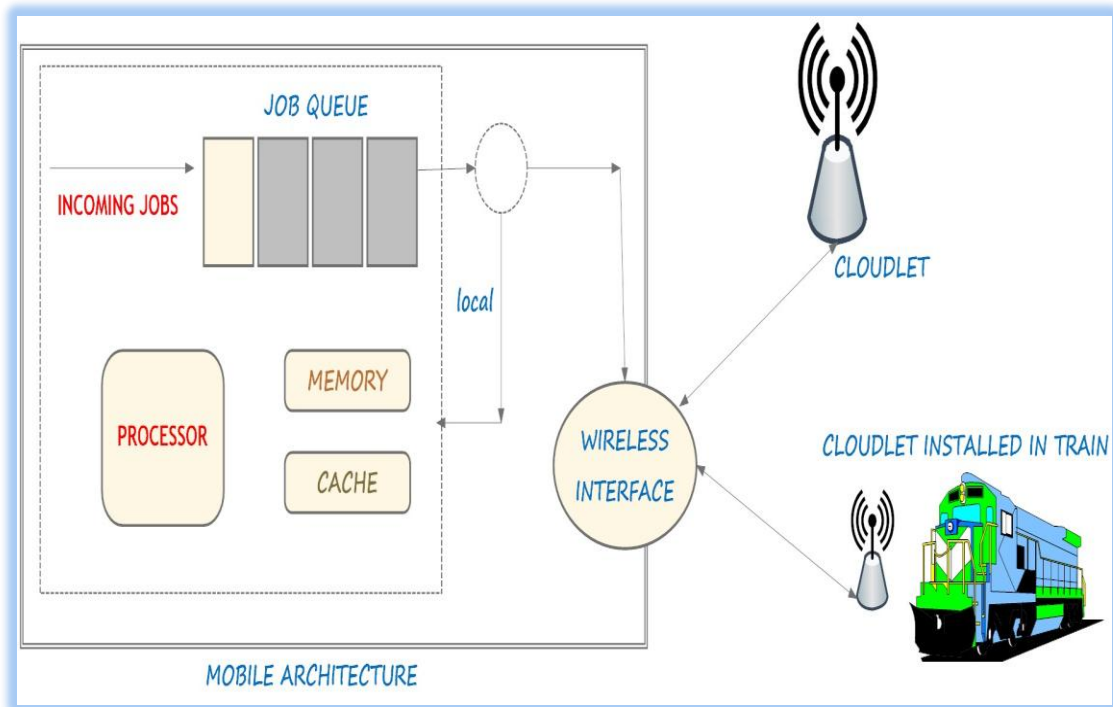


Figure 3-7: Offloading Task to cloudlet

3.5. REPRESENTATIONAL STATE TRANSFER (REST)

REST architecture is used for the implementation purpose. REST is a term coined by Roy Fielding to describe an architecture style of networked systems. REST is an acronym standing for Representational State Transfer. According to Roy Fielding "Representational State Transfer is intended to evoke an image of how a well-designed Web application behaves: a network of web pages (a virtual state-machine), where the user progresses through an application by selecting links (state transitions), resulting in the next page (representing the next state of the application) being transferred to the user and rendered for their use." Fig 3-5 shows the REST service working.

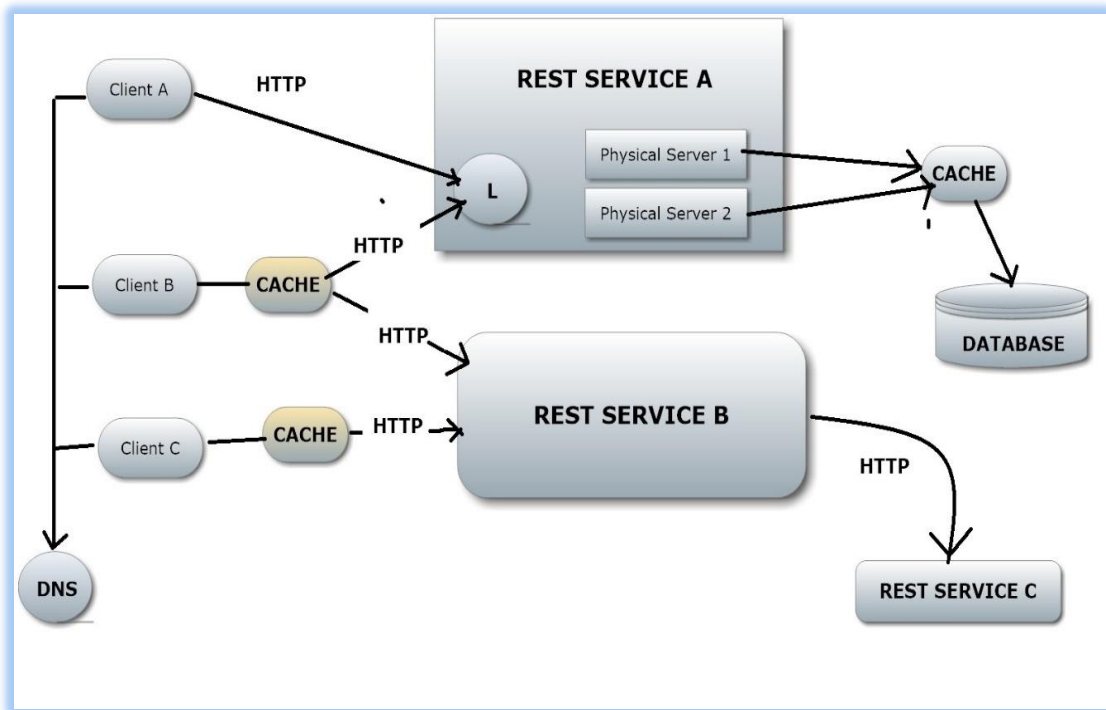


Figure 3-8: REST Architecture working

An application or architecture considered RESTful or REST-style is characterized by:

- State and functionality are divided into distributed resources
- Every resource is uniquely addressable using a uniform and minimal set of commands (typically using HTTP commands of GET, POST, PUT, or DELETE over the Internet)
- The protocol is client/server, stateless, layered, and supports caching

3.5.1. Comparisons between SOAP and REST

Table 3.1: Comparison between SOAP and REST architecture

#	SOAP	REST
1	A XML-based message protocol	An architectural style protocol
2	Uses WSDL for communication between consumer and provider	Uses XML or JSON to send and receive data
3	Invokes services by calling RPC method	Simply calls services via URL path
4	Does not return human readable result	Result is readable which is just plain XML or JSON
5	Transfer is over HTTP. Also uses other protocols such as SMTP, FTP, etc.	Transfer is over HTTP only
6	JavaScript can call SOAP, but it is difficult to implement	Easy to call from JavaScript
7	Performance is not great compared to REST	Performance is much better compared to SOAP - less CPU intensive, leaner code etc.

We use SOAP architecture for following application:

- Internet Application for more maturing tooling.
- B2B applications for strong contract.
- Exposing existing applications.

We use REST architecture for following application:

- Cloud-based and mobile applications.
- Creating high scalable stateless applications.

3.6. PACKET CAPTURE

This project is implemented in java language. Since there is no inbuilt library present in java for capturing IP packets therefore an external open source libraries like jpcap, jnetpcap, jNetStream libraries is used for capturing, storing and reading the packets. Using these libraries we can perform following jobs:

- Capture real time packets directly from the Ethernet.
- Saves all the packets to an offline file. E.g. *.jpcap file.
- Using jpcap we automatically identify packet types and generate corresponding Java objects (for Ethernet, IPv4, IPv6, ARP/RARP, TCP, UDP, and ICMPv4 packets).
- Also provides the filter feature for the incoming packets.
- Also allows to send the raw packets to the network.

3.7. Modified LRU Caching for Faster Access

In this section, we assume an infinite request stream and finite cache with a capacity of C web pages. The cache can hold C web pages regardless of the size of each web page. Furthermore, we assume that the cache holds the C most popular pages as indicated by the ordering i . This ordering is determined by measuring the request frequency for each page, which is equivalent to assuming that the cache has a Perfect-LRU page removal policy.

