ИУ7-54Б, 16_KOZ, Рунов Константин

СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ

Ниже представлены термины в контексте данной работы и их определения.

Клиент — это программное или аппаратное обеспечение, которое инициирует запросы к серверу для получения определенных услуг, ресурсов или информации.

Сервер — это программное или аппаратное обеспечение, которое предоставляет услуги, ресурсы или информацию клиентам. Сервер обрабатывает запросы, поступающие от клиентов, и предоставляет им необходимую функциональность или данные.

Нагрузка — это объем данных, поступающий на сервер в определенный период времени.

Запрос — это данные, отправляемые клиентом серверу, с целью получения определенных услуг, ресурсов или информации. Запрос содержит необходимую информацию для предоставления сервером функциональности или данных, запрашиваемых клиентом. Запросы составляют нагрузку.

Группа серверов — это совокупность двух или более серверов, объединенных для совместной работы с целью обеспечения более высокой производительности, отказоустойчивости или распределения нагрузки.

ВВЕДЕНИЕ

Одной из задач, с которыми сталкиваются интернет-компании, является обеспечение бесперебойного доступа клиентов к предоставляемым компаниями интернет-ресурсам и, следовательно, к серверам компании. Сервера же могут выходить из строя по многим причинам: аппаратные сбои, проблемы с электропитанием, человеческие ошибки, перегрузка; в связи с чем, у компаний возникает естесственная потребность — минимизировать вероятность выхода из строя своих серверов. От аппаратных сбоев и человеческого фактора защититься бывает крайне сложно, но, если грамотно распределять между серверами поступающий объём данных, можно уменьшить вероятность выхода из строя серверов в следствие перегрузки.

Этим и занимаются так называемые «балансировщики нагрузки». Балансировщик нагрузки выполняет роль «регулировщика», стоящего перед серверами и направляющего запросы клиентов на все серверы, способные выполнить эти запросы таким образом, чтобы максимально увеличить скорость и загрузку мощностей и не допустить перегрузки одного сервера, что может привести к снижению производительности. Если один из серверов выходит из строя, балансировщик нагрузки перенаправляет трафик на оставшиеся работающие серверы. При добавлении нового сервера в группу серверов балансировщик нагрузки автоматически начинает направлять на него запросы. [1]

Сервер, на который следует направить клиентский запрос, балансировщики нагрузки выбирает в соответствии с различными алгоритмами и методами балансировки нагрузки.

Целью данной работы является анализ и классификация методов балансировки нагрузки высоконагруженных систем.

Для достижения поставленной цели необходимо решить следующие задачи:

— рассмотреть основные подходы к балансировке нагрузки;

— описать методы балансировки нагрузки, относящиеся к одному из подходов;

— предложить и обосновать критерии оценки качества описанных методов;

— сравнить методы по предложенным критериям оценки;

— выделить методы, показывающие лучшие результаты по одному или нескольким критериям.

1 Анализ предметной области

TODO

1.1 Актуальность проблемы

TODO

1.2 Области применения балансировки нагрузки

TODO

2 Описание существующих решений

TODO

2.1 Статическая балансировка

TODO

2.2 Динамическая балансировка

TODO

2.2.1 Методы на основе хеширования

Методы балансировки нагрузки на основе хеширования работают по общему принципу:

1) Из пришедшего запроса выбрать информацию (например, IP-адрес или URL-адрес), которая считается ключом хеш-функции в рамках данного алгоритма.

2) На основе ключа вычислить значение хеш-функции, которое соответствует идентификатору узла, на который следует перенаправить запрос для его обработки.

3) Перенаправить запрос на узел, чей идентификатор был вычислен ранее.

Хеширование на основе IP-адреса

Алгоритм балансировки нагрузки «Хеширование на основе IP-адреса» работает по общему принципу методов балансировки нагрузки на основе хеширования. Ключом хеш-функции в данном алгоритме считается IP-адрес источника запроса [1, 2, 3]. Запросы, имеющие один и тот же IP-адрес, будут обслужены одним и тем же узлом. То есть, если имеются запросы r_1, r_2, узлы l_1, l_2, моменты времени t_1, t_2 и функция расписания s, то, в соответствии с данным алгоритмом, будет выполнено следующее:

$$(\forall t_1, t_2)\left(\begin{cases} s(l_1, t_1) = r_1, \\ s(l_2, t_2) = r_2, \\ r_1.ip\_address = r_2.ip\_address. \end{cases} \Rightarrow l_1 = l_2\right) \quad (1)$$

Особенности алгоритма «Хеширование на основе IP-адреса»:

— алгоритм гарантирует, что все запросы от одного и того же пользователя направляются на тот же сервер;

— алгоритм предсказуем;

— добавление новых серверов потребует лишь изменения хеш-функции для корректной работы алгоритма;

— неравномерная нагрузка на узел, если запросы начинают приходить из сети, использующей NAT, с большим количеством пользователей.

