CS 6390.001 Advanced Computer Networks

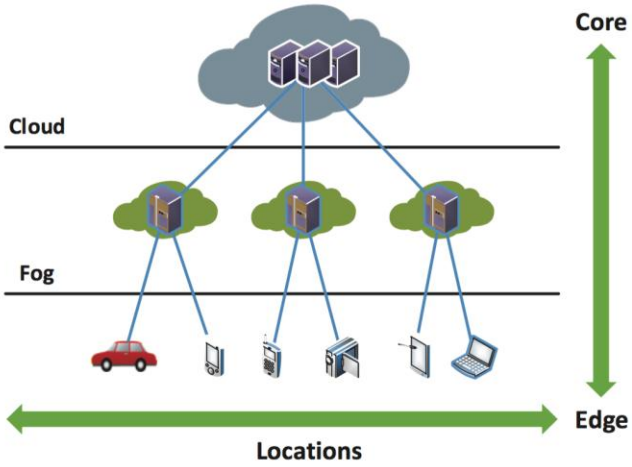
Fall 2016 - Programming Project

Initial Draft Description

Introduction

In this project you will implement functionality of a node in a distributed system in the context of *Fog Computing*. Fog computing is a newly-introduced concept that aims to put *the Cloud* closer to the user for better quality of service (QoS). Fog computing is studied mostly in the context of Internet of Things (IoT) as it provides low latency, location awareness, support for wide-spread geographical distribution, and mobility support for IoT applications. Fog is not a substitute for but a powerful complement to the cloud.

A very simplistic model of a fog is shown in the figure below. In this architecture, the cloud is in the core, IoT nodes are at the edge of the Internet, and the fog sits between the cloud and the IoT nodes closer to the IoT nodes (normally within 1 or 2 hop distance). Fog nodes collaborate among each other and provide relevant cloud services such as computing, networking and storage to IoT devices. In this project, the main use of fog computing is to reduce the response time delay for requests coming from the IoT nodes.



When a fog node receives a request, depending on its current work load, it will either handle the request by itself or will forward it on. A request can be forwarded among the fog nodes at most *Forward_Limit* times. If a request is forwarded *Forward_Limit* many times and it cannot be served by the last fog node (due to its current work load), the node will forward the request to the cloud for processing. The node processing the request (either a fog node or the cloud) will directly send its response back to the IoT node.

Protocol

According to the figure above, the system includes three types of nodes as (1) IoT nodes, (2) fog nodes, and (3) cloud node. In this project, we will provide you with an IoT node implementation that will be generating requests and sending them to the fog nodes. Your main task will be to implement the functionality of a fog node. You will also simulate the functionality of the cloud node as we will discuss below.

The operation of a fog node includes two tasks: (1) receiving requests and, depending on its response time for this request, processing or forwarding the request and (2) periodically exchanging information with its 1-hop neighbors about their current response time so that they know who to forward a request when needed.

When a fog node decides to process a request, it will place it to the end of the processing queue that it keeps pending requests in. The decision on processing or forwarding a request will be based on the expected *response time* for the request. The response time includes queuing delay and processing delay for the request. Each request will include a parameter that will define the processing time of the request. When a new request R_{new} arrives at a fog node A, using the processing time info in R_{new}, the fog node A can calculate the queuing time for R_{new}. The queuing time will be given by the summation of the processing times of all the currently queued requests at A. As a result, the expected response time of R_{new} can be calculated by adding up the queuing delay and the processing time of R_{new}.

When a request is up for processing at a fog node, the processing operation should be simulated by having a thread sleep or wait for processing time of the request. Once this time elapses, a response should be created and sent back to the request originator IoT node.

If a fog node decides that it is too busy to accept this new request (which will be defined by an input parameter of *Max_Response_Time* at each node), it will check if it can forward the request to a neighbor or not. This depends on the *Forward_Limit* and how many times this request has been forwarded 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. The best neighbor is the one that reported the smallest response time in the most recent round.

If a fog node cannot accept a new request and the *Forward_Limit* is reached for the request, the fog node will need to forward this request to the cloud. In this project, instead of forwarding the request to the cloud, you will simulate this operation in the following way: your fog node implementation will include a thread which will represent the cloud node. When a fog node needs to forward a request to the cloud, it will pass the request to the cloud thread which will incur the necessary processing delay (by using a sleep or wait system call for the duration of processing time recorded in the request message) and then will send the response directly to the request originator IoT node. Your simulation of the cloud node will not consider any queuing delay. This way, we will avoid implementing a stand alonoe cloud node in the project.

Selection of the best neighbor depends on the response time of the 1-hop neighbors. This information needs to be periodically exchanged among 1-hop neighbors so that they can use this up to date information to decide on their best neighbors to forward a request when needed. When the time comes to send a periodic response time information to 1-hop neighbors, a fog node will calculate the response time that will be needed for an incoming request at this very moment and will send it as its response time to its neighbors. Once each neighbor does this, everyone will be able to use this information to figure out their best neighbors.

In this project, we will use requests of 10 different types each with a different processing time. A request of type x incurs $2x$ processing time units (assumed to be in seconds). Request type will be decided by the request originator IoT node and will be included in the request message indicating the processing time needed for the request.

Given that a request may visit multiple nodes in the system, for debugging and testing purposes, you are required to come up with a scheme to carry necessary information on the message body. When the corresponding response arrives at the IoT node, the IoT node will print it out . The information should clearly indicate which fog nodes are visited and what each fog node did on the request. This information can be stored as a text string in the message body while it propagates in the system. Each fog node can append its own message to the end of the string. The node sending a response for this request, can copy over the text string into the response message and append its own information to it before sending it back to the requesting IoT node.

Implementation Details

For implementation, you are allowed to use C, C++, Java, or python. TCP/IP sockets must be used for communication between nodes. All numerical parameters and constants of the program must be defined as variables (no value should be hard coded).

We believe that the project involves implementing three protocol operations between the nodes as below:

1) Periodic response time update messages among the fog nodes
2) IoT node to fog node request/response communication
3) Fog to fog request offloading

The operations (1) and (3) above should be implemented over TCP sockets and the operation (2) should be implemented using UDP sockets.

Please note that this is the initial draft of the project and the final version will be developed with the input and help of the students. We will have a class session to discuss the project details and use this discussion to finalize the description. Hence, it is important that you read this draft description and think about the application and contribute to the development of the final description.

Once the project description is finalized, you will be asked to develop a design document and submit it to the TA. You will then have a meeting with the TA to go over your design. This meeting will help insure that our expectations and your understanding of the project work is inline so that we do not have any surprises at the end of the semester.