The asymptotic hit-ratio $H(C)$ is given by:

$$H(C) = \sum_{i=1}^C P_N(i) \quad (3.1)$$

Probability of hitting the cache on cloudlet: $P_N(i) = \frac{1}{c}$. (3.2)

For storing the response from the video streaming server in our cloud 'AMBAR' we use time ratio parameter α .

$$\alpha = \frac{\textit{Time to load using cloudlet}}{\textit{Time to load without cloudlet}}, \text{ where } 0 < \alpha < 1. \quad (3.3)$$

Let β be the threshold time ratio and β is calculated after the analysis of the different α values obtained from the project. For a size S web page if the α value for a particular website is more than β then only its response will be saved in the hash table otherwise its will not be saved. Also if $\alpha > \beta$ and the hash table is full then only we will use least replacement used algorithm (LRU) for removing an entry from hash table and the video from the cloudlet.

4. IMPLEMENTATION

4.1. CLOUD SET UP

The OpenNebula technology is used for creating the cloud named “AMBAR” in the high performance computing (HPC) lab. In this cloud we have created various Virtual machine for different purposes. Streaming VMs are used for storing the videos in mpd (media presentation description) format. OpenNebula is a cloud computing platform that enables us to manage the large virtualized data centers in an efficient and simple way. OpenNebula helps in establishing the private, public, and hybrid cloud infrastructure.

4.1.1. OpenNebula Design Principle:

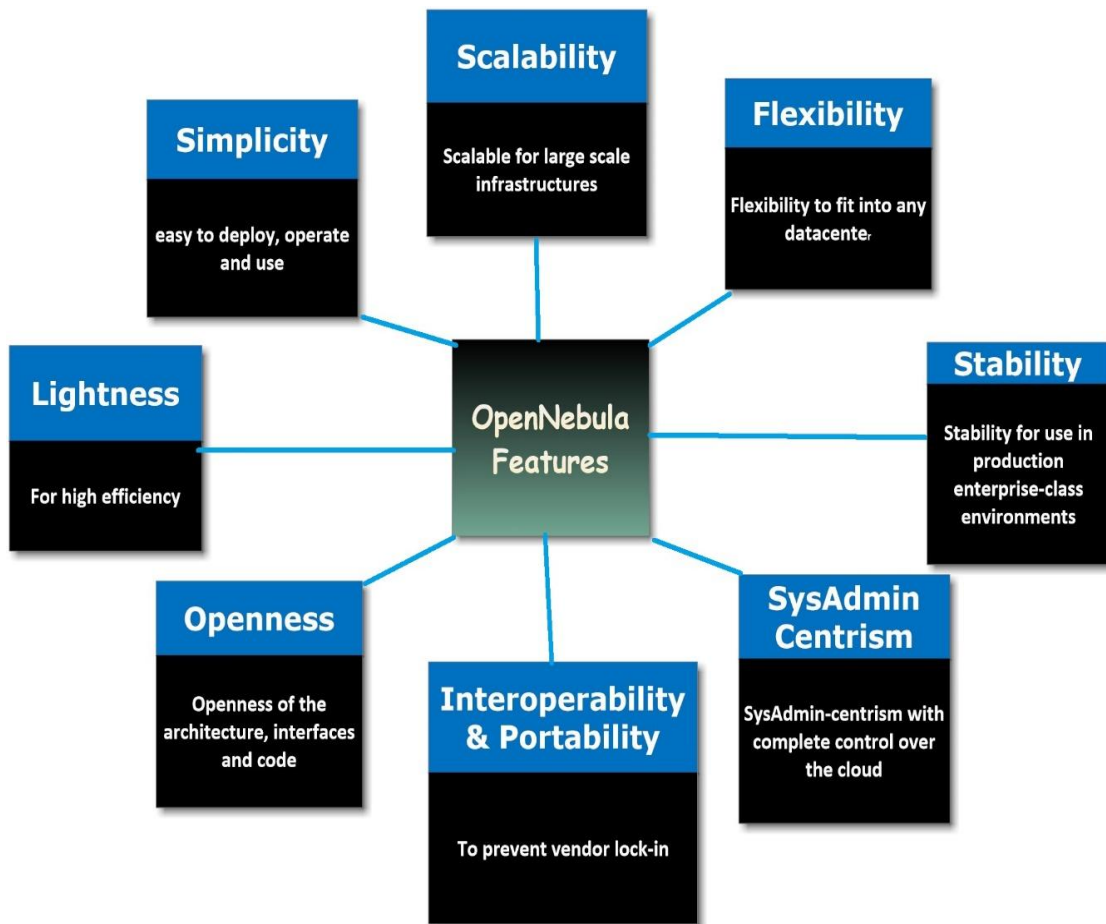


Figure 4-1: OpenNebula Features

4.1.2. OpenNebula functionalities:

- Management of data center virtualization
- Management of Cloud

4.1.3. OpenNebula Architecture

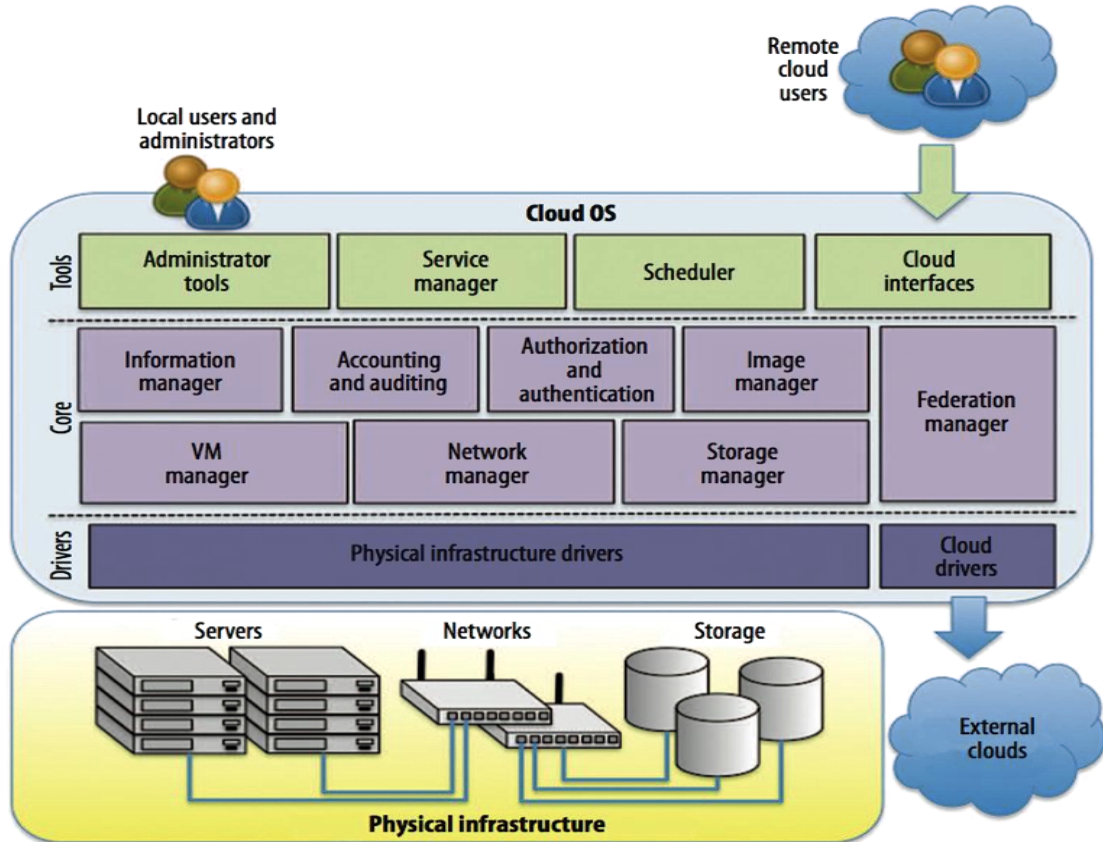


Figure 4-2: OpenNebula Architecture [opennebula.org]

4.2. VIDEO STORAGE

The cloud named *AMBAR* is being used here to store the videos in the mpd format. MPD stands for media presentation format. Each video in our server is present in different quality pixels. We stored video in this way because the service we provide is adaptive bit rate video streaming means according to the network bandwidth available in the network we will provide corresponding quality of video. If bandwidth is low then lower quality videos and if bandwidth is high then higher quality videos will be delivered to the client. These video

can only be played by dash player that comes embedded in the chrome browser. Dash player enables HTML5 adaptive streaming with MPEG-DASH native in your browser with no need for plugins like Flash or Silverlight. Due to the native integration with the browser it is possible to play back very high resolutions such as 4K or very high frame rates like 60 fps.

