Национальный исследовательский университет «Высшая школа экономики»
Факультет компьютерных наук
Программная инженерия

Операционные системы

Отчёт по индивидуальному домашнему заданию№4
Вариант 8

Работу выполнила:
М. В. Царахова
Группа: БПИ-213
Преподаватель:
А. И. Легалов

Москва
2023

Содержание

Постановка задачи

Сетевые взаимодействия с применением транспортного протокола UDP

Разработать клиент–серверное приложение, в котором сервер (или серверы) и клиенты независимо друг от друга отображают только ту информацию, которая поступает им во время обмена. То есть, отсутствует какой-либо общий вывод интегрированной информации, отображающий поведение системы в целом.

Военная задача. Анчуария и Тарантерия — два крохотных латиноамериканских государства, затерянных в южных Андах. Диктатор Анчуарии, дон Федерико, объявил войну диктатору Тарантерии, дону Эрнандо. У обоих диктаторов очень мало солдат, но очень много снарядов для минометов, привезенных с последней американской гуманитарной помощью. Поэтому армии обеих сторон просто обстреливают наугад территорию противника, надеясь поразить минометы противника. Стрельба ведется хаотично до полного уничтожения всех минометов противника, размещенных на некоторой прямоугольной площадке размером N*N. То есть координаты целей, хоть и случайные, согласуются между разными минометчиками одной армии (это предохраняет от стрельбы в одну и ту же точку). Интервалы между выстрелами (задержки) зависят от расстояния до точки попадания, формируемой случайно. Повторная стрельба по одним и тем же координатам не производится.
Количество минометов у каждой стороны одинаково и задается аргументом командной строки. Размер поля также задается в командной строке. Создать приложение, моделирующее военные действия.

Каждая страна — отдельный клиент. Сервер отвечает за прием координат от клиентов и передает эти координаты другим клиентам. Он также получает информацию от клиентов об их уничтожении. Расположение минометов порождается каждым клиентом случайно.

Программы должны быть написаны на языке программирования C и выполняться в среде ОС Linux

4-5 баллов

В файле helper.h расположены дополнительные функции для сервера и клиента, как например создание сокета или слушание сервера. В файле war.h расположены дополнительные функции для логики клиента.

В данной программе реализованы серверная и клиентская часть. Серверная часть отвечает за прием сообщения от одного клиента и отправляет его другому клиенту до тех пор пока один из клиентов не будет уничтожен.

Клиентская часть принимает 4 аргумента - ip, port, размер поля и количество минометов. С помощью функции createSocket определяется сокет, затем устанавливается адрес сервера с помощью функции setServerAddressWithIP. Я буду хранить в каждом клиенте информацию о собственном поле - где устанавливаются минометы, а также информацию о поле противника, чтобы сохранять точки в которые уже была стрельба. А затем начинается отправка и получение сообщений с сервером. Сначала клиент получает от сервера в точки, в которые была произведена стрельба, устанавливает у себя эти координаты, а затем отправляет серверу свою стрельбу. Стрельба генерируется с помощью дополнительной функции, которая сохраняет массив координат. При этом до отправки стрельбы выполняется проверка на окончание. Получение данных с помощью функции recv и отправка с помощью send. По завершении вся память очищается и сокет закрывается.

Для того чтобы понимать в каком состоянии сейчас война есть функция check_status, которая проверяет что у каждой стороны остался хотя бы один действующий gun.

Серверная часть имеет в себе два процесса, для того чтобы обрабатывать два клиента. Сначала создается сокет, затем устанавливается адрес сервера как и в клиенте. Также сохраняется клиент который будет взаимодействовать с данным процессом сервера, в переменных clientSocket и clientAddress. Мы вызываем функцию listen server чтобы указать что теперь сервер готов получать данные. Затем уже привычным способом инициализируем разделяемую память, в которой мы будем хранить координаты стрельбы, т.к. необходимо получать данные от одного клиента и отправлять другому. В программе будут использоваться два семафора, которые будут помогать устанавливать связь между процессами. Для каждого процесса устанавливается на ожидание соответствующий семафор, а когда данные получены, противоположный семафор делает post. Это необходимо чтобы в другом процессе семафор начал работу и отправку сообщения клиенту.

Компилирование сервера : gcc helper.c server.c -o server

Компилирование клиента : gcc helper.c client.c -o client

Пример запуска сервера: ./server 5000

Пример запуска клиента: ./client 127.0.0.1 5000 5 2 (необходимо запустить два клиента для корректной работы)