ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

Implementation of Virtual Machine Based Cloudlet Architecture Using MQTT Protocol By Providing Reliability Service

Atul, Dr. Rekha Patil

M. Tech Student, Dept. of CSE, PDA College of Engineering, Kalaburagi, Karnataka, India

Associate Prof. Dept. of CSE, PDA College of Engineering, Kalaburagi, Karnataka, India

ABSTRACT: Cloud based storage services has been popular in recent times due to the high degree of reliability and "access from everywhere" advantage. Google drives, SkyDrive, Dropbox are some of the popular storage service providers. One of the major issues with cloud based storage is the need for high bandwidth internet access. For frequently exchanged data in the cloud, bandwidth usage for frequent uploading and downloading of common files are extremely high. In this work, we have addressed this issue by proposing an additional layer of storage as a service at the edge of the network in conjunction of the core storage cloud. The proposed system offers a local cloudlet implementation where a broker coordinates with multiple virtual machines to offer storage as a service within a local network. A file is split into multiple chunks based on virtual machines and their resource availability and stored in multiple virtual machines. The chunks are also stored at the core. The broker maintains a metadata of the files and their corresponding chunks with respective virtual machines. We also provide reliability service in a cloudlet, in case a virtual machine holding a chunk of a file fails at the time of the file request, then that particular chunk is downloaded from the core where as the available local chunks are provided from the edge virtual machines. This saves bandwidth and offers very low latency solution for frequently accessed files.
The proposed architecture is implemented on the top of MQTT protocol which offers QoS at the core protocol level with low protocol header overhead. Unlike the conventional cloudlet where files are stored in a distributed file system, we propose a novel technique to keep the files in the main memory of the virtual machines through No-Sql- Redis server. This reduces the disk and IO latency significantly. Results show that the proposed system provides a fair load sharing between the virtual machines and offers storage as a service at the edge through proposed cloudlet architecture.

KEYWORDS: Broker, Cloudlet, MQTT, Virtual Machine.

I. INTRODUCTION

As single web server not able to handle too many client requests, it will exhaust with its resource. Because of this client suffers a slow connection. To avoid this we use cloud computing. Cloud computing extends server as multiple server interconnected with each other, so even if one of the server loses its resources, it forwards to another server. It also provides multiple data center by means of which same data stored in various locations. By this, it ensures data integrity. So basically everything we do in the cloud is as a service.

Weight, Power, Size, CPU and Memory are major constraint of the hardware system, so we go for cloud computing. Most of the cloud computing systems depend on web technology environment. For example, if we have to watch a video on YouTube we have to have good bandwidth. Even though YouTube is in the cloud itself, unless we have good bandwidth we are not able to watch the video which is jitter free. So despite the fact that cloud computing solves the problem of resource exhaustion at server level it cannot solve the bandwidth problem between the cloud and its clients. So to overcome this problem run a cloud services using the local cloud.

Local cloud is created using hotspot like Wi-Fi. Many edge devices are connected to the hotspot to create an edge network. Suppose consider, we have Wi-Fi network. If this is used to connect ten small devices then we can create a small cloud within this network itself. This small cloud is known as cloudlet. Now we can run small cloud services on this edge network. This Cloudlet can also be connected to core cloud using hotspot like Wi-Fi.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

So, cloudlet is small tiny cloud services running in the client devices. Considering there are several client devices connected to the same local network like Wi-Fi network or Ethernet LAN and if we can run fraction of service on each of these client machine and combine them into small local cloud fragment that becomes Cloudlet. Then each of these devices small storage capacity used to build large storage using cloudlet. The cloudlet is used to reduce the latency between a cloud and its client thus providing a faster access of the data. We also provide reliability service within the cloudlet such that, if any client machine holding a file chunk is failed during file download request, then that particular file chunk will be retrieved using core cloud connected with it.

The rest of the paper is organized as follows. Section I presents an introduction to the project, Section II outlines the related work. Section III provides a brief introduction to MQTT protocol. Section IV gives details of the proposed system. Implementation of the proposed system explained in section V. Section VI analyses the result of the proposed system. Then finally concludes the paper and outlines future works in section VII.