4.3. WEB SITE DESIGNING

A website is being created as part of the project to provide the simple interface to the clients, where user can make their account and can get access to the videos. The whole project is designed to provide a high quality seamless video streaming. The website is also mobile compatible. Snapshot of the website are as follows.



Figure 4-3: Website

4.4. REQUEST PROCESSING

Now coming to the algorithm, whenever the user wants to see the video he/she clicks on the video. Then the request is sent to the cloud server. But this request is being stopped by installing a cloudlet in between of the cloud and client. It's a communication among cloud-cloudlet-client. By REST architecture client and server are established and the client behave as the mobile device whereas server behaves as a cloudlet. So when a request is fired from the client mobile device then the request is captured in between. Then request ID is being extracted from the URL request and then search that ID in the hash table. If this ID exist in the hash table then response is send directly from cloudlet to client mobile device otherwise request is send to the cloud. When the response comes from the cloud it passes through cloudlet. In cloudlet data coming from the cloud are captured and stored and simultaneously response is sent back to client.

4.5. SEARCHING USING HASH TABLE

Hash Table: Dynamic-set data structure for storing items indexed using *keys*. Hash Table supports Insert, Search, and Delete Operations. Methods in hash table are as follows:

- `get(k)`: If in hash table, the map M has a key k then it returns its associated value else it returns null.
- `put(k, v)`: The method call returns the previous value of the specified key in this hash table, or null if it did not have one.
- `remove(k)`: if the map M has an entry with key k , remove it from M and return its associated value; else, return null
- `size()`, `isEmpty()`
- `keys()`: return an iterator of the keys in M
- `values()`: return an iterator of the values in M

4.6 ANALYSIS

4.6.1. Data Collected of the top 10 most popular video sites

One of the most popular internet activities "Videos" have grown to be one of the most modern cargos on the Web. People watch videos like news, tutorials, TV shows, and celebrity videos etc. Here are the top 10 most popular video streaming sites based on its traffic rankings.

Table 4: Data of website's monthly visitors

WEBSITE	RANK (Alexa rank)	MONTHLY VISITORS
YouTube	3	450,000,000
Netflix	100	55,800,000
Hulu	171	40,000,000
Dailymotion	101	27,000,000
Metacafe	191	26,000,000
Myspace.	205	24,200,000
Screen.Yahoo.	221	17,000,000
Vimeo	258	16,950,000
Break.com	447	15,000,000
Tv.com	718	13,900,000

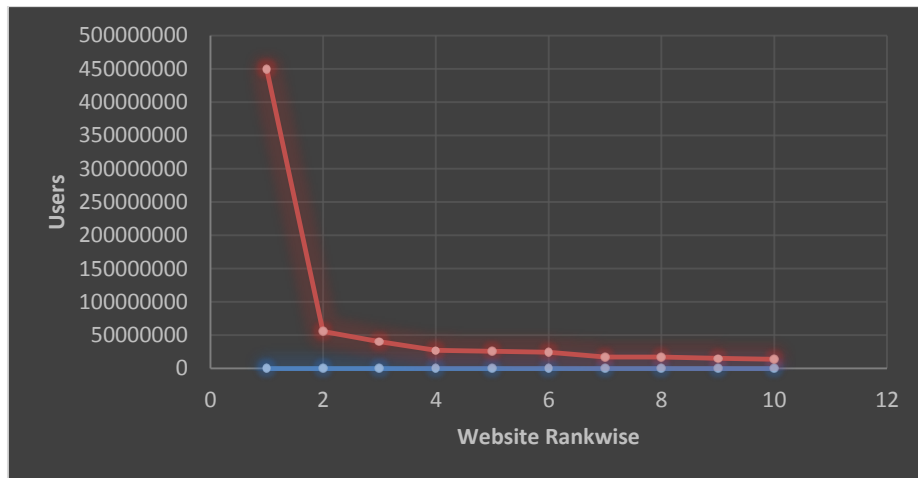


Figure 4-4: Graph showing monthly visitors on websites

4.6.2. Zipf's Law

Zipf's law states that “given some corpus of natural language utterances, the frequency of any word is inversely proportional to its rank in the frequency table. Thus the most frequent word will occur approximately twice as often as the second most frequent word, three times as often as the third most frequent word, etc.”

4.6.3. Zipf's Distribution

- ❖ A few elements occur *very frequently*.
- ❖ A medium number of elements have medium frequency.
- ❖ Many elements occur *very infrequently*.
- ❖ The product of the frequency of words and their rank is approximately constant.

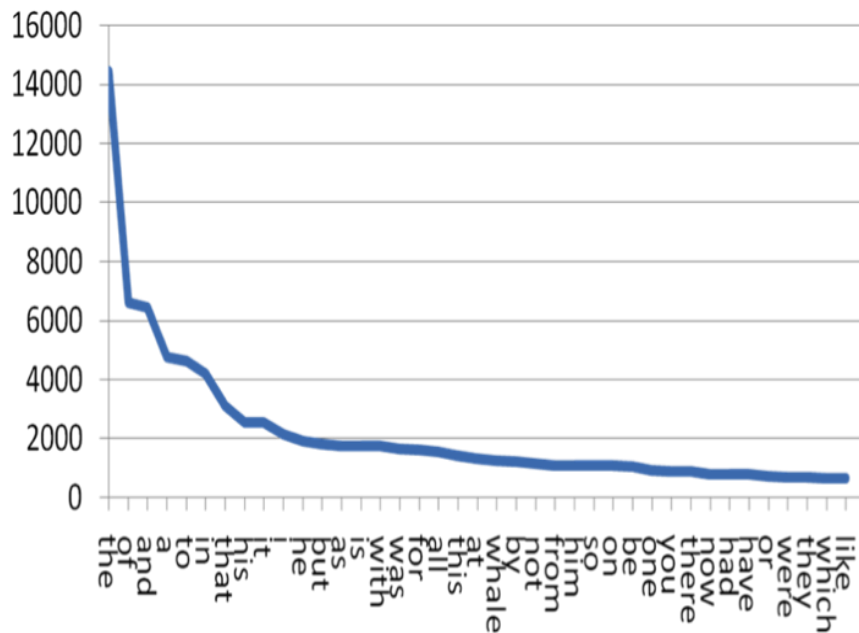


Figure 4-5: Graph showing Zipf's law on words [github.com]

4.7. HASH TABLE DIVISION

The graph which is obtained of the website according to the user view count resembles the Zipf's law. According to this law, popular video streaming websites like YouTube will always be popular with comparison to the less popular video streaming websites and the view count of popular website will increase at a greater pace with comparison to the less popular websites. So in order to save the request ID in hash table, firstly its division should be defined according to the websites popularity like YouTube should be given around half the space of the hash table then $1/3^{\text{rd}}$ space should be provided to the next popular website. Similarly the corresponding videos will be stored in the cloudlet. This can be easily understood by the following fig. 4-6.

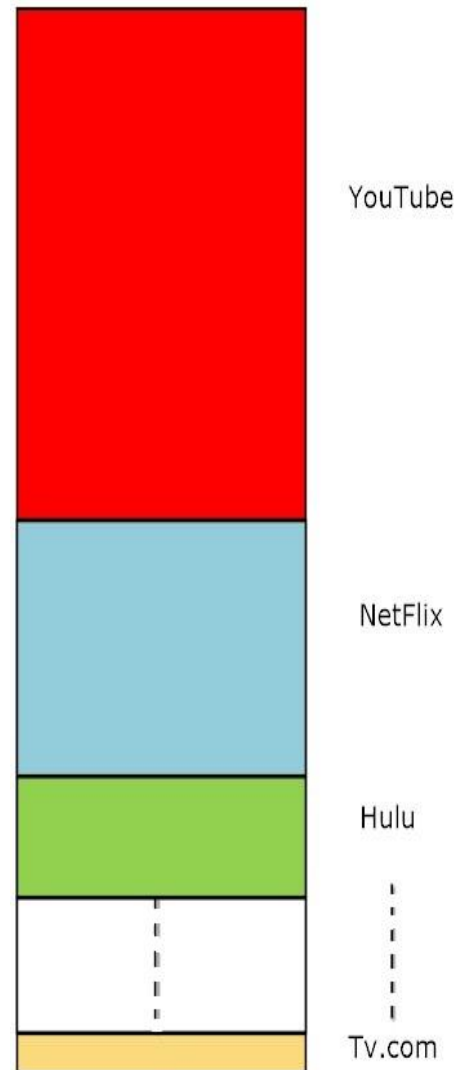


Figure 4-6: Divisions in hash table

4.8. PACKET CAPTURING

The whole project is implemented in java language. Since packet capturing facility is not provided by java so the external libraries for the required purpose. Jpcap, Jnetpcap is the external library. These libraries are open source so anyone can get it on www.sourceforge.net website. A 32 bit compatible jpcap library. Fig 4-4 shows the list of jar files added in order to do packet capturing. Adding jpcap requires following steps:

- First download jpcap jar file and jpcap.dll file from www.sourceforge.net.
- Install *winpcap* if you are window user.
- Install eclipse IDE and jdk 32 bit version because jpcap is 32 bit version.

