

Модель TCP\IP

Чаще всего балансировка уровня Сетевого(протокол скорее всего ip), транспортного(протокол скорее всего tcp) ,прикладного(как правильно на протоколе http) уровней

Сетевой уровень:

При запросе от клиента ,балансировщик определяет на какую машину он должен послать запрос(например какой сервер ближе к клиенту), сам балансировщик нечего не делает

Транспортный уровень:

В этом случае балансировщик общается с клиентом, он получает запрос, обогащает его дополнительными данными и далее он общается с конкретным сервером(например какой сервер менее загружен),

Прикладной уровень:

Балансировщик служит не просто перенаправлением трафика, он думает ,какой контент запрашивает клиент , и исходя из нахождения контента , он пойдет делать запрос в конкретное место ,и уже распределит нагрузку

Цели балансировки:

1.Справедливость , распределяем нагрузку для всех пользователей одинаково

2.Эффективность, чтобы сервера были заняты максимально продуктивно

3. Сокращение времени запроса(как вариант отработать на сетевом уровне ,чтобы сервер был ближе к клиенту)  
4.Сокращение времени отклика(отработать на прикладном уровне, распределяя нагрузку при запросе по контенту и точечному поиску по бд и серверам)

Алгоритмы:

1.Round Robin – по сути круговое обслуживание ,то есть поочередно 1й,сервер,далее 2й и 3й,на сетевом уровне ,к примеру по паре dns и ip адресу, после того ,как он пройдет круг начнет с начала

Плюсы его в том, что мы не зависим от высоких уровней, это быстро ,но он не предполагает отслеживания нагрузки на сервер , что плохо может повлиять на скорость обработки большого потока ,так же сервера не нужно связывать между собой, но с другой стороны сложнее отслеживать эффективность и справедливость

2.Weighted Round Robin – улучшенная версия Round Robin, добавлена возможность подсчета весового коэфицента, так как сервера с разной производительностью ,те у которых производительность больше, будут получать больше запросов, тем самым решая проблему с эффективностью

3.Least Connections – считает не количество запросов, а открытых коннектов и распределяет нагрузку относительно этого

4. Weighted Least Connections – улучшенный Least Connections ,мы добавляем расчет коэфицента

5.Destination Hash Scheduling \ Sourse Hash Scheduling – есть список ip адресов(dest) ,и запросы идут на конкретные ip серверов, информация хешируется и получается более быстрый доступ\ тут мы получаем ip именно отправителя информации и в соответствии с ним распределяем нагрузку в ровном порядке

6.Sticky Session – 1 клиент = 1 сервер,сам клиент закреплен за сервером