Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«Московский физико-технический институт
(государственный университет)»

Факультет радиотехники и кибернетики

Кафедра теоретической и прикладной информатики

Глобальная балансировка нагрузки серверов

Выпускная квалификационная работа
(магистерская работа)

Направление подготовки: Прикладные математика и физика

Выполнил:
студент 113 группы _____ Павловский Андрей Владимирович

Научный руководитель:
к.ф.-м.н., _____ Кудрин Максим Юрьевич

Москва 2017

Содержание

Введение

Масштабируемость, высокая доступность и производительность имеют основополагающее значение для успеха в развертывании большого коммерческого продукта. Многие компании пытаются наращивать мощности путем увеличения количества серверов и расширения инфраструктуры в одном месте, в то время как этот подход имеет некоторые сопутствующие ограничения. Централизованное представление веб сервисов представляет собой единую точку отказа приложений. При потере соединения с глобальной сетью участок становится недоступным для пользователей и клиентов, что может существенно повлиять на их бизнес. Качество предоставляемых сервисом услуг очень чувствительно к нарушениям соединения с сетью в непосредственной близости от участка. Более того, пользователи могут иметь существенные задержки в доступе по мере увеличения их географического расстояния от места дислокации предоставляемого сервиса, которые также усугубляются большим количеством обрабатываемых сервисом запросов в том числе и других пользователей, необходимых для передачи данных. Централизованная архитектура построения сервисов также не является подходящей для интернациональных компаний, предоставляющих свои услуги пользователям в различных частях земного шара.

Глобальная балансировка нагрузки серверов позволяет преодолевать эти проблемы методом перенаправления запросов через набор сторонних серверов и точек доступа в зависимости от географического положения. Пользователи автоматически перенаправляются к ближайшему или наименее загруженному участку в момент запроса, сводя к минимуму вероятность длительных задержек или сбоев в обслуживании. Кроме того, исследования показали, что быстрый и

надежный доступ к контенту и приложениям имеет решающее значение для онлайн-бизнеса и позволяет скорее добиться успеха компании в силу того, что конечные пользователи, как известно, нетерпеливы, и отказ ответить в течение семи секунд может вынудить, по меньшей мере 30 процентов пользователей, отказаться от приложения или сервиса.

В настоящее время существует достаточно большое количество GSLB решений различных компаний, большинство из которых предоставляет свой функционал по устранению проблемы балансировки и имеют свой набор достоинств и недостатков. Целью данной работы является описание и создание “идеального” балансировщика методом обобщения существующих решений с добавлением новых возможностей и устранением существующих недостатков.

1. Global Server Load Balancing

В середине 90-х годов на веб-сайтах стали наблюдаться возросшие объемы трафика, отдельные серверы достигали пределов своих вычислительных способностей, чтобы справиться с возросшей нагрузкой. Для расширения и увеличения количества приложений и внедрения новых технологий требовались дополнительные ресурсы, чтобы конечным пользователям казалось, что они обращаются к одному серверу. Первым способом решения этой проблемы масштабируемости было DNS (Domain Name System), также называемое «циклическим DNS». Этот метод присваивает группе уникальных внутренних IP-адресов серверам, расположенным за брандмауэром, одно имя DNS. Когда пользователь запрашивает разрешение имени веб-сайта, DNS будет отвечать с несколькими адресами в порядке, например 10.1.0.10, 10.1.0.11 и 10.1.0.12. Следующий запрос, сделанный в DNS, будет содержать одни и те же адреса, однако они будут повернуты так, чтобы второй сервер был первым (10.1.0.11, 10.1.0.12 и 10.1.0.10). DNS будет продолжать перестраиваться через серверы для каждого последовательного ответа.

Round-Robin DNS было простым решением, которое решило проблему масштабируемости, предложив почти неограниченное количество серверов для добавления в DNS. Однако без возможности узнать статус сервера на принимающей стороне запроса пользователи могут быть перенаправлены на сервер, который в данный момент может быть выключен или перегружен. Вскоре стали доступны многие программные подходы к балансировке нагрузки для решения проблемы доступности сервера, как правило, в составе операционной системы или прикладного программного обеспечения. Эти системы создавали серверные кластеры, которые постоянно контактировали друг с другом и обменивались информацией о состоянии сервера, соединениях

и других данных для обеспечения форм проверки работоспособности сервера. Запросы на подключение будут переданы первому доступному серверу, чтобы затем быть направлены к наиболее доступному, либо основному, либо альтернативному серверу в кластере. Данный подход хорошо проявил себя при работе с небольшими приложениями с менее чем 10 серверами. Большие приложения видели резкое снижение производительности с каждым новым добавленным сервером из-за постоянной потребности в поддерживании контакта друг с другом. Эта ограниченная емкость в сочетании с проприетарным программным обеспечением привела к необходимости использования нового решения, которое могло бы надежно масштабировать и поддерживать большие приложения или множество приложений.

1.1 The Hardware-based Load Balancer

Начиная с конца 1990-х годов, производители представили первые аппаратные средства балансировки нагрузки, именуемые Hardware-based Load Balancer. Отделяя распределение нагрузки от самих приложений, устройства могут полагаться на использование методов сетевого уровня, таких как трансляция сетевых адресов (Network Address Translation) для маршрутизации входящего и исходящего трафика на серверы. Другим ключевым компонентом, который был представлен, была проверка работоспособности сервера. Через заданные промежутки времени балансировщик нагрузки проверял состояние сервера, чтобы определить, был ли он доступен и какова была его загрузка. Если сервер не работает, трафик будет направляться на альтернативные операционные серверы. Перенаправление трафика происходит до до тех пор, пока загрузка на базовый сервер не станет ниже установленных порогов свободных мощностей, необходимых для обеспечения дальнейшей работы. В этом случае приложения могут масштабироваться и пользователи будут иметь

надежные соединения. Единственным ограничивающим фактором была способность самого оборудования. В большинстве случаев организации, которые мигрировали с использованием балансировки нагрузки на основе DNS или программного обеспечения, увеличили в среднем на 25% производительность сервера, уменьшив необходимость добавления новых ресурсов для увеличения мощностей.