Manually copy the jpcap jars and jpcap.dll file in the jdk and jre folder respectively. (e.g. “C:\Program Files (x86)\Java\jdk1.8.0_77\bin” and “C:\Program Files (x86)\Java\jdk1.8.0_77\jre\lib\ext”).

- Now we can use the jpcap classes for our required project.

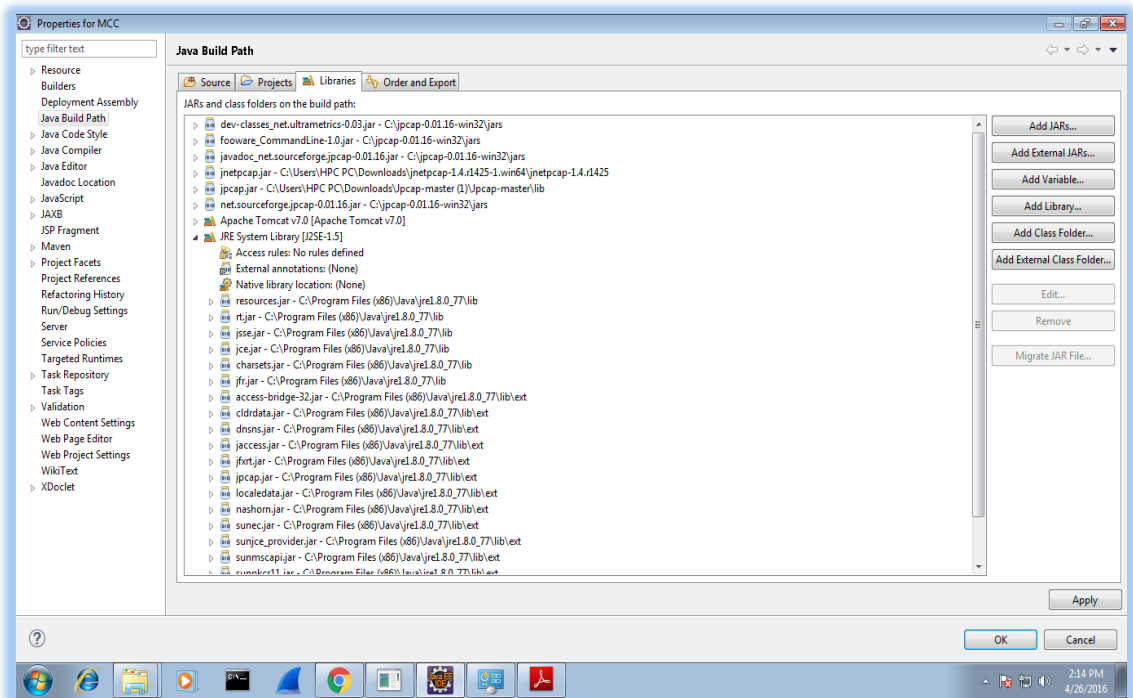


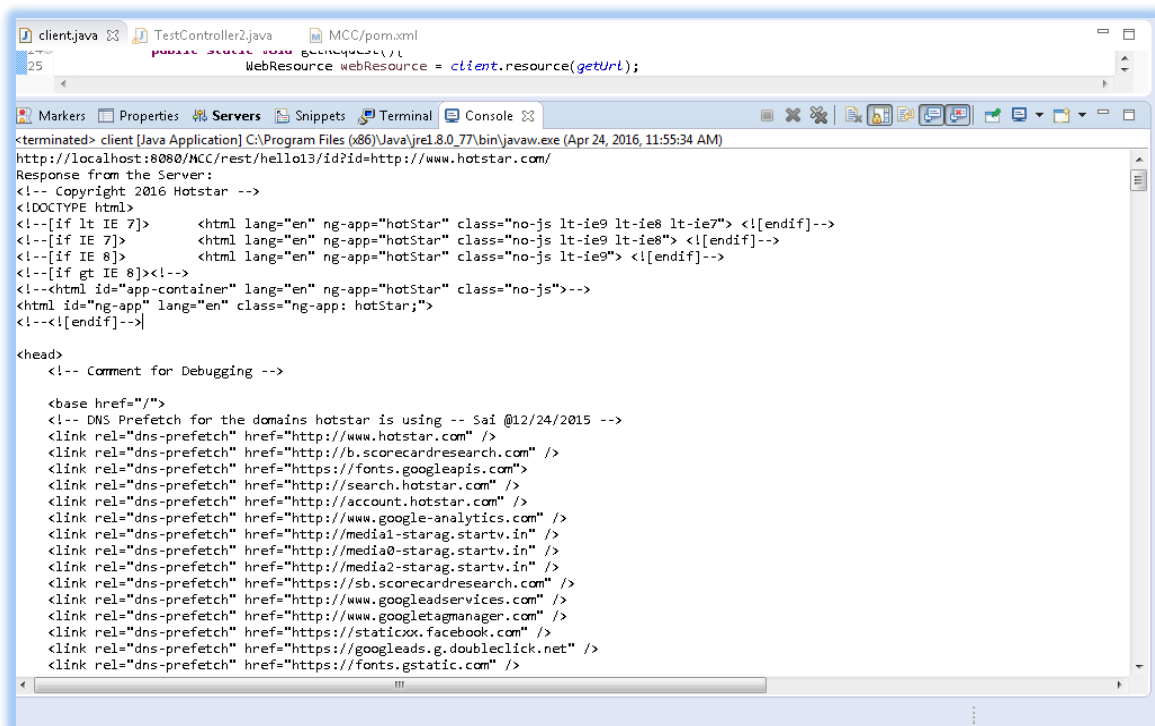
Figure 4-7: Jars files included in the project

4.9. OUTPUT

The written code tries to capture all the packets coming to our cloudlet. For this we gain access to the Ethernet port. All the packets coming to Ethernet are all captured. The file which we obtain by capturing packets is of pcap format. So when the request ID is not present in the hash table the request is forwarded to the cloud. The response which arrive to cloudlet are all captured. The data is extracted from the tcp packet and the raw data which we obtained from packet is stored in the cloudlet. Simultaneously response is send back to the client.

