CS3102 Practical 1

A Reliable and Efficient File Distribution System

This practical is worth 50% of the overall coursework mark and 20% of the final module grade.

Overview

Design, implement and measure the performance of a system, which *reliably* and *efficiently* replicates whole files to the local persistent storage (SSD) of nodes in the SoCS lab network. "Reliability" means a bit-perfect transfer, *it does not imply secure communication*. "Efficiency" means that the cost of copying a file to 1 node is not significantly different from what can be achieved using `sftp`, and that distributing to 3 and 6 nodes should not have a worse than linear time increase, i.e. replicating a file to three nodes should not take more than three times longer than replicating to one node.

Systems must consist of your *original* code in Java and able to be compiled and executed with the `javac` and `java` commands from the Linux command line on SoCS computers in the JHB or JCB.

Design

File replication is a common requirement for distributed systems. It can be achieved by simply transferring one copy at a time to each remote node. Approaches that are more efficient, but also more complex, include parallel copying, peer-to-peer and multicast. Experimental sophisticated schemes such as FCast[1] exist. Compression can be useful but will not add value if a file is already in a compressed format e.g. PNG or video. Other design decisions include the choice/mix of transport protocols, and the design of any original application protocol(s). The Unix `rsync` program is efficient because it applies differential updates, but this coursework requires *whole file* copying. Network file systems such as NFS and CIFS typically work by serving parts of a file on demand, but this coursework requires the transfer of *whole files* from a source node's local persistent storage to each destination node's local persistent storage across the SoCS network

Test and Measurement

It is important to check which files systems are local and which are network types. Files placed in those locations may persist through a reboot, so a tidy-up script may be needed. It is also important for the meaningfulness of your measurements not to mix machines with different capabilities. (See the accompanying document on SoCS Lab Computer Information).

A set of three files are provided. Measurements of time to completion are required values for replicating each file to 1, 3 and 6 nodes. Graphs produced should show the averaged total replication time (Y axis) for each of these files plotted against the number of replicas (X axis). Make sure your results are reasonably robust by taking the average from several runs for each case. Watch out for outliers caused by destination nodes being busy with some other work.

[1] Gemmell, J., J. Gray, and E. Schooler, *Fcast multicast file distribution.* IEEE Network. Mag. 2000. **14**(1): p. 58-68.

The Report

Importantly, your report should include a design section, which explains what approaches you considered and/or tried and why you settled on the one you implemented.

You report must include graphs showing your system's completion times for distributing the three provided files to 1, 3 and 6 remote nodes. Regardless of the measured performance, your report should include a commentary explaining your results.

State clearly in the report how many runs you carried out, which nodes you used, and exactly how you took measurements. Include your results spreadsheet in your submission. Explain in your report how you ensured the reliability of the individual transfers and the final status of a replication run e.g. did all nodes receive a copy? Does the system keep going if there is a problem with a single node?

SoCS Lab Computer Information

A separate document gives information on the SoCS Lab computers.

Marking

The general mark descriptors from the SoCS student handbook apply (https://info.cs.st-andrews.ac.uk/student-handbook/learning-teaching/feedback.html). Note that it is possible to achieve a good grade without adding extra features. You should aim for quality of design, implementation, test and measurement, and the final report. Extra features can be added however (if time allows) and may attract extra credit. These could include: demonstrating scalability by carrying out measurements for larger numbers of nodes; adding fault tolerance or fault recovery e.g. from a receiving node crashing or becoming disconnected during a transfer; good automation e.g. the system finds the required number of free nodes on the network rather than working from a pre-allocated list, <your own extra feature>.

Hand in via MMS, by the specified due time:

- Your source files, executables, test files and measurements spreadsheet.
- A report in PDF format, maximum length 6 sides, minimum 11 point font, explaining your system design, the decisions you made, how you tested your system, how you made and presented your measurements and how you solved any difficulties that you encountered. Supplementary screenshots or diagrams may be placed in a separate directory for reference. *The report must contain clear instructions on how to compile and run your system from the Linux CLI.*

Lateness

The standard penalty for late submission applies:
https://info.cs.st-andrews.ac.uk/student-handbook/learning-teaching/assessment.html

Good Academic Practice

The University policy on Good Academic Practice applies:
https://info.cs.st-andrews.ac.uk/student-handbook/academic/gap.html