II. RELATED WORK

In this section surveys outline different fault tolerance mechanisms carried out to provide reliability service in a cloud and it also analyses impact of cloudlet over cloud. Instead of implementing fault tolerance solution to all components, the fault tolerance technique is applied to only the significant component of cloud application [1].Using Low Latency Fault Tolerance, applications which are developed using TCP IP socket or web services are replicated in the cloud without any modification to original application [2].Reliability of a node calculated based on execution of a node within the time limit of the application [3].The fault tolerance mechanism is implemented using a dedicated system layer called as Fault tolerance manager [4].To reduce the service time and increase the system availability, virtualization fault tolerance used in the cloud where reliability of node is calculated based on success rate of a node [5].Analysis of the impact of the cloudlet over cloud for the application scenario discussed [6].Analysis of the performance of different cloudlet architecture is discussed [7].

III. MQTT PROTOCOL

The most important protocol used for cloudlet is a MQTT (Message Queuing Telemetry Transport) protocol which is developed by IBM. It is de facto protocol for machine to machine communication. In MQTT protocol, the messages communicated between the machines are through byte stream.

MQTT protocol provides the packet header which is 127 bits, which is significantly less compared to the web based protocol packet header. Communication between nodes occurs through the MQTT Server. To connect with MQTT server, each node must create an object of MQTT client. So, every node acts as MQTT client with respect to MQTT server, as shown in Fig. 1. To connect with MQTT server, MQTT assigns the unique id to each communicating node. Whenever a MQTT client wants to use MQTT services, it must register with a unique id assigned to it. When any node wants to access the service, it must subscribe to a particular topic in MQTT server. The whole MQTT protocol works with simple messaging service commonly known as publish-subscribe protocol. One node publishes to MQTT Server and another node subscribes. Any number of nodes can subscribe with a single service. Whenever nodes subscribe to a particular topic in MQTT server, MQTT server notes down which topic is subscribed by how many numbers of nodes. So whenever message gets published into MQTT to a particular topic then MQTT sever asynchronously pushes that message to every node connected with that topic. Therefore MQTT is extensively used in cloudlet and IOT oriented solution because the developer does not have to do other than subscribe and publish to a topic as the rest of the things are carried out by MQTT server.

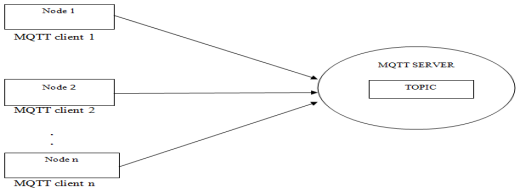

Fig. 1.Nodes Connecting to MQTT Server

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

IV. PROPOSED SYSTEM

Cloudlet created at the edge of the network using hotspot such as Wi-Fi is used to connect several nodes such as a laptop. There is a special node in the cloudlet called as Broker which hosts the local MQTT server. Remaining nodes in cloudlet act as virtual machines by running a storage service using No-Sql redis server in it. Broker and virtual machines which are part of cloudlet are connected to local MQTT server using same hotspot network. Virtual machine's state in the cloudlet is tracked by the broker. Whenever the broker gets a job, it divides the job and assigns it to the virtual machine of the Cloudlet.

The main objective of implementing cloudlet is to access the data quickly compare to accessing the data from core cloud, which requires more time than cloudlet takes. Cloudlet is a small cloud, so we connect it with core cloud to have large data storage service and also to provide data integrity service. So we connected broker in a cloudlet to Microsoft SkyDrive which is a core cloud. Proposed system Architecture shown in the Fig. 2.