4.9.1. Output at the client site :

Application level output which is received at the client site is displayed in the Fig. 4-5 below. This response is provided by the cloudlet after checking in its hash table. If hash table has the response corresponding to the request ID then the response is sent by cloudlet otherwise from the main server via cloudlet.



```
clientjava TestController2.java MCC/pom.xml
25 WebResource webResource = client.resource(getUrl);

<terminated> client [Java Application] C:\Program Files (x86)\Java\jre1.8.0_77\bin\javaw.exe (Apr 24, 2016, 11:55:34 AM)
http://localhost:8080/MCC/rest/hello13?id?id=http://www.hotstar.com/
Response from the Server:
<!-- Copyright 2016 Hotstar -->
<!DOCTYPE html>
<!--[if lt IE 7]> <html lang="en" ng-app="hotStar" class="no-js lt-ie9 lt-ie8 lt-ie7"> <![endif-->
<!--[if IE 7]> <html lang="en" ng-app="hotStar" class="no-js lt-ie9 lt-ie8"> <![endif-->
<!--[if IE 8]> <html lang="en" ng-app="hotStar" class="no-js lt-ie9"> <![endif-->
<!--[if gt IE 8]><!-->
<!--<html id="app-container" lang="en" ng-app="hotStar" class="no-js">-->
<html id="ng-app" lang="en" class="ng-app: hotStar;">
<!--<![endif-->

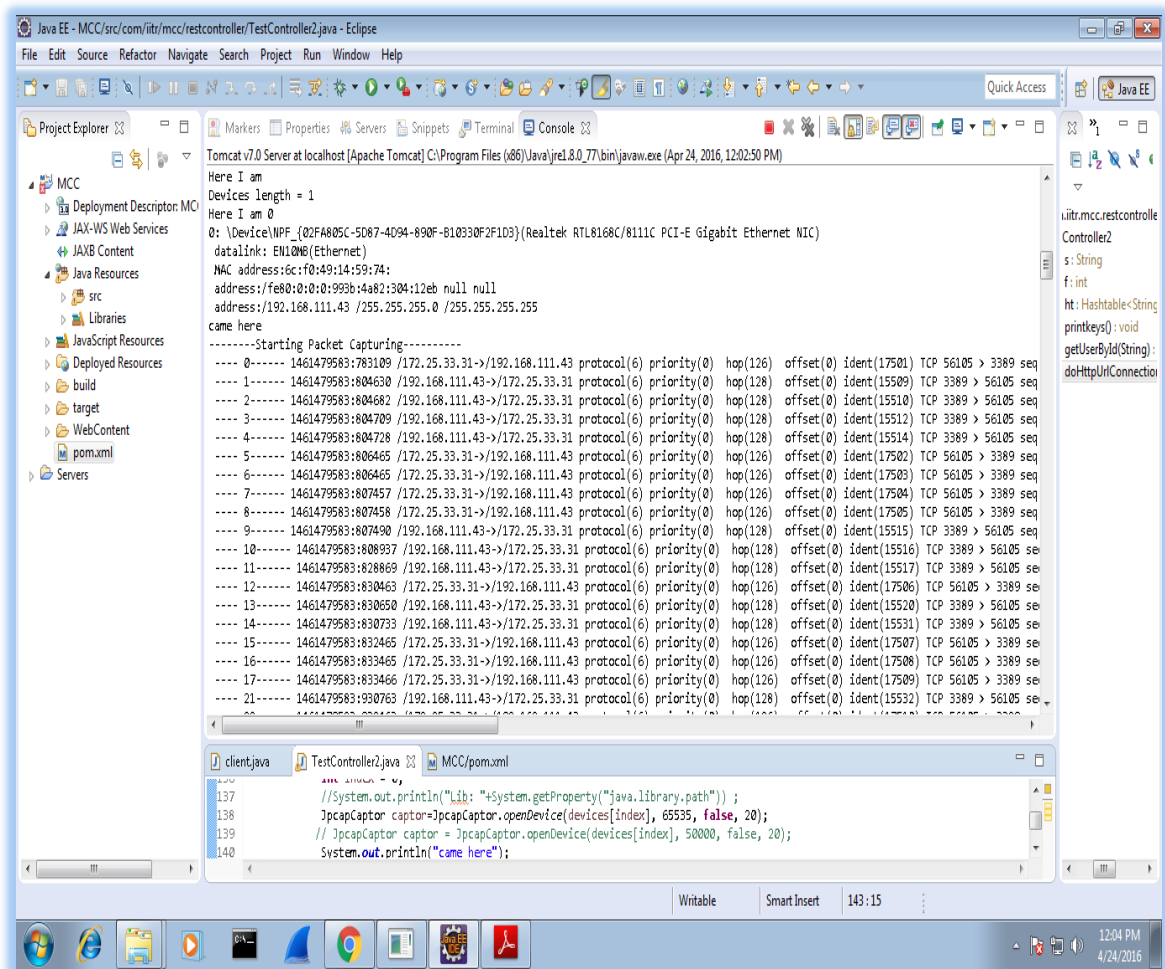
<head>
<!-- Comment for Debugging -->

<base href="/">
<!-- DNS Prefetch for the domains hotstar is using -- Sai @12/24/2015 -->
<link rel="dns-prefetch" href="http://www.hotstar.com" />
<link rel="dns-prefetch" href="http://b.scorecardresearch.com" />
<link rel="dns-prefetch" href="https://fonts.googleapis.com">
<link rel="dns-prefetch" href="http://search.hotstar.com" />
<link rel="dns-prefetch" href="http://account.hotstar.com" />
<link rel="dns-prefetch" href="http://www.google-analytics.com" />
<link rel="dns-prefetch" href="http://media1-starag.startv.in" />
<link rel="dns-prefetch" href="http://media0-starag.startv.in" />
<link rel="dns-prefetch" href="http://media2-starag.startv.in" />
<link rel="dns-prefetch" href="https://sb.scorecardresearch.com" />
<link rel="dns-prefetch" href="http://www.googleadservices.com" />
<link rel="dns-prefetch" href="http://www.googletagmanager.com" />
<link rel="dns-prefetch" href="https://staticxx.facebook.com" />
<link rel="dns-prefetch" href="https://googleads.g.doubleclick.net" />
<link rel="dns-prefetch" href="https://fonts.gstatic.com" />
```

Figure 4-8: Response at client site

4.9.2. Output at the cloudlet site :

Output at the cloudlet site of capturing the packets is shown in below fig. 4-6. The packets are captured and processed in order to extract data and store it in cloudlet. Since cloudlet concept is new and is not yet globally deployed into our current network architecture so the various video streaming service provider can exploit the cloudlet concept in storing those videos into the cloudlet which has a larger number of view count during the packet capturing phase. The video streaming service provider should deploy there secure application for the processing and storing data to their format, which will be obtained during streaming from their server. The detailed information of the packets are presented as the output below during watching the video on YouTube.



```
Java EE - MCC/src/com/itrr/mcc/restcontroller/TestController2.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help

Tomcat v7.0 Server at localhost [Apache Tomcat] C:\Program Files (x86)\Java\jre1.8.0_77\bin\javaw.exe (Apr 24, 2016, 12:02:50 PM)

Here I am
Devices length = 1
Here I am 0
0: (Device)\NPF_{02FA805C-5D07-4D94-890F-B10330F2F1D3}(Realtek RTL8168C/8111C PCI-E Gigabit Ethernet NIC)
datalink: EN10MB(Ethernet)
MAC address:fc:f0:49:14:59:74;
address:/fe80:0:0:0:993b:4a82:304:12eb null null
address:/192.168.111.43 /255.255.255.0 /255.255.255.255
came here
-----Starting Packet Capturing-----
---- 0----- 1461479583:783109 /172.25.33.31->/192.168.111.43 protocol(6) priority(0) hop(126) offset(0) ident(17501) TCP 56105 > 3389 seq
---- 1----- 1461479583:804630 /192.168.111.43->/172.25.33.31 protocol(6) priority(0) hop(128) offset(0) ident(15509) TCP 3389 > 56105 seq
---- 2----- 1461479583:804682 /192.168.111.43->/172.25.33.31 protocol(6) priority(0) hop(128) offset(0) ident(15510) TCP 3389 > 56105 seq
---- 3----- 1461479583:804709 /192.168.111.43->/172.25.33.31 protocol(6) priority(0) hop(128) offset(0) ident(15512) TCP 3389 > 56105 seq
---- 4----- 1461479583:804728 /192.168.111.43->/172.25.33.31 protocol(6) priority(0) hop(128) offset(0) ident(15514) TCP 3389 > 56105 seq
---- 5----- 1461479583:806465 /172.25.33.31->/192.168.111.43 protocol(6) priority(0) hop(126) offset(0) ident(17502) TCP 56105 > 3389 seq
---- 6----- 1461479583:806465 /172.25.33.31->/192.168.111.43 protocol(6) priority(0) hop(126) offset(0) ident(17503) TCP 56105 > 3389 seq
---- 7----- 1461479583:807457 /172.25.33.31->/192.168.111.43 protocol(6) priority(0) hop(126) offset(0) ident(17504) TCP 56105 > 3389 seq
---- 8----- 1461479583:807458 /172.25.33.31->/192.168.111.43 protocol(6) priority(0) hop(126) offset(0) ident(17505) TCP 56105 > 3389 seq
---- 9----- 1461479583:807490 /192.168.111.43->/172.25.33.31 protocol(6) priority(0) hop(128) offset(0) ident(15515) TCP 3389 > 56105 seq
---- 10----- 1461479583:808937 /192.168.111.43->/172.25.33.31 protocol(6) priority(0) hop(128) offset(0) ident(15516) TCP 3389 > 56105 seq
---- 11----- 1461479583:828869 /192.168.111.43->/172.25.33.31 protocol(6) priority(0) hop(128) offset(0) ident(15517) TCP 3389 > 56105 seq
---- 12----- 1461479583:830463 /172.25.33.31->/192.168.111.43 protocol(6) priority(0) hop(126) offset(0) ident(17506) TCP 56105 > 3389 seq
---- 13----- 1461479583:830650 /192.168.111.43->/172.25.33.31 protocol(6) priority(0) hop(128) offset(0) ident(15520) TCP 3389 > 56105 seq
---- 14----- 1461479583:830733 /192.168.111.43->/172.25.33.31 protocol(6) priority(0) hop(128) offset(0) ident(15531) TCP 3389 > 56105 seq
---- 15----- 1461479583:832465 /172.25.33.31->/192.168.111.43 protocol(6) priority(0) hop(126) offset(0) ident(17507) TCP 56105 > 3389 seq
---- 16----- 1461479583:833465 /172.25.33.31->/192.168.111.43 protocol(6) priority(0) hop(126) offset(0) ident(17508) TCP 56105 > 3389 seq
---- 17----- 1461479583:833466 /172.25.33.31->/192.168.111.43 protocol(6) priority(0) hop(126) offset(0) ident(17509) TCP 56105 > 3389 seq
---- 21----- 1461479583:930763 /192.168.111.43->/172.25.33.31 protocol(6) priority(0) hop(128) offset(0) ident(15532) TCP 3389 > 56105 seq

client.java TestController2.java MCC/pom.xml
137 //System.out.println("lib: "+System.getProperty("java.library.path"));
138 JpcapCaptor captor=JpcapCaptor.openDevice(devices[index], 65535, false, 20);
139 // JpcapCaptor captor = JpcapCaptor.openDevice(devices[index], 50000, false, 20);
140 System.out.println("came here");
```

