Implementation of FOG computing:

PURNA CHANDRA VINAY KUMAR. VEMALI

NARAYANASWAMY POOJA. BENNAGHATTA

Objective:

We have implemented functionality of a node in a distributed system in the context of Fog computing. Fog computing aims to put the cloud closer to the user for better quality of service and it also provides low latency, location awareness, support for wide-spread geographical distribution and mobility support Fog IoT applications.

Project Description:

In our model fog node sits between the cloud (is in core) and IOT (at the edge of the internet). In this project the main use of fog node is to reduce the response time-delay for requests coming from IoT nodes.

our project consists of three parts:

1. IoT node
2. Fog node
3. cloud node

IoT Node:

IoT node is the request initiator and it periodically sends request to the randomly choosen fog nodes.

Request packet contains:

1. Packet sequence number
2. Request Forwarding_Limit - randomly chosen number in the range of [2, 5] in seconds.
3. Request Processing_Time - randomly chosen number in the range of [3, 7] in seconds.
4. IP address of the IoT node
5. UDP port number of the IoT node.

The IoT node will be run in the terminal with the following command:

```
IoT node interval IP1 UDP1 IP2 UDP2 ...
```

Fog Node:

Fog node includes two tasks:

1) Receives requests and process it or forwarding the request. The decision of processing or forwarding the request will be based on the response time for the request.

Response time = Queuing delay + Processing delay.

Queuing delay= the summation of the processing times of all the currently queued requests at fog node.

Processing delay= Each request will include a parameter that will define the Processing_Time of the request.

When the response time has not reached the Max_Response_Time, the request is up for processing at a fog node, the processing is simulated by having a thread sleep processing time of the request. Once the time elapses, a response should be created and sent back to the request originator IoT node.

2) Periodically exchanging information with its 1-hop neighbors about their current queueing time so that they know who to forward a request when needed.

If fog node decides that it is too busy to accept this new request i.e it has reached the Max_Response_Time at fog node, it will check if it can forward the request to a neighbor or not.

This depends on the Forward_Limit of the request and how many times this request has beenforwarded so far. If the request has not reached the Forward_Limit, the fog node will identify the best neighbor among its 1-hop neighbors and forward the request to that neighbor.

Best Neighbor: The one that reported the smallest response time in the most recent round with an exception that the best neighbor cannot be the neighbor who has forwarded this request to the current

fog node. The best neighbor is the neighbor that the request is just received from and an offload is needed, the fog node sends the request to the second best neighbor (To avoid loop scenario).

Selection of the best neighbor is based on the queueing time of the 1-hop neighbors. This information is exchanged every 3 secs among 1-hop neighbors.

A fog node will be run in the terminal with the following command:

```
FogNode Max_Response_Time t MY_IP MY_UDP C TCP0 N1 TCP1 N2 TCP2
```

Cloud Node:

Cloud node puts the incoming request from fog node in its processing queue and processes the request from its queue one after the other, processing operation should be simulated by having a thread sleep for processing time of the request. Once this time elapses, a response is created and sent back to the request originator IoT node.

The cloud node will be run in the terminal with the following command:

```
CloudNode MY_IP MY_TDP
```

Difficulties encountered:

1. To establish connect between fog nodes using threads.

2. Maintaining different queues for requests and responses and forwarding them to the best neighbors.

Contributors:

Vinay Kumar:

I mainly focused on developing the functionality of the FOG node to establish connection to the IoT, neighboring nodes and the cloud node. The Fog node after successfully establishing all the necessary connections, starts the threads to communicate to the Iot, cloud and neighboring fog nodes. The request received from the Iot node, which is implemented by my co-contributor, processes or forwards to the Fog node or the cloud node, which is implemented by my team-mate.

When I was struck in in designing the program structure with the help of my team mate, I was able to come out with an effective solution for the problem. Another challenge I faced while implementing the threads for communicating with the neighbors.

Pooja:

My task is to implement the functionality of the IoT node to effectively send requests to the fog node and receive the responses from the Fog and cloud nodes and the cloud node that receives requests from the Fog node, implemented by my team-mate, and send responses to the IoT node.

I faced challenge while implementing the different queues for requests and responses, in the multithreading. With the help of my team-mate, I was able to effectively implement and maintain queues.