1.2 Global Server Load Balancer

Global Server Load Balancer представляет собой набор технологий, используемых для развития бизнеса и устранения неисправностей, возникающих в результате работы сервисов или приложений. В зависимости от природы развернутой инфраструктуры преследуются различные цели, такие как:

- Высокая доступность и отказоустойчивость: обеспечение альтернативно расположенного средства доступа к ресурсам в случае неполадок.
- Распределение нагрузки: распределение запросов между множеством серверов с целью ускорить выполнение той или иной задачи.
- Производительность: выбор локации в зависимости от географического расположения пользователя и близости сети или по принципу наименьшей загруженности с целью снижения задержек при доступе к ресурсу.

Для обеспечения выполнения поставленных задач, GSLB полагается на такие технологии как DNS redirection, HTTP redirection, Route Health Injection (RHI) и другие. Обычно GSLB настраивается в режим активного центра обработки данных или как резервный. В первом случае GSLB напрямую обрабатывает клиентские запросы и балансирует нагрузку между центрами

обработки данных на данном участке. Во втором случае данный GSLB “спит” до возникновения необходимости в его работе и начинает работать как активный центр, если общая производительность сервиса вырастет при его участии. Функциональность GSLB и балансировку нагрузки обеспечивают контроллеры доставки приложений (ADC).

1.3 The Application Delivery Controller

Контроллер доставки приложений - это устройство, которое обычно размещается в центре обработки данных между брандмауэром одним или несколькими серверами приложений в области, известной как Demilitarized Zone. Контроллеры доставки приложений первого поколения, в основном, выполняли ускорение приложений и балансировку нагрузки между серверами. Современные контроллеры доставки приложений выполняют гораздо более широкий спектр задач, включая контроль загрузки сетевых каналов и поддерживают SSL, а также служат брандмауэром веб-приложений.

ADC устройства обрабатывают трафик приложений и оптимизируют производительность сервера приложений путем разгрузки многих вычислительных процессов, которые в противном случае перегружают процессоры, замедляя работу задач, выполняющихся параллельно.

1.4 Дополнительные возможности ADC

Среди современных функций ускорения, присутствующих в современных процессорах - технология разгрузки SSL, сжатие данных, оптимизация протокола TCP и HTTP и виртуализация. Разгружая и ускоряя шифрование SSL, дешифрование и управление сертификатами с серверов, ADC позволяют веб-серверам и серверам приложений использовать свои ресурсы процессора и

памяти исключительно для доставки контента приложений и, таким образом, более оперативно реагировать на запросы пользователей. Веб-приложения состоят из множества различных объектов данных, которые могут доставляться серверами разных типов. ADC предоставляют основанную на приложениях маршрутизацию с использованием типов файлов, чтобы перенаправлять пользователей на сервер или группу серверов, настроенных для обработки конкретных информационных запросов, таких как приложения ASP или PHP. Запросы пользователей могут направляться на разные серверы путем отправки запросов на статические типы файлов (jpg, html и т.д.).

Приложения на основе транзакций требуют подключения к одному серверу для правильной работы. Наиболее известным примером этого является проблема «корзины покупок», когда Вы устанавливаете сеанс с одним сервером, чтобы добавить элемент в свою корзину, а затем балансируете нагрузку на другой сервер для проверки. Если у вас нет постоянной связи с исходным сервером, вы увидите, что ваша корзина пуста. ADC используют состояние сеанса с заголовками HTTP и cookies, чтобы гарантировать, что пользователи и серверы остаются «постоянными». Без этой возможности, если пользователь перейдет на другой сервер, предыдущая история транзакций будет потеряна, и пользователю потребуется начать новую транзакцию.

Балансировка глобальной балансировки серверов для ADC решает сложную задачу масштабирования приложений в нескольких центрах обработки данных для аварийного восстановления или для снижения времени отклика приложений для географически рассредоточенных пользователей. Используя подход, основанный на DNS, в сочетании с настраиваемыми бизнес-правилами, пользовательские запросы разрешаются в ближайших, наиболее эффективных центрах обработки данных. Если центр обработки

данных недоступен из-за стихийного бедствия или запланированного обслуживания, пользователи автоматически маршрутизируются в другой центр обработки данных, пока первичный центр обработки данных снова не будет подключен к сети.

Link Load Balancing управляет несколькими широкополосными каналами (WAN) в Интернете из ADC, чтобы улучшить время отклика приложений, сократить потребности в пропускной способности и обеспечить отказоустойчивость в случае сбоя канала. Если подключение к Интернету становится переполненным или недоступным, трафик автоматически перенаправляется на альтернативные адреса. Наконец, современные ADC должны уметь работать с виртуальными средами, а также иметь возможность управлять ими. Расширенные ADC предлагают глубокое управление ресурсами виртуальных сред, а не просто базовую проверку работоспособности серверов. С помощью этой жесткой виртуальной интеграции ADC может принимать решения о балансировке нагрузки на основе состояния виртуальных машин и серверов, на которых они выполняются.

2. Существующие проблемы

Существует множество проблем связанных с созданием правильно построенной архитектуры облачной балансировки. Некоторые из этих проблем являются результатом неполноценности существующих облачных решений, основанных на требованиях рынка и опыте. Другие проблемы, однако, требуют установления стандартов, прежде чем они будут в достаточной степени учтены.