Figure 4-9: Packet capturing at cloudlet site

4.10. STORED DATA

The content of the file is stored with the help of jpcap library is showed in the following snapshot. The file clearly shows that the packet are successfully captured and stored in the pcap format while streaming the video from our amber cloud. In the snapshot we can see that from our client machine having *ip* address 192.168.111.43 a video request is made for video streaming purpose to the cloud server. Successful packet capturing is done which are coming from our “AMBAR” cloud. In the below snapshot we can see that a HTTP request is made from our client machine (192.168.111.43) to our AMBAR cloud (192.168.111.69). The response coming from our cloud can be seen below in the snapshot. So these are the packets which were captured using the written code.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.111.43	192.168.111.69	TCP	54	51496 → 80 [FIN, ACK] Seq=1 Ack=1 Win=65335 Len=0
2	0.000123	192.168.111.43	192.168.111.69	HTTP	624	GET /videos/sintel_new/sintel_2400k_10.m4s HTTP/1.1
3	0.000822	192.168.111.69	192.168.111.43	TCP	60	80 → 51496 [ACK] Seq=1 Ack=2 Win=629 Len=0
4	0.005363	192.168.111.69	192.168.111.43	TCP	1514	[TCP segment of a reassembled PDU]
5	0.005623	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=1461 Ack=571 Win=408 Len=1460
6	0.005626	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=2921 Ack=571 Win=408 Len=1460
7	0.005649	192.168.111.43	192.168.111.69	TCP	54	51504 → 80 [ACK] Seq=571 Ack=4381 Win=16425 Len=0
8	0.005741	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=4381 Ack=571 Win=408 Len=1460
9	0.006194	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=5841 Ack=571 Win=408 Len=1460
10	0.006197	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=7301 Ack=571 Win=408 Len=1460
11	0.006199	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=8761 Ack=571 Win=408 Len=1460
12	0.006219	192.168.111.43	192.168.111.69	TCP	54	51504 → 80 [ACK] Seq=571 Ack=10221 Win=16425 Len=0
13	0.006585	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=10221 Ack=571 Win=408 Len=1460
14	0.007445	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=11681 Ack=571 Win=408 Len=1460
15	0.007448	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=13141 Ack=571 Win=408 Len=1460
16	0.007477	192.168.111.43	192.168.111.69	TCP	54	51504 → 80 [ACK] Seq=571 Ack=14601 Win=16425 Len=0
17	0.010785	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=14601 Ack=571 Win=408 Len=1460
18	0.010999	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=16061 Ack=571 Win=408 Len=1460
19	0.011023	192.168.111.43	192.168.111.69	TCP	54	51504 → 80 [ACK] Seq=571 Ack=17521 Win=16425 Len=0
20	0.011693	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=17521 Ack=571 Win=408 Len=1460
21	0.011696	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=18981 Ack=571 Win=408 Len=1460
22	0.011699	192.168.111.69	192.168.111.43	TCP	1514	80 → 51504 [ACK] Seq=20441 Ack=571 Win=408 Len=1460
23	0.011732	192.168.111.43	192.168.111.69	TCP	54	51504 → 80 [ACK] Seq=571 Ack=21901 Win=16425 Len=0

▶ Frame 1: 54 bytes on wire (432 bits), 54 bytes captured (432 bits)
▶ Ethernet II, Src: Giga-Byt_14:59:74 (6c:f0:49:14:59:74), Dst: Dell_44:d1:7f (b8:ac:6f:44:d1:7f)
▶ Internet Protocol Version 4, Src: 192.168.111.43, Dst: 192.168.111.69
▶ Transmission Control Protocol, Src Port: 51496 (51496), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 0

```
0000 b8 ac 6f 44 d1 7f 6c f0 49 14 59 74 08 00 45 00 ..oD..I.I.Yt..E.
0010 00 28 6a 8d 40 00 00 06 00 00 c0 a8 6f 2b c0 a8 .(j_@... ..o+..
0020 6f 45 c9 28 00 50 25 74 a4 f7 fc 4c 74 29 50 11 oE.(.P%t ...Lt)P.
0030 ff 37 5f dc 00 00 .7 ...
```

Figure 4-10: Captured Packets

5. RESULTS

5.1. OUTPUT

5.1.1 Output showing size of webpage

The website which are accessed by the client machine are shown in following Fig. and the corresponding size is also shown. The response time not only depends upon the round trip time, client execution time, server execution time but also on the size of the data which is being carried over the network. A webpage may contain many resources like static images (jpeg, png, gif, etc.), linked file through hyperlinks, metadata, JavaScript, forms, etc. Following websites are used for the response time calculation.

