Тестовое задание: балансировщик tcp/ip соединений

Реализовать асинхронный (epoll io) tcp/ip сервер для проксирования клиентских соединений на разные сервера в зависимости от ID клиента. ID клиента содержится в первом сообщении, которое отправляется клиентом сразу после установления соединения с сервером.

Протокол клиент-серверного взаимодействия

Типы сообщений:

1. Инициализационное сообщение - хранит в себе ID клиента
2. Сообщение с передаваемыми данными от клиента

Последовательность передаваемых сообщений (msg#<msg type>):
msg#1 | msg#2 | msg#2 …. | msg#2

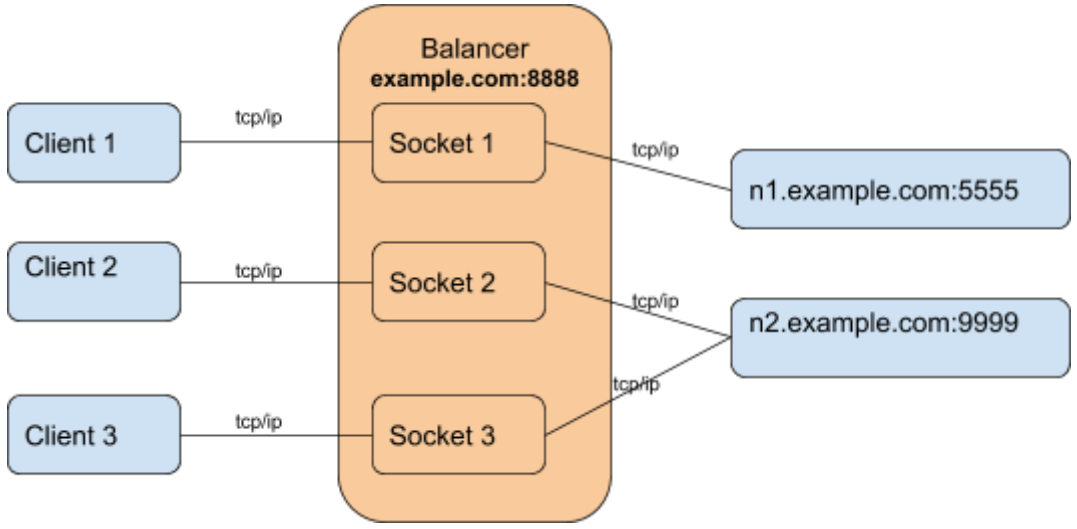
Формат протокола передачи данных выбирается разработчиком самостоятельно(выбранный формат должен быть основан на **TLV).**
Приветствуется реализация WebSocket протокола.

Входные данные для балансировщика

1) Файл с таблицей маршрутизации:

ID Клиента	Хост	Порт
1	n1.example.com	7777
2	n2.example.com	9999
3	n2.example.com	9999

Схема tcp/ip соединений для приведенной таблицы маршрутизации:



Тестирование и отладка

Для проведения тестирования и проверки результата работы балансировщика предлагается разработать(взять готовое) клиентское и серверное приложение с поддержкой протокола взаимодействия, реализованном в рамках решения данной задачи.

Блок-схема работа клиента (пример для клиента#1):

Сценарий проведения тестирования:

1) развернуть приемочные tcp/ip сервера на портах 0.0.0.0:5555 и 0.0.0.0:9999.
2) в /etc/hosts добавить запись 127.0.0.1 n1.example.com
3) в /etc/hosts добавить запись 127.0.0.1 n2.example.com
4) в /etc/hosts добавить запись 127.0.0.1 example.com
5) запустить балансировщик на 0.0.0.0:8888
5) запустить клиентские приложения Client1, Client2, Client3

Рекомендуемый инструментарий для решения задачи

Стандарт языка: C++ 11 и выше
Вспомогательные библиотеки: STL, log4cplus
Сетевой стек: libevent
Сборка и развертывание: cmake + docker
Тесты: google test