COMP3331/9331 Computer Networks and Applications
Assignment for Term 1, 2020

Individual Assignment

Due: 5pm, 21 April 2020

Implementation of Peer-to-Peer Network using Distributed Hash Table

Goal and Learning Objectives

The goal of this assignment is to gain hands-on experience in implementing the concept of peer-to-peer (P2P) networks. Instead of connecting geographically distributed peers over the Internet, *the scope of the assignment is confined to a single operating system with sockets or processes acting as independent peers*.

On completion of this assignment, students will:

1. Learn techniques for implementing peer-to-peer networks
2. Gain experience in implementing distributed hash tables (DHTs), the bedrocks for many distributed systems
3. Master socket programming

Background on P2P Networks and DHT

Week 3 lecture covers the necessary background on P2P networks. The lecture notes are supplemented by relevant pages from the 6th edition of the textbook as well as a scientific article, called CHORD, which provide more detailed coverage of DHT. You will implement the methods described in these documents.

Assignment Specifications

You will implement the following **five P2P services**:

1. **Data insertion:** An external entity can request any peer to store a new data record into the distributed database implemented as the DHT.
2. **Data retrieval:** An external entity can request any peer to retrieve a data record from the DHT.
3. **Peer joining:** A new peer can approach any of the existing peers to join the DHT.
4. **Peer departure (graceful):** A peer can gracefully leave the DHT by announcing its departure to other relevant peers before shutting down.
5. **Peer departure (abrupt):** Peers can depart abruptly, e.g., by "killing" a peer process using CTRL-C command.

For insertion, retrieval, or joining services, you are required to display the route taken along the circular DHT to complete the service.

Following guidelines apply for implementing these services:

- A peer is implemented as a *process* and represented as an *xterm*, which can be used as an interface to approach the peer. The xterm can also be used for the peer to display any internal messages.
- The DHT identity of a peer is drawn from the range [0,255]. This means that technically, the DHT can support a maximum of 256 peers to join the network. For tractability, however, actual testing will be conducted with a small number of peers.
- Data records are files stored in the same directory where the assignment codes are stored.
- Filenames are four-digit numbers such as 0000, 0159, 1890, etc. Filenames such as a912 and 32134 are invalid because the former contains a non-numeral character and the latter does not consist of exactly 4 numerals
- Hash function used to produce the key is given as *modulus(filename/256),* which results in a key space of [0,255] matching the DHT identity space. For example, the hash of file 2012 is 220.
- A file producing a hash of *n* is stored in the peer that is the closest successor of *n* in the *circular DHT.*

Additional Notes

You should use client-server socket programming paradigm to implement all communications between the peers. For example, you can choose specific unique port numbers for each peer to listen for communications, which should be known by all peers. You can use TCP sockets for reliable communications.

What to submit

You will submit your code with a readme file explaining how to run your code. You can choose from C, Java, or Python to implement your code. You must make sure that the submitted code compiles and run on CSE Linux servers, because the markers will test your code there. Your assignment cannot be marked if it does not work in CSE servers and will be treated as a failure.

Late submission penalty

Assignments submitted late will be penalised at the rate of 10% per day late. For example, a student receiving an original (without penalty) mark of 18/20 but submitting 2 days late, will receive a final mark (after penalty is applied) of 18x0.8 = 14.4.

Marking policy

If your code does not compile, the maximum mark you can obtain is 20%.

If your code compiles, the marks are distributed equally between the five services you are required to implement.

Plagiarism

You are to write all of the code for this assignment yourself. All source codes are subject to strict checks for plagiarism, via highly sophisticated plagiarism detection software. These

checks may include comparison with available code from the Internet and assignments from previous semesters. In addition, each submission will be checked against all other submissions of the current term. Do not post this assignment on forums where you can pay programmers to write code for you. We will be monitoring such forums. We are aware that a lot of learning takes place in student conversations, and do not wish to discourage those. However, it is important, for both those helping others and those being helped, not to provide/accept any programming language code in writing, as this is apt to be used exactly as is, and lead to plagiarism penalties for both the supplier and the copier of the codes. Write something on a piece of paper, by all means, but tear it up/take it away when the discussion is over. It is OK to borrow bits and pieces of code from sample socket code out on the Web and in books. You MUST however acknowledge the source of any borrowed code. This means providing a reference to a book or a URL when the code appears (as comments). Also indicate in your report the portions of your code that were borrowed. Explain any modifications you have made (if any) to the borrowed code. If detected, the penalty for plagiarism can be a ZERO mark for the assignment as well as further measures related to student conduct.