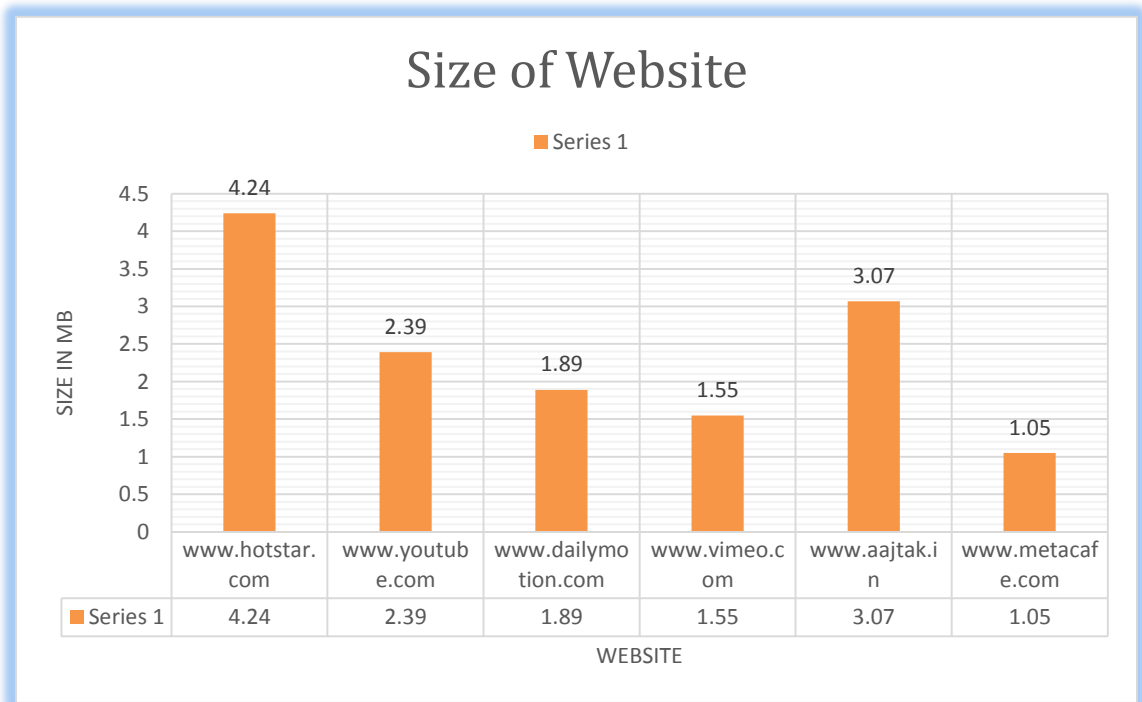


Figure 5-1: Size of webpage

5.2 RESPONSE TIME OUTPUT

5.2.1 Output obtained without cloudlet

The chart below demonstrate the response time when the cloudlet did not have any data in its hash table. The response time is calculated three times for a better result analysis. Every website is accessed without the use of cloudlet concept. And the Response Time (**RT**) is recorded every time and plotted below. All the results are calculated on different traffic conditions in the network.

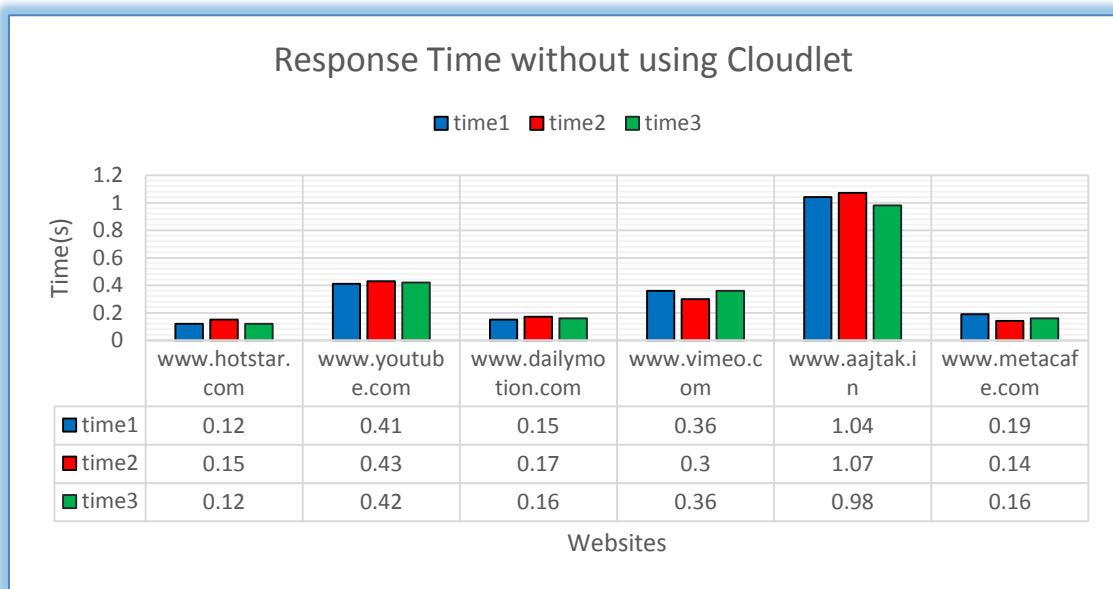


Figure 5-2: Response Time without using Cloudlet

Snapshot below shows the response time of video streaming site hotstar.com. The response time shown below is actually the output which is observed at the client site. The response time is coming from the actual server not by using the cloudlet concept.

```
URL: http://localhost:8080/MCC/rest/hello13/id?id=http://www.hotstar.com
start time = 1461836755175 || end time = 1461836755187
Time taken in getting response : 0.12 seconds
```

Figure 5-3: Output obtained without using Cloudlet

5.2.2 Output obtained using cloudlet

The chart below demonstrate the response time when the cloudlet has the data present in its hash table. Every time when a request is made for a particular website, request ID is first extracted from the request. This ID is checked in the hash table. Below represented graph is the case in which every time the response is sent back by the cloudlet itself.

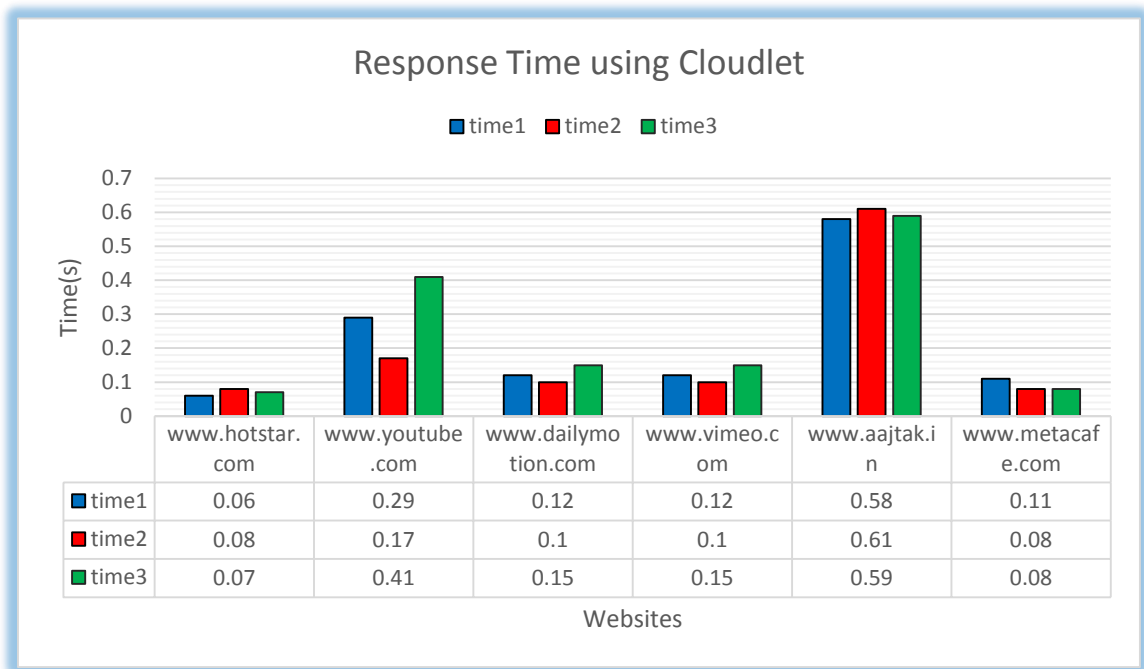


Figure 5-4: Response Time using Cloudlet

Snapshot below shows the response time of video streaming site hotstar.com. The response time shown below is also observed at the client site. The response time is not coming from the actual server but by coming from the cloudlet.

```
URL: http://localhost:8080/MCC/rest/hello13/id?id=http://www.hotstar.com
start time = 1461836332266 || end time = 1461836332272
Time taken in getting response : 0.06 seconds
```

Figure 5-5: Output obtained using Cloudlet

5.3 COMPARISION

The following bar chart clearly shows that by using cloudlet we can perform power aware access to the web site. In the following Fig. 5-6 comparison the red bar shows the decrement in the response time when reply is coming from the cloudlet while the blue bar shows the actual response time. The graph below is constructed on the assumption that mobile device is consuming a constant amount of power every time the same website is running on client device. So with the time decrease energy also gets decreased and hence the battery of the mobile device gets saved. Also the network cost also gets reduced by using cloudlet.