2.1 Развивающийся рынок

Одной из первых проблем для организаций является поиск поставщика облачных вычислений, удовлетворяющего поставленным требованиям. Прозрачность в сфере услуг провайдера все еще находится в зачаточном состоянии, поэтому поиск подходящего предложения может занять достаточно много времени. В условиях динамичного рынка процесс поиска становится все более сложнее. По мере того как инфраструктура провайдеров продолжает развиваться, и сами провайдеры реагируют на требования клиентов и рынка, предложения неизбежно изменятся. К тому же, внутренние облачные вычисления набрали обороты в качестве одного из жизнеспособных вариантов (имеющим место быть), но сравнение публичных облачных провайдеров с частными облаками не всегда является простой задачей, поскольку затраты представлены в разных масштабах. Покупка сервера для повышения внутренней мощности облака является одноразовым мероприятием, но добавление мощности в публичном облаке включает в себя всего несколько ежемесячных платежей.

2.2 Портируемость приложений

Отсутствие стандартов облачных провайдеров в отношении миграции, развертывания приложений и доставки мета-структуры, которая должна сопровождать миграцией, делает процесс переносимости приложений гораздо сложнее, если не невозможным во многих ситуациях. Еще более усложняют переносимость наличие в конечном итоге требования для решений межоблачной и облачной балансировки - отсутствие совместимости на уровне приложений. Хотя виртуализация является основным механизмом, посредством которого приложения развертываются в практически во всех средах облачных вычислений, технологии виртуализации могут варьироваться в зависимости от собственных коммерчески доступных платформ. Собственные платформы могут существенно осложнить реализацию решения балансировки в облаке, которое включает в себя развертывание локальных центров обработки данных. Коммерчески доступные платформы могут обеспечить более простые реализации, если платформы виртуализации являются однородными, но гетерогенная среда виртуализации может оказаться столь же стимулирующим фактором как и собственная платформа. Поэтому переносимость приложений методом облачных сервисов должна происходить на контейнерном слое, с помощью virtualization-agnostic environments, позволяющей полностью перемещать контейнеры через облачных сервисы. Это портируемость может быть достигнута за счет сочетания API, и принятие единой модели дескриптора виртуального данных, таких как Open Virtualization Format (OVF). Также был достигнут значительный прогресс в области виртуализации приложений, что позволяет осуществлять портирование данных между серверами, которые разделяют общую операционную систему. Это еще одна часть головоломки, которая в конечном счете приведет к полной автоматизации сети доставки приложений. Когда существующая сборка может быть скопирована или

перенесена на альтернативную инфраструктуру, только отсутствие возможности запускать и останавливать эту инфраструктуру по своему желанию, в зависимости от спроса, мешает полной автоматизации балансировки.

2.3 Интеграция

Наибольшая эффективность облачной балансировки достигается при условии хорошей интеграции решений глобальной и локальной служб доставки приложений. Облачная балансировка зависит от множества факторов, которые должны быть известны в локальной среде, т.е. глобальные и локальные решения должны быть в состоянии разделить эту информацию. Принятие стратегии одного поставщика для решения этой проблемы является решением, но удовлетворяющим немногие сторонние организации в отношении обоих (и организации и провайдера) из-за нежелания полагаться на одного поставщика для обслуживания и потому, что ослабляет их позиции на переговорах за столом лицензирования. В то же время, нет никакой гарантии, что каждое облако будет делиться решения одних и тех же производителей. Таким образом, реализация стратегии облачной балансировки требует динамической среды, а также поставщика нейтрального и приемлемого для всех решения. Пока поставщик нейтральных решений не предоставлен, организации должны использовать существующие API-интерфейсы компонентов для достижения интеграции переменных обычно не связанных с измерениями сетевого уровня, таких как стоимость за одну транзакцию для внутренних и внешних облаков. Эти переменные могут быть вычислены через регулярные промежутки времени и затем предоставлены внешнему контроллеру доставки приложений через его API для поддержки основополагающих данных по принятию решений в актуальном состоянии.

2.4 Архитектурная совместимость

Тесно связанной с проблемой интеграции глобальных и локальных решений по доставке приложений является архитектурная совместимость. Наличие стандартизации доставки приложений облегчает вопросы, связанные с оперативными различиями работы решений и облачных сред. Эти вопросы включают в себя увеличение расходов и времени развертывания инфраструктуры в то время как операторы и администраторы знакомятся с различными решениями.

В то время как виртуальные девайсы могут решить некоторые из проблем, возникающих в связи с архитектурной несовместимостью, они не обеспечивают полное решение, потому что некоторые модели облачной инфраструктуры не основаны на технологиях виртуализации и коммерческой являются частной собственностью по своей природе. Это привносит организации некоторые трудности в воспроизведении архитектуры через облака и поддерживать архитектурную совместимость через развертывание облачных сервисов.

Одним из путей достижения архитектурного сходства является virtual Application Delivery Controller (vADC). ADC обеспечивает локальный балансировщик нагрузки, необходимой для реализации архитектуры с поддержкой облачной балансировки, но в то же время нет никаких гарантий, что облачные провайдеры будут иметь доступные необходимые для балансировки нагрузки решения для клиентов. Развертывание vADC с приложением в облачной среде гарантирует что организация располагает средствами для мониторинга и управления состоянием этой облачной развертки приложений. VADC также предусматривает архитектурную неоднородность, требуемую глобальным контроллером доставки приложений, чтобы включить мириады (myriad) и переменные, используемые в облаке балансировки для принятия решения о глобальной маршрутизации приложений.