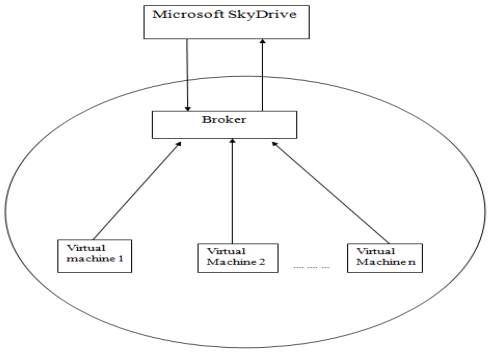

Fig. 2.Architecture of the Proposed System

The following main functionalities are carried out by proposed system are -

1. Whenever virtual machine joins the cloudlet within the same Wi-Fi network, the broker will create account of how many virtual machines are connected in the network. Whenever new virtual machines are connected, the broker creates table entry for each virtual machine having its IP address, available main memory, residual processing capability and penalty. Penalty value is used to specify how many times a particular virtual machine has failed. For new virtual machine joining the cloudlet, its penalty is zero. So, as and when virtual machines are connected, the broker keeps on adding details in the virtual machines (VMS) table entry.
2. Whenever a virtual machine which is part of cloudlet tries to upload the file into the cloudlet, the broker will perform the following tasks.
   a. Extract the file content.
   b. Find the number of the active virtual machines available.
   c. It contains split value which notifies the broker, how the file should be spit, into how many number of chunks. For split=n, broker must select n number of virtual machines having highest available main memory (RAM) in the cloudlet for storing file chunks.
   d. Calculate memory fraction of selected virtual machine.
   e. Split the file into file chunks according to the memory fraction calculated for each virtual machine. It creates a table entry called Meta data of files which specifies which chunk of the file is stored in which virtual machine.
   f. After making entry in Meta data of file, broker will store the file chunks in respective main memory of the virtual machines through No-Sql redis server.
   g. It not only stores the file chunks in cloudlet, it also uploads the file chunks in Microsoft SkyDrive.
3. When any one of the virtual machine, which is part of the cloudlet, requests for download of the file which is stored in the cloudlet, the broker will perform the following tasks.
   a. Broker first checks whether virtual machines holding file chunks are active using information in Meta data of files table and VMS table created at the broker. When all virtual machines holding file chunks are

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

active, then, file chunks are retrieved from the redis server of each respective virtual machine and the whole file is downloaded at the virtual machine requesting the file download.

b. If any one of the virtual machine holding file chunk is failed during download request, then broker will download file chunk of failed virtual machine from the SkyDrive. Remaining file chunks are retrieved from the redis server of the respective active virtual machines. Then the whole file is downloaded at the virtual machine requesting the download.

4. When the failed virtual machine tries to become part of the cloudlet, then broker will perform the following tasks.
   a. Increase the penalty of the corresponding virtual machine and add its details in VMS table entry.
   b. Decrease the available main memory of corresponding virtual machine by Penalty*(Reduction Factor). Update this information in VMS table entry, where the reduction factor is numerical value used for that application.

V. IMPLEMENTATION

In our work, we used local MQTT server known as Mosquitto Server to implement the cloudlet. This MQTT server is used to connect the nodes in a cloudlet. Mosquitto server runs in a special node called as Broker in a cloudlet. Remaining nodes act as Virtual machines in a cloudlet by running the storage service using No-Sql redis server. In our work Broker subscribes with mosquitto server to a topic rupam/cloudlet. Each Virtual machine subscribes with mosquitto server to a topic of its own IP address. In our work we used the mosquitto server which runs in window OS. It allows us to connect maximum 1023 virtual machines.

Following are the implementation details of the proposed system.

A. *Connecting Broker and Virtual machine to mosquito server.*

Step 1: Broker with its unique id connects to mosquito server and subscribe with it to the topic rupam/cloudlet.
Step 2: Each Virtual machine with its unique id connects to mosquito server and subscribe with it to the topic of its own IP address.

B. *Connecting Broker to Microsoft SkyDrive for data integrity and storage services.*

C. *Virtual machines join Broker by creating a cloudlet Setup.*

Step 1: Once virtual machines connect to mosquitto server, each virtual machine requests for join by publishing its IP address, CPU capability and available main memory to mosquitto server to the topic rupam/cloudlet.
Step 2: As rupam/cloudlet is subscribed by broker, the join request reaches the broker. For new virtual machine joining the broker its penalty is zero. For failed virtual machine rejoin its penalty is increased and its corresponding available main memory is decreased. Broker adds these details of each virtual machine to its VMS table entry. Broker updates a VMS table entry for every 5 seconds.

D. *Virtual machine is uploading a file in the cloudlet.*

Step 1: Virtual machine uploads a text file in the cloudlet by publishing a file to mosquito server to the topic rupam/cloudlet.
Step 2: As rupam/cloudlet is subscribed by broker, the mosquitto server asynchronously sends the file to the broker.
Step 3: Broker selects the number of virtual machines equal to split size value specified at the broker. It selects those virtual machines which are having the highest available main memory. It further calculates memory fraction for each selected virtual machine by dividing available main memory of each virtual machine by total main memory of all selected virtual machine. Broker, then splits the whole file content into file chunks according to the memory fraction calculated for each selected virtual machine. It adds file chunks details in Meta data of files which specifies which chunk of the file is stored in which virtual machine.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

Step 4: Once broker splits the file into file chunks, it publishes these file chunks to mosquito server to the topic of IP address of the respective virtual machines which are selected for storing the file chunks in it.
Step 5: As every virtual machine is subscribed with mosquitto server with a topic of its own IP address, the file chunks corresponding to respective virtual machine are asynchronously sent by mosquito server to intended virtual machine.
Step 6: Virtual machines store the file chunks in their respective main memory through No-Sql redis server.