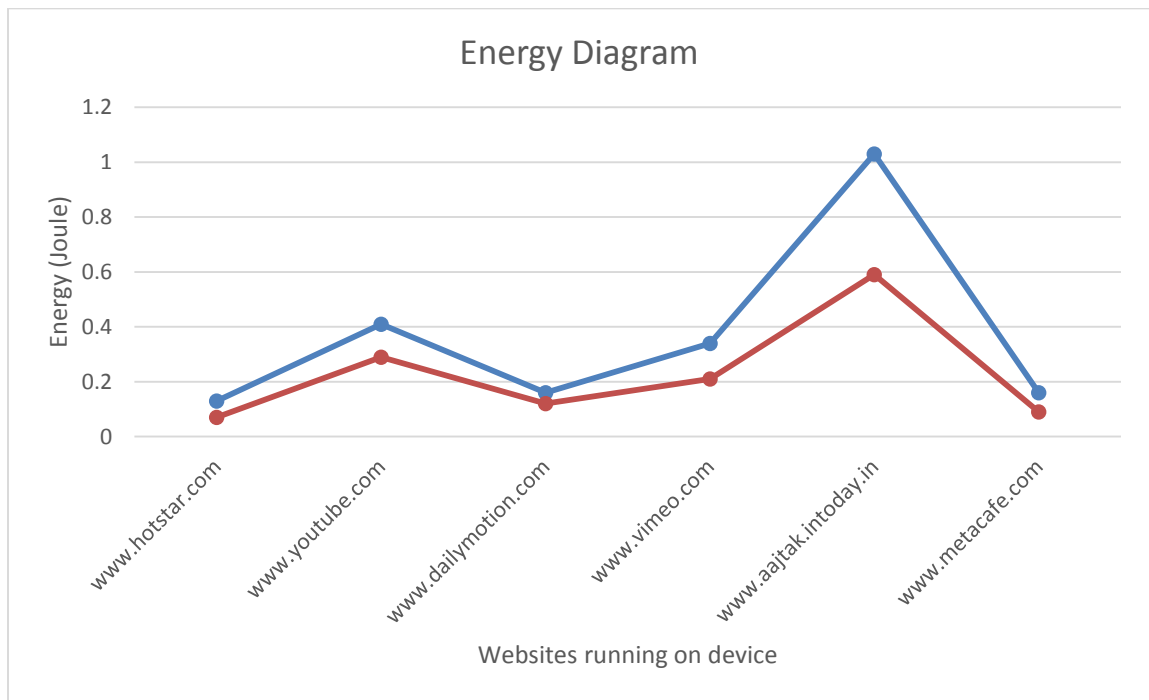


Figure 5-6: Comparison of Energy Consumption

Let's consider YouTube response time.

$$Energy\ saved = \frac{Energy_1 - Energy_2}{Energy_1} * 100\% \tag{5.1}$$

Where, **Energy₁**: The energy consumed by a mobile device without using the cloudlet.

Energy₂ : The energy consumed by a mobile device using the cloudlet concept.

Since **Energy = Power * Time** therefore

$$\mathbf{Energy\ saved\ \% = \frac{(Power*Time_1)-(Power*Time_2)}{Power*Time_1} * 100\ \%} \quad (5.2)$$

$$\mathbf{Energy\ saved\ \% = \frac{0.41 - 0.29}{0.41} * 100\ \%}$$

$$\mathbf{Energy\ saved = 29.27\ \%}$$

Similarly energy saved calculated for other websites and the range of percentage we obtained is [25.00 % – 65.00 %] .

6. CONCLUSION AND FUTURE WORK

In this report the discussed algorithm can be used for a seamless high quality video services in an energy efficient manner. Among the traffic generated by global mobile devices nearly 69% of traffic is due to video demand so there is a lot of scope in order to develop the various energy efficient models. Till now a cloud named “AMBAR” is established in which many virtual machines for different purposes like transcoding, video streaming are established. The video that are stored into the cloud are in encrypted format. Each video has its own media presentation description (MPD). A media presentation description (MPD) describes segment information (timing, URL, media characteristics like video resolution and bit rates), and can be organized in different ways such as Segment List, Segment Template, Segment Base and Segment TimeLine, depending on the use case. Up to 2020 the computing from local desktop will shift to the cloud via networking devices. We can say Hybrid Cloud technology will rule the world. Now after focusing on the data center energy efficiency there is also a need to focus on the energy consumption by the transmission and switching networks. Transmission and switching networks is very much important for connecting the users with cloud. The 5G technology is on the verge to come so new scope of solving the energy efficiency problem are there such as 5G technology will be using femtocloud for faster processing and many more benefits [6]. There are no bound to the innovation of models that are required to save every possible bit of energy.

For the future work a proper storage technique should be implemented in the cloudlet for storing the data. Database can be used for storing an entity describing the video files and for storage, streaming and managing purpose use image-video management platform like cloudinary.

REFERENCES

- [1] Yujin Li and Wenye Wang, “The Unheralded Power of Cloudlet Computing in the Vicinity of Mobile Devices”, Global Communications Conference (GLOBECOM), IEEE 2013. Pp 4994-4999.
- [2] Yunmin Go, Oh Chan Kwon, and Hwangjun Song, “An Energy-Efficient HTTP Adaptive Video Streaming with Networking Cost Constraint over Heterogeneous Wireless Networks” , IEEE Transaction on multimedia, Vol. 17, No. 9, September 2015. Pp 1646-1657.
- [4] Yang Zhang, Dusit Niyato, Senior Member, IEEE, and Ping Wang, Senior Member, IEEE, “Offloading in Mobile Cloudlet Systems with Intermittent Connectivity”, IEEE Transactions on Mobile Computing, VOL. 14, NO. 12, DECEMBER 2015
- [3] T. Stockhammer, “Dynamic adaptive streaming over HTTP: Standards and design principles”, Proc. ACM Conference, Multimedia Syst., 2011. Pp. 133–144.
- [4] Wei-Tsung Su and Kiat Siong Ng. “Mobile Cloud with Smart Offloading System”, IEEE/CIC International Conference on Communications in China (ICCC), 2013. Pp 680-685.
- [5] Bo Gao, Ligang He, Limin Liu, Kenli Li and Stephen A. Jarvis, “From Mobiles to Clouds: Developing Energy-aware Offloading Strategies for Workflows”, ACM/IEEE 13th International Conference on Grid Computing, 2012. Pp 139-146.
- [6] J. Liu and Y.-H. Lu, “Energy Savings in Privacy-Preserving Computation Offloading with Protection by Homomorphic Encryption”, Proceedings of the 2010 international conference on Power Aware computing and systems (HotPower '10), Vancouver, BC, Canada, 2010
- [7] Yi Dong, Liang Zhou, Jianxin Chen, Baoyu Zheng, Jingwu Cui, “Energy Efficient Virtual Machine Consolidation in Mobile Media Cloud”, IEEE Conference Publication 2015. Pp 248-252.

- [8] Yi Anubha Jain, Manoj Mishra, Sateesh Kumar Peddoju, Nitin Jain, " Energy Efficient Computing- Green Cloud Computing", in International Conference on Energy Efficient Technologies for Sustainability (ICEETS), Nagercoil, April, 2013.Pp 978-982.
- [9] Sergio Barbarossa, Stefania Sardellitti, and Paolo Di Lorenzo, "Distributed mobile cloud computing over 5G heterogeneous networks" in IEEE Signal Processing Magazine, November 2014.
- [10] Zichuan Xuy, Weifa Liangy, Wenzheng Xuzy, Mike Jiay, and Song Guo, "Capacitated Cloudlet Placements in Wireless Metropolitan Area Networks" in IEEE Conference on Local Computer Networks, 2015.
- [11] Ramona-Oana Grigoriu, Giorgian, Neculoiu, Ionela Halcu, Virginia Cristiana Săndulescu, Oana Niculescu-Faida, Mariana Marinescu, Viorel Marinescu, "Energy Efficiency in Cloud Computing and Distributed Systems" in IEEE Conference Publications, 2013. Pp 1-5.
- [12] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," IEEE Pervasive Comput., vol. 8, no. 4, Oct./Dec. 2009, Pp. 14–23.
- [13] Z. Li, C. Wang, and R. Xu, "Computation offloading to save energy on handheld devices: A partition scheme," in Proc. Int. Conf. Compilers, Archit., Synthesis Embedded Syst., 2001, pp. 238–246.
- [14] K. Kumar and Y.-H. Lu, "Cloud computing for mobile users: Can offloading computation save energy," IEEE Comput., vol. 43, no. 4, pp. 51–56, Apr. 2010.
- [15] Yujin Li and Wenye Wang, "The Unheralded Power of cloudlet Computing in the Vicinity of Mobile Devices", IEEE Global Communications Conference (GLOBECOM), 2013.
- [16] Yujin Li and Wenye Wang," Can Mobile Cloudlets Support Mobile Applications?", IEEE INFOCOM 2014.