Хеширование на основе URL-адреса

Алгоритм балансировки нагрузки «Хеширование на основе URL-адреса» работает по общему принципу методов балансировки нагрузки на основе хеширования. Ключом хеш-функции в данном алгоритме считается URL-адрес, к которому обращается источник запроса [1, 3, 4]. Запросы к одному и тому же URL-адресу, будут обслужены одним и тем же узлом. То есть, если имеются запросы r_1, r_2, узлы l_1, l_2, моменты времени t_1, t_2 и функция расписания

s, то, в соответствии с данным алгоритмом, будет выполнено следующее:

$$(\forall t_1, t_2)\left(\begin{cases} s(l_1, t_1) = r_1, \\ s(l_2, t_2) = r_2, \\ r_1.url\_address = r_2.url\_address. \end{cases} \Rightarrow l_1 = l_2\right) \quad (2)$$

Особенности алгоритма «Хеширование на основе URL-адреса»:

— алгоритм гарантирует, что все запросы к одному и тому же URL направляются на тот же сервер;

— алгоритм предсказуем;

— изменение структуры URL может потребовать перенастройки балансировщика;

— популярные URL могут создавать неравномерную нагрузку на узлы.

2.2.2 Метод фиксированных весов

В методе фиксированных весов, администратор назначает каждому узлу вес, после чего все запросы будут приходить на узел с максимальным весом [3]. Если узел перестаёт справляться с нагрузкой, запросы начинают перенаправляться на узел, с весом меньше.

Шаги инициализации алгоритма:

1) N = количество узлов

2) nodes = массив узлов

3) weights = массив весов таких, что weights[i] — вес узла i, назначенный администратором

4) i = 1

5) Пока i <= N:

   — nodes[i].weight = weights[i]

   — i = i + 1

Шаги работы алгоритма:

1) w = max(weights)

2) request_sent_flag = 0

3) Пока request_sent_flag = 0 и w > 0:

   — node = узел с весом w

   — Если узел node работоспособен, перенаправить запрос узлу node, request_sent_flag = 1

   — Иначе, w = w - 1

Таким образом, если имеется запрос r, узел l, момент времени t, функция расписания s, и выполняется равенство $s(l, t) = r$ то, l — узел с наибольшим весом, доступный в момент времени t.

Особенности метода фиксированных весов:

— алгоритм гарантирует, что все запросы будут направляться на доступный в текущий момент времени узел с максимальным весом;

— алгоритм предсказуем;

— веса узлов назначаются вручную;

— вес узла не меняется в процессе работы.

3 Классификация существующих решений

TODO

3.1 Иерархия методов

TODO

3.2 Оценка и сравнение

TODO

3.3 Вывод

TODO

ЗАКЛЮЧЕНИЕ

В итоге, в ходе данной работы:

— рассмотрены основные подходы к балансировке нагрузки;

— описаны методы балансировки нагрузки;

— предложены и обоснованы критерии оценки качества, по которым проведено сравнение описанных методов;

— выделены методы, показывающие лучшие результаты по одному или нескольким критериям.

Таким образом, все поставленные задачи были выполнены, а цель работы — достигнута.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. What Is Load Balancing? [Электронный ресурс]. — Режим доступа: https://www.nginx.com/resources/glossary/load-balancing (дата обращения: 12.11.2023).

2. Amazon Web Services [Электронный ресурс]. — Режим доступа: https://aws.amazon.com/what-is/load-balancing (дата обращения: 12.11.2023).

3. Load Balancing Algorithms and Techniques. [Электронный ресурс]. — Режим доступа: https://kemptechnologies.com/load-balancer/load-balancing-algorithms-techniques (дата обращения: 04.01.2024).

4. Ramirez N. Load Balancing with HAProxy: Open-Source Technology for Better Scalability, Redundancy and Availability in Your IT Infrastructure. – Nick Ramirez, 2016 – 172 с.