E. *Virtual machine request for a list of available files stored in a cloudlet.*

Step 1: Virtual machine requests for the available files in the cloudlet by publishing request to mosquitto server to the topic rupam/cloudlet.
Step 2: As rupam/cloudlet is subscribed by broker, the mosquitto server asynchronously sends the request to the broker.
Step 3: Broker retrieves the list of available files in the cloudlet using information stored in Meta data of files at the broker.
Step 4: Once the broker retrieves the list of available files, it publishes to mosquito server to the topic of IP address of a virtual machine which is has requested for the available files.
Step 5: As requested virtual machine is subscribed with broker with a topic of its own IP address, the mosquitto server asynchronously sends the available files to requested virtual machine.
Step 6: List of available files in the cloudlet are displayed at the virtual machine requested for the available files list.

F. *Virtual machine is downloading a file in a cloudlet when all the virtual machines holding file chunk are active.*

Step 1: Virtual machine requests for the file download by publishing file download request to mosquitto server to the topic rupam/cloudlet.
Step 2: As rupam/cloudlet is subscribed by broker, the mosquito server asynchronously sends the download request to the broker.
Step 3: Broker checks whether all the virtual machines having file chunks are active by using information available in the VMs table and Meta data of files table at the broker.
Step 4: If all virtual machines having file chunks are active then broker notifies each virtual machine holding a file chunk by publishing to mosquitto server to the topic of its virtual machine IP address.
Step 5: As every virtual machine is subscribed with a mosquito server with topic of its own IP address, the mosquitto server asynchronously sends the notify request to respective virtual machine which are holding chunk.
Step 6: Each virtual machine retrieves the file chunk data from its redis server and publishes it to mosquito server to the topic of requested virtual machine IP address.
Step 7: As file download requester virtual machine is subscribed with mosquitto server with a topic of its own IP address, the mosquitto server asynchronously sends the file chunks to a virtual machine which has requested the file download.
Step 8: File download requested virtual machine rearranges all the received file chunks in order and concatenates all the file chunks by displaying the file contents at virtual machine which has requested for the file download.

G. *Virtual machine downloading a file in a cloudlet when a virtual machine holding a file chunk fails.*

Step 1: Virtual machine requests for the file download by publishing file download request to mosquitto server to topic rupam/cloudlet.
Step 2: As rupam/cloudlet is subscribed by broker, the mosquito server asynchronously sends the download request to broker.
Step 3: Broker checks whether all the virtual machine having file chunks are active by using information available in the VMS table and Meta data of files table at the broker.
Step 4: If any one of the virtual machine holding a file chunk fails, then the broker retrieves the file chunk of failed virtual machine using Microsoft SkyDrive and the broker also notifies remaining active virtual machines holding file chunks by publishing to mosquitto server to the topic of its respective virtual machine IP address.
Step 5: As every virtual machine is subscribed with mosquito server with a topic of its own IP address, the mosquitto server asynchronously sends the notify request to respective active virtual machines which are holding the file chunk.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

Step 6: Each active virtual machines holding a file chunk retrieves the file chunk from its respective redis server and publish to mosquito server to the topic of IP address virtual machine IP requested for the file download.
Step 7: Broker publishes the retrieved file chunk of the failed virtual machine to mosquito server to the topic of IP address virtual machine which has requested for the file download.
Step 8: Virtual machine requesting file download is subscribed with mosquitto server with a topic of its own IP address, the mosquitto server asynchronously sends the file chunks to virtual machine which has requested for the file download.
Step 9: Virtual machine requesting file download arranges all the received file chunks in the order and concatenates all the file chunks, by displaying the file contents at the virtual machine requesting file download.

VI. RESULT ANALYSIS

Output analysis is carried out by using three virtual machines in the cloudlet for split=2.The output analysis graph is drawn with a sequence number which represents time in sequence versus latency time in millisecond. Each color line in the graph indicates virtual machine response with respect to broker. Virtual machine is indicated with a red line used for uploading a file and downloading a file in cloudlet.

