Sinh viên thực hiện:

Họ và tên	MSSV
Lê Nhật Quang	19120340
Chung Hoàng Tuấn Kiệt	19120553

Chuyên đề hệ thống phân tán

Đồ án 2: Cài đặt Schiper-Eggli-Sandoz Algorithm

BÁO CÁO ĐỒ ÁN

I. Ghi chú và cấu trúc chương trình

- Môi trường: GNU/Linux.
- Ngôn ngữ lập trình: C (POSIX Threads).
- Công cụ: gcc, make, bash.
- Cấu trúc project gồm các thư mục:
  - src: chứa mã nguồn chương trình và file biên dịch (Makefile)
  - test: chứa các tập tin config để chạy chương trình và file shell scripts dùng để tự động chạy nhiều process cùng lúc.

II. Biên dịch và chạy chương trình

1. *Config:*
   - Do chúng ta phải chạy ở 15 process khác nhau, mỗi process sẽ có listening port khác nhau do đó cần phải có 15 file config khác nhau. 15 file trên đã được tạo sẵn tương ứng với id và listening port của từng config. Với:
     + n: Số lượng process (n = 15).
     + id: Số thứ tự của process (id = 0 -> 14)
     + nmsg: Số lượng tin nhắn gửi đi (cho mỗi process khác, nmsg = 150)
     + rmin: Tần suất gửi tin tối thiểu (Đơn vị: message/phút). Mặc định: rmin = 80.
     + rmax: Tần suất gửi tin tối đa (Đơn vị: message/phút). Mặc định rmax = 100.
     + lport: Cổng lắng nghe của process.
     + log: Đường dẫn đến tập tin log của process (log = process_{id}_log.txt).
     + process:X:Y:Z - Thông tin truy cập của process X.

2. *Biên dịch:*
   - Trong thư mục src chạy lệnh *make* ở terminal. Thư mục obj sẽ được tạo ra chứa các file có extension là ***.o***. Các file object kết hợp với nhau sẽ tạo thành chương trình *ses-algo*. Có thể su763 dụng mode *delay=true* để thự nghiệm độ trễ của message khi test ở local.
     (VD: *make delay=true*)
   - Để clean các file ***.o*** và tệp chương trình ***ses-algo*** có thể gọi lệnh *make clean*.
3. *Thực thi:*

- Chạy chương trình bằng các shell scripts được viết sẵn: ***./run.sh ./ses-algo***

III. Thiết kế và cài đặt

1. *Quy trình thực thi:*

- Khi bắt đầu, chương trình đọc tập tin config để ghi vào giá trị các biến. Sau đó, chương trình khởi tạo vector clock và vector gói tin đã gửi với giá trị 0.
- Lúc này, chương trình khởi tạo n-1 threads để gửi tin nhắn đến các process

khác một cách đồng thời. Nếu macro _TEST_DELAY_MESSAGE_ được định nghĩa (biên dịch bằng make delay=true), thì với mỗi message muốn gửi, thread hiện hành sẽ phải tạo một thread mới, thread mới này delay ngẫu nhiên từ 0 đến 3 giây rồi mới thực sự gửi message đi.

- Cuối cùng, chương trình dùng file descriptor polling (thủ tục select()) để xử lý tuần tự các messages nhận vào và quyết định delivery hay buffer message bằng thuật toán SES.
- Khi một thread gửi đủ nmsg cho một process khác, nó sẽ gửi thêm một message có nội dung “FIN:SEND” đến process đó. Chương trình chỉ kết thúc khi đã nhận và delivery đủ n-1 message “FIN:SEND”.

2. Cài đặt:

- main.c: File main của chương trình, khởi tạo biến toàn cục (thông qua hàm *config_default()*) và quản lý việc gửi message.
- defs.h: Khai báo các biến toàn cục và các struct dùng trong chương trình.
- timevec.c và timevec.h: Kiểu dữ liệu biểu diễn vector clock và các hàm liên quan.
- sentvec.c và sentvec.h: Kiểu dữ liệu biểu diễn vector chứa thông tin các message đã gửi.
- config.c và config.h: Các hàm khởi taọ giá trị cho các thuộc tính (từ config file).
- logs.c và logs.h: Các hàm để write vào file log.
- threads.c và threads.h: Cài đặt các hàm để khởi tạo thread, gửi message giữa các process (có cài đặt mode delay để test độ trễ message)
- ses.c và ses.h: Cài đặt thuật toán SES theo yêu cầu đề bài.