VADC также может служить платформой для глобальной балансировки нагрузки и маршрутизации DNS позволяющей всем облачным реализациям (внутренним и внешним) работать в унисон, словно они находятся в одной сети, которая предоставляет выполнение запрашиваемой услуги в зависимости от наиболее благоприятного географического расположения, а именно близости, затрат на предоставляемые мощности и другие факторы, заранее определенные организацией. С vADCs в архитектуре облака и физическим ADC в основном центре обработки данных, скоординированный ответ на изменения в сети или условий в которых находится приложение могут быть отосланы автоматически. Если vADC вдруг перестает отвечать на запросы, то GSLB и глобальные системы DNS на первичном ADC может прекратить отправку запросов к этому провайдеру и оповестить операторов задачи.

2.5 Безопасность и доступность

Никто не может контролировать то, к чему он не имеет доступа. Представляя vADC с GSLB в облачной среде обеспечивает контроль ADC и удобство развертывания облака.Security от распределенного отказа в обслуживании (DDoS) защиты от DNSSEC требуется определенный уровень контроля, который не предлагается большинством облачных провайдеров в настоящее время. Использование виртуализированной решение GSLB в облачной архитектуре позволяет работать рука об руку с физическим GSLB решением в центре обработки данных и обеспечивает защиту DNS DDoS и гибкость для развертывания DNSSEC в координации с физическим, "мастер" GSLB устройством. Объединение интеллектуальных, географически чувствительное переключение с этими дополнительными мерами безопасности

обеспечивает спокойствие сотрудникам ИТ, предлагая высокую доступность даже в случае стихийного бедствия.

3. Принципы балансировки GSLB

В соответствии с протоколом TCP/IP, когда клиент предоставляет символическое имя («URL») для запроса доступа к прикладной программе или другому типу ресурса, часть имени узла URL-адреса должна быть преобразована в IP-адрес сервера. Например, URL-адрес http://www.foundrynet.com/index.htm включает часть имени узла www.foundrynet.com, которая должна быть преобразована в IP-адрес. Часть имени хоста сначала предоставляется клиентом локальному распознавателю имен, который затем запрашивает локальный DNS-сервер, чтобы получить соответствующий IP-адрес. Если соответствующий IP-адрес локально не кэширован во время запроса или если время жизни (TTL) соответствующего кэшированного IP-адреса истекло, DNS-сервер действует как преобразователь и отправляет рекурсивный запрос к другому DNS-серверу. Этот процесс повторяется до тех пор, пока не будет достигнут авторитетный DNS-сервер для домена, например, foundrynet.com в данном случае. Авторитетный DNS-сервер возвращает один или несколько IP-адресов, каждый из которых соответствует адресу, по которому может быть достигнут конечный сервер, на котором размещено приложение («хост-сервер») под именем хоста. Эти IP-адреса передаются обратно через локальный DNS-сервер к исходному преобразователю. Затем приложение на клиенте использует один из полученных IP-адресов для установления TCP-соединения с соответствующим хост-сервером. Каждый раз DNS-сервер кэширует список IP-адресов, полученных от авторитетного DNS, для ответа на будущие запросы, касающихся того же имени хоста, до истечения TTL IP-адресов.

Чтобы обеспечить некоторое распределение нагрузки между хост-серверами, многие авторитетные DNS-серверы используют простой

циклический алгоритм для поворота IP-адресов в списке откликающихся IP-адресов, чтобы одинаково распределять запросы на доступ среди хост-серверов. Традиционный метод, описанный выше для разрешения имени узла его IP-адресами, имеет несколько недостатков. Во-первых, авторитетный DNS не обнаруживает сервер, который не работает. Следовательно, полномочный DNS-сервер продолжает возвращать IP-адрес запрещенного хост-сервера до тех пор, пока внешний агент не обновит записи авторитетного DNS-сервера. Во-вторых, при предоставлении своего списка IP-адресов авторитетный DNS-сервер не учитывает расположение хост-серверов по отношению к клиенту. Географическое расстояние между сервером и клиентом является фактором, влияющим на время ответа для доступа клиента к главному серверу. Например, при равных условиях трафика клиент из Японии может получить меньшее время отклика с хост-сервера в Японии, чем с хост-сервера в Нью-Йорке. Кроме того, обычный алгоритм DNS позволяет недействительным IP-адресам (например, соответствующему не активному серверу) сохраняться на локальном DNS-сервере до тех пор, пока не истечет срок действия TTL для недопустимого IP-адреса.

Описываемая теоретическая реализация балансировщика обеспечивает усовершенствованный способ и систему по предоставлению IP-адресов клиенту на основе выбранного набора метрик производительности. Коммутатор балансировки нагрузки глобального сервера (GSLB) предоставляется в качестве прокси-сервера для полномочного DNS-сервера, вместе с одним или несколькими коммутаторами сайта, каждый из которых связан с одним или несколькими серверами. Как GSLB устройство, так и коммутатор сайта могут быть реализованы с использованием одного и того же типа коммутационного оборудования в одном варианте реализации. Каждый коммутатор узла предоставляет GSLB устройству текущую информацию об узле, связанную с коммутатором сайта. Когда полномочный DNS-сервер разрешает имя хоста в

запросе и возвращает один или несколько IP-адресов, устройство GSLB фильтрует IP-адреса, используя показатели производительности, собранные из информации о конкретном сайте. Затем коммутатор GSLB возвращает запрашиваемому ранжированный или взвешенный список IP-адресов. В дальнейшем IP-адрес, который, как ожидается, будет обеспечивать наибольшую производительность работы с клиентом, помещается в верхней части списка.

Примеры подходящих показателей производительности включают метрики доступности (состояние сервера или приложения), показатели загрузки (пропускная способность сеанса коммутатора узла или соответствующий заданный порог) и метрики близости (время перехода туда и обратно между коммутатором сайта и запрашивающий DNS-сервер, географическое расположение хост-сервера, топологическое расстояние между хост-сервером и клиентской программой, где топологическое расстояние - это количество переходов между сервером и клиентом. Еще одним показателем близости является скорость «мгновенного возврата» коммутатора сайта, то есть, насколько быстро коммутатор получает результат проверки работоспособности. Еще одним показателем является показатель загрузки подключений, основанный на измерении новых соединений в секунду на сайте. Упорядоченный список также может регулироваться другими политиками, такими как наименее используемый хост-сервер.

В последующем описании даны многочисленные конкретные детали, чтобы обеспечить полное понимание вариантов осуществления предмета исследования. Однако специалист в соответствующей области признает, что реализация может быть осуществлена на практике без одной или более конкретных деталей или другими способами, компонентами и т.д. Некоторые указанные аспекты реализации не показаны или не описаны подробно, поскольку представляют собой коммерческую тайну.

3.1 Общие принципы реализации

Данный пример иллюстрирует один вариант реализации устройства GSLB, который обеспечивает конфигурацию балансировки нагрузки глобального сервера. Как показано на рис. 1 коммутатор 12 балансировки нагрузки глобального сервера (GSLB) подключен к Интернету 14 и действует как прокси-сервер 16 сервера доменных имен (DNS) для выбранного домена «foundrynet.com». Хотя фактическая служба DNS предоставляется сервером 16 DNS, IP-адрес, известный остальной части Интернета для авторитетного DNS-сервера домена «foundrynet.com», представляет собой виртуальный IP-адрес, сконфигурированный на коммутаторе GSLB 12. Конечно, DNS-сервер 16 может также действовать одновременно как авторитетный DNS для других доменов. Коммутатор 12 GSLB сообщает через Интернет 14 с коммутаторами 18A и 18B на сайте 20, коммутаторами 22A и 22B на сайте 24 и любыми другими аналогично сконфигурированными переключателями сайта. Коммутаторы 18A, 18B, 22A и 22B участка показаны соединенными с маршрутизаторами 19 и 21 соответственно и с серверами 26A, , , , 26I,. , , 26N. Некоторые или все из серверов 26A, . . . , 26I, . . . , 26T могут работать с приложениями сервера, используя например, http и ftp. Эти серверы достигают через коммутаторы 18A, 18B, 22A и 22B сайта, используя один или несколько виртуальных IP-адресов, настроенных на коммутаторах сайта, которые действуют как прокси-серверы. Подходящим коммутатором для реализации коммутатора GSLB 12 или любого из коммутаторов 18A, 18B, 22A и 22B сайта является, например, продукт «ServerIron», доступный от Foundry Networks, Inc. Изображение также показывает клиентскую программу 28, связанную с Интернетом 14, и осуществляет связь с локальным DNS-сервером 30. Когда браузер на клиенте 28 запрашивает веб-страницу, например, используя универсальный указатель ресурса (URL) на

http://www.foundrynet.com/index.htm, запрос отправляется на локальный DNS-сервер 30 для разрешения символического имени хоста www.foundrynet.com на IP-адрес хост-сервера. Клиентская программа получает от DNS-сервера 30 список IP-адресов, соответствующих разрешенному имени хоста. Этот список IP-адресов извлекается из кэша локального DNS-сервера 30, если TTL IP-адресов в кэше не истек или получен из коммутатора GSLB 12 в результате рекурсивного запроса. Однако, в отличие от предшествующего уровня техники, этот список IP-адресов упорядочен коммутатором 12 GSLB на основе показателей производительности, более подробно описанных ниже. Далее предполагается, что список возвращенных IP-адресов является виртуальными IP-адресами, настроенными на прокси-серверах на коммутаторах 18A, 18B, 22A и 22B (сайты Пункты 20 и 24).

В одном варианте осуществления коммутатор 12 GSLB определяет, какой коммутатор узла обеспечит наилучшую ожидаемую производительность (например, время ответа) для клиента 28 и вернет список IP-адресов с виртуальным IP-адресом, сконфигурированным на коммутаторе узла, расположенном в верхней части. Клиентская программа 28 может получать упорядоченный список IP-адресов и обычно выбирает первый IP-адрес в списке, чтобы получить доступ к соответствующему хост-серверу.

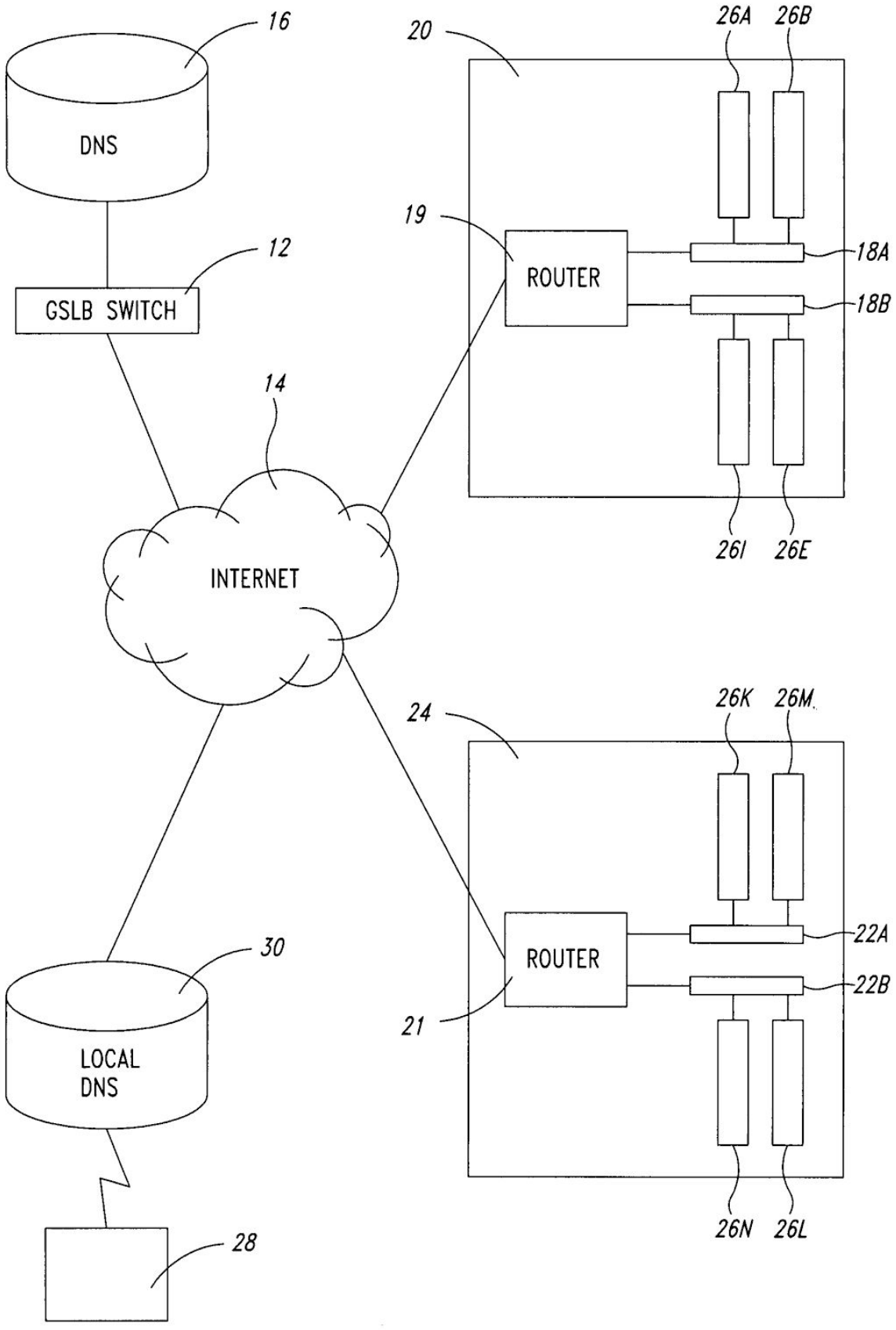

16
DNS
12
GSLB SWITCH
14
INTERNET
20
26A
26B
19
ROUTER
18A
18B
26I
26E
24
26K
26M
ROUTER
22A
22B
21
26N
26L
30
LOCAL
DNS
28

Рисунок 1.

На рис 2. изображена блок-схема, показывающая функциональные модули коммутатора 12 GSLB и коммутатора 18A сайта, относящиеся к функции глобального распределения нагрузки на сервер. Переключатель 12 GSLB включает в себя контроллер 4L переключателя GSLB, модуль 402 проверки работоспособности, модуль 403 прокси-сервера DNS, агент 404 метрики, сборщик 405 метрики маршрутизации и сборщик 406 метрики участка. Контроллер 401 переключателя GSLB предоставляет общие функции управления переключателем GSLB 12. Модуль 402 проверки работоспособности отвечает за периодический запрос или запрос по требованию хост-серверов и соответствующих приложений, размещенных на хост-серверах, для определения «работоспособности», независимо от того, доступен ли он или нет. Сборщик 406 метрик на конкретном участке взаимодействует с метрическими агентами в коммутаторах, специфичных для сайта, чтобы собирать метрики для конкретных сайтов, такие как количество доступных сеансов на определенном хост-сервере и/или данные загрузки соединения на этом хост-сервере.

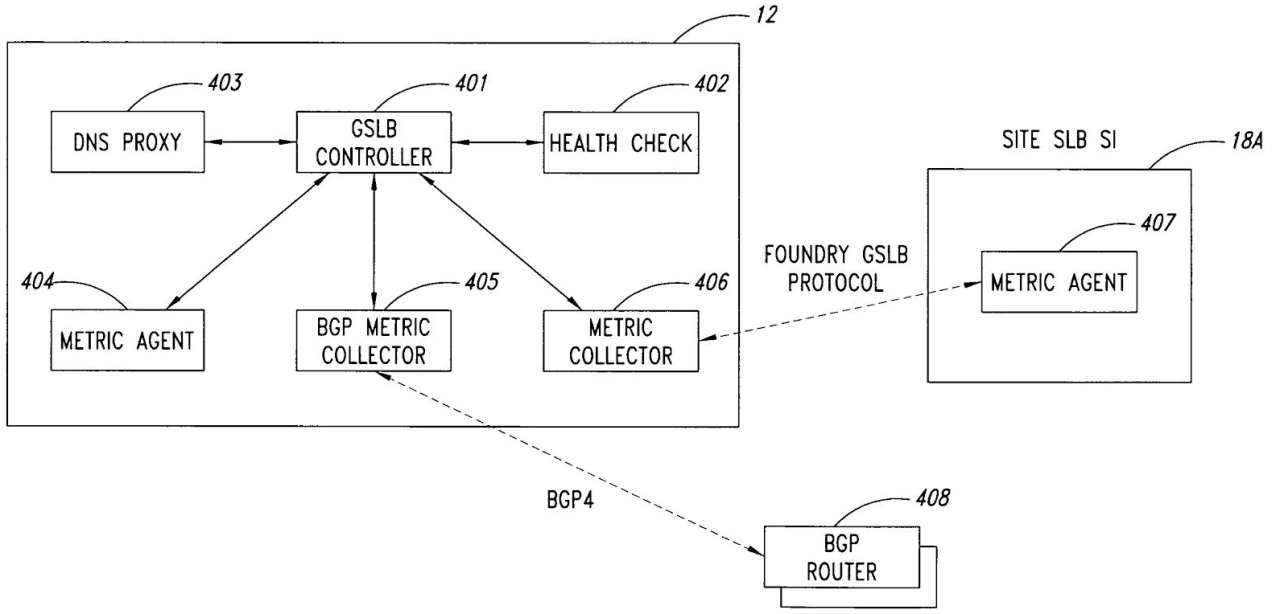


Рисунок 2.

Агент(ы) 407 метрики на уровне объекта могут выполнять сбор информации для получения метрий соединений в секунду на своем соответствующем сайте, а затем получают средние значения нагрузки из выборок или выполняют другие вычисления. После этого узловой счетчик 406 узла GLSB получает средние нагрузки от специализированного метрического агента(ов) 407 и предоставляет эти средние нагрузки контроллеру 401 переключателя, чтобы позволить ему использовать эти показания для ранжирования IP-адресов в упорядоченном списке. В качестве альтернативы или в дополнение к метрическому агенту (агентам) 407 для конкретных сайтов контроллер 401 коммутатора может выполнять, по меньшей мере, некоторые или большинство вычислений нагрузки соединения из данных выборки, предоставляемых агентом(ами).

Сборщик 405 метрик маршрутизации собирает информацию от маршрутизаторов (например, топологические расстояния между узлами в Интернете). На рисунке 2 также показан маршрутизатор 408, обеспечивающий сборщик 405 метрик получающий топологическое расстояние между переключателем балансировки нагрузки и маршрутизатором с использованием протокола граничного шлюза (BGP). Модуль 403 прокси-сервера DNS (a) принимает входящие DNS-запросы, (b) предоставляет имена хостов, которые должны быть разрешены для DNS-сервера 16, (c) принимает от DNS-сервера 16 список реагирующих IP-адресов, (d) заказывает IP-адреса на список, полученный от сервера 16 DNS в соответствии с вариантом осуществления настоящего изобретения, с использованием метрик, собранных сборщиком маршрутизации 405 и сборщиком 406 сайта, и значениями любого другого соответствующего параметра, и (e) обеспечивает упорядоченный список IP Адреса к запрашиваемому DNS-серверу. Поскольку коммутатор 12 GSLB

может также действовать в качестве коммутатора сайта, коммутатор 12 GSLB предоставляет агента 404 конкретного участка для сбора метрик для коллектора метрик на конкретном участке.

Используемые в коммутаторе 12 GSLB метрики включают в себя (a) проверку работоспособности каждого хост-сервера и выбранных приложений, (b) порог пропускной способности сеанса каждого коммутатора узла, (c) время прямого прохождения (RTT) между коммутатором узла, а также имя клиента в предыдущем доступе, (d) географическое расположение хост-сервера, (e) показатель загрузки соединения для новых подключений в секунду на коммутаторе сайта, (f) текущая доступная емкость сеанса на каждом сайте, g) скорость «обратного проскальзывания» между каждым коммутатором узла и коммутатором GSLB, то есть, как быстро каждый коммутатор узла реагирует на проверку работоспособности от переключателя GSLB) и (h) политика, называемая «выбор с наименьшим откликом», (LRS), который предпочитает сайт, наименее используемый ранее. Многим из этих показателей производительности могут быть предоставлены значения по умолчанию. Каждый отдельный показатель можно использовать в любом порядке и каждый показатель можно отключить.

На рисунках. 3A-3D приведены иллюстрации на блок-схемах, в которых один вариант осуществления алгоритма оптимизации, используемого коммутатором 12 GSLB для обработки списка IP-адресов, принятого от DNS-сервера 16, в ответ на запрос, полученный в результате выполнения программы-клиента 28, рисунок 3D показывает относительное положение частей технологической схемы, показанной на рисунках 3A-3C. По меньшей мере некоторые из элементов блок-схемы могут быть реализованы в программном обеспечении или другой машиночитаемой инструкции,

хранящейся на одном или более машиночитаемых носителях. Такое программное обеспечение для выполнения частей алгоритма может присутствовать в коммутаторе 12 GSLB в одном варианте осуществления и исполнено контроллером 401 переключателя.

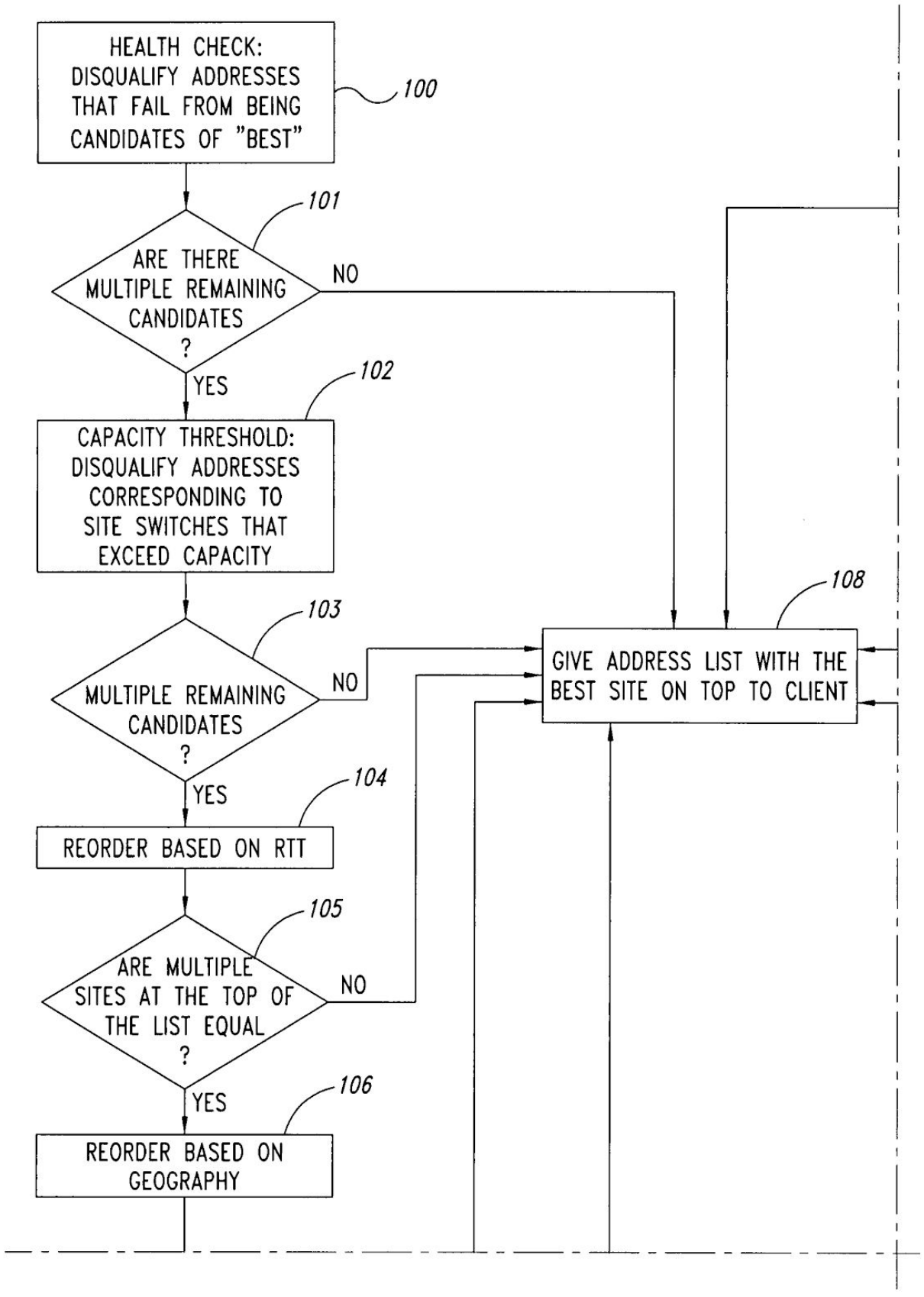

HEALTH CHECK: DISQUALIFY ADDRESSES THAT FAIL FROM BEING CANDIDATES OF "BEST"
100
ARE THERE MULTIPLE REMAINING CANDIDATES ?
101
NO
YES
CAPACITY THRESHOLD: DISQUALIFY ADDRESSES CORRESPONDING TO SITE SWITCHES THAT EXCEED CAPACITY
102
MULTIPLE REMAINING CANDIDATES ?
103
NO
YES
REORDER BASED ON RTT
104
ARE MULTIPLE SITES AT THE TOP OF THE LIST EQUAL ?
105
NO
YES
REORDER BASED ON GEOGRAPHY
106
GIVE ADDRESS LIST WITH THE BEST SITE ON TOP TO CLIENT
108

Рисунок 3А.

Как показано на фиг. 3A, в действии 100 после получения списка IP-адресов от DNS-сервера 16 коммутатор 12 GSLB выполняет для каждого IP-адреса в списке IP-адресов (например, хост-сервер 261, подключенный к коммутатору 18B сайта) проверку работоспособности уровней 4 и 7. Здесь уровни 4 и 7 относятся соответственно к протоколам транспорта и приложений в уровнях протокола Open System Interconnection (OSI). Проверкой работоспособности уровня 4 может быть проверка работоспособности протокола TCP (Transmission Control Protocol) или проверка работоспособности протокола UDP (User Datagram Protocol). Такая валидация может быть достигнута, например, посредством операции «ping-like», определенной в соответствии с соответствующим протоколом. В соответствии с TCP-протоколом пакет TCP SYN может быть отправлен, а работоспособность целевого объекта устанавливается, когда соответствующий пакет ACK TCP получен обратно от целевого объекта. В этом варианте осуществления проверка работоспособности уровня 7 предоставляется для определенных приложений, таких как широко известный протокол передачи гипертекста (HTTP) и приложения протокола передачи файлов (FTP). Если хост-сервер или связанное приложение отказывают в какой-либо проверке работоспособности, он лишается права (акт 100) на «лучший» сайт и может быть исключен из списка IP-адресов, который должен быть возвращен в клиентскую программу 28. Поскольку проверка работоспособности показывается независимо от того, доступен ли хост-сервер или связанное приложение, метрика проверки работоспособности подходит для исключения IP-адреса от кандидатов на «лучший» IP-адрес, т.е.хост-сервер должен обеспечить максимальную производительность). После действия 100, если список IP-адресов имеет только один IP-адрес (действие 101), список IP-адресов возвращается клиентской программе 28 на этапе 108.

После шага 100, если список возможных IP-адресов для лучшего сайта имеет несколько IP-адресов, он дополнительно оценивается в действии 102 на основании порога емкости коммутатора узла, обслуживающего этот IP-адрес. Каждый коммутатор сайта может иметь разное максимальное количество сеансов TCP, которые он может обслуживать. Например, номер по умолчанию для продукта «ServerIron» Foundry Network - один миллион сеансов, хотя его можно настроить на меньшее число. Виртуальный IP-адрес, сконфигурированный на коммутаторе 18B сайта, может быть дисквалифицирован как «лучший» IP-адрес, если число сеансов для коммутатора 18B превышает заданный пороговый процент (например, 90%) от максимального количества сеансов. Конечно, пороговое значение 90% от максимальной емкости может быть изменено. После действия 102, если список IP-адресов имеет только один IP-адрес (действие 103), список IP-адресов возвращается клиентской программе 28 на этапе 108.

Рисунок 3B.