Timing analysis of the virtual machines in the cloudlet presents the resource allocation by the broker to the virtual machine. In case if there are more number of the virtual machines, the resource allocated to each virtual machine differs. In such situation the timing or the latency from the response from each virtual machine generally varies to great deal due to jitter being introduced by the variable loads being suffered by different virtual machine. Virtual machines are having higher available main memory have a minimum load which intern have a less latency access time. So, in fig. 3(a) latency time for the virtual machine indicated with orange line is minimum, compared to virtual machine indicated with blue line compared with virtual machine indicated with red line.

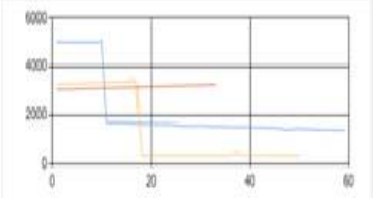

Fig. 3(a).Three virtual machine joining broker

During uploading a file in the cloudlet, the broker selects the virtual machines indicated with orange line and the blue line for storing the file chunks as they have high available main memory. There is increase in the virtual machine latency for a fraction of time while uploading a file this is because of jitter produced due to variable load at the virtual machine as shown in fig. 3(b).

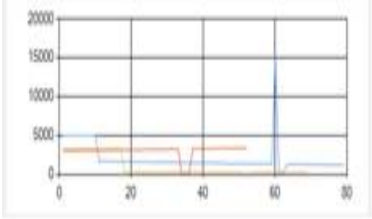

Fig. 3(b).Uploading File in a Cloudlet

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 6, June 2016

During downloading a file when all the virtual machines are active then there is no change in the latency, latency remains constant as shown in Fig. 3(c).

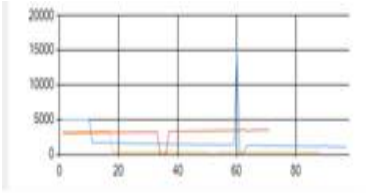

Fig. 3(c).Downloading File in a Cloudlet

There is advantage in selecting virtual machines having high available main memory in the cloudlet for storing file chunks, as they have low latency access time which helps in is faster accessing data in the cloudlet. Therefore we can clearly state that by implementing virtual cloudlet and managing the files and data through this virtual cloudlet we always obtain the optimum latency with minimum amount of jitter under realistic load condition.

VII. CONCLUSION AND FUTURE

In our work, we clearly demonstrate that cloudlet architecture using MQTT protocol reduces the latency time thereby providing the faster accessing of the data. We also provided the reliability service for the virtual machine in the cloudlet using available main memory of the virtual machine with the help of penalty based approach, where more data stored in virtual machine having higher available main memory compared to other virtual machine. If virtual machine fails then we retrieve the file chunk of the failed virtual machine from SkyDrive at the same time we increase the penalty of the failed virtual machine and decrease its available main memory by which it helps the broker to assign the task to virtual machines which are having high available memory.

REFERENCES

1. Z. Zheng, T. Zhou, M. Lyu, and I. King, “FTCloud: A Component Ranking Framework for Fault-Tolerant Cloud Applications,” in Proc. of ISSRE’10, 2010, pp. 398–407.
2. Wenbing W. Zhao, P. M. Melliar-Smith, and L. E. Moser, “Fault Tolerance Middleware for Cloud Computing,” in Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing, ser. CLOUD ’10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 67–74.
3. S. Malik and F. Huet, “Adaptive fault tolerance in real time cloud computing,” in Proceedings of the 2011 IEEE World Congress on Services, ser. SERVICES 2011, Jul. 2011, pp. 280–287
4. R. Jhawar, V. Piuri, and M. Santambrogio, “A Comprehensive Conceptual System-Level Approach to Fault Tolerance in Cloud Computing,” in Proc. of IEEE SysCon’12, Vancouver, Canada, Mar 2012, pp. 1–5.
5. Pranesh Das, Dr.Pabitra Mohan Khilar, “VFT: A Virtualization and Fault Tolerance Approach for Cloud Computing”, IEEE Conference on Information and Communication Technologies (ICT 2013), Kanyakumari, Tamil Nadu, in press, 2013.
6. D. Fesehaye, Y. Gao, K. Nahrstedt, and G. Wang, “Impact of Cloudlets on Interactive Mobile Cloud Applications,” in Enterprise Distributed Object Computing Conference (EDOC), 2012 IEEE 16th International, 2012, pp. 123–132.
7. A.S. Jaiswal,V.M. Thakare, S.S. Sherekar, “Performance Based Analysis Of Cloudlet Architectures In Mobile Cloud Computing” International Journal Of Computer Applications (0975 – 8887), National Conference On Recent Trends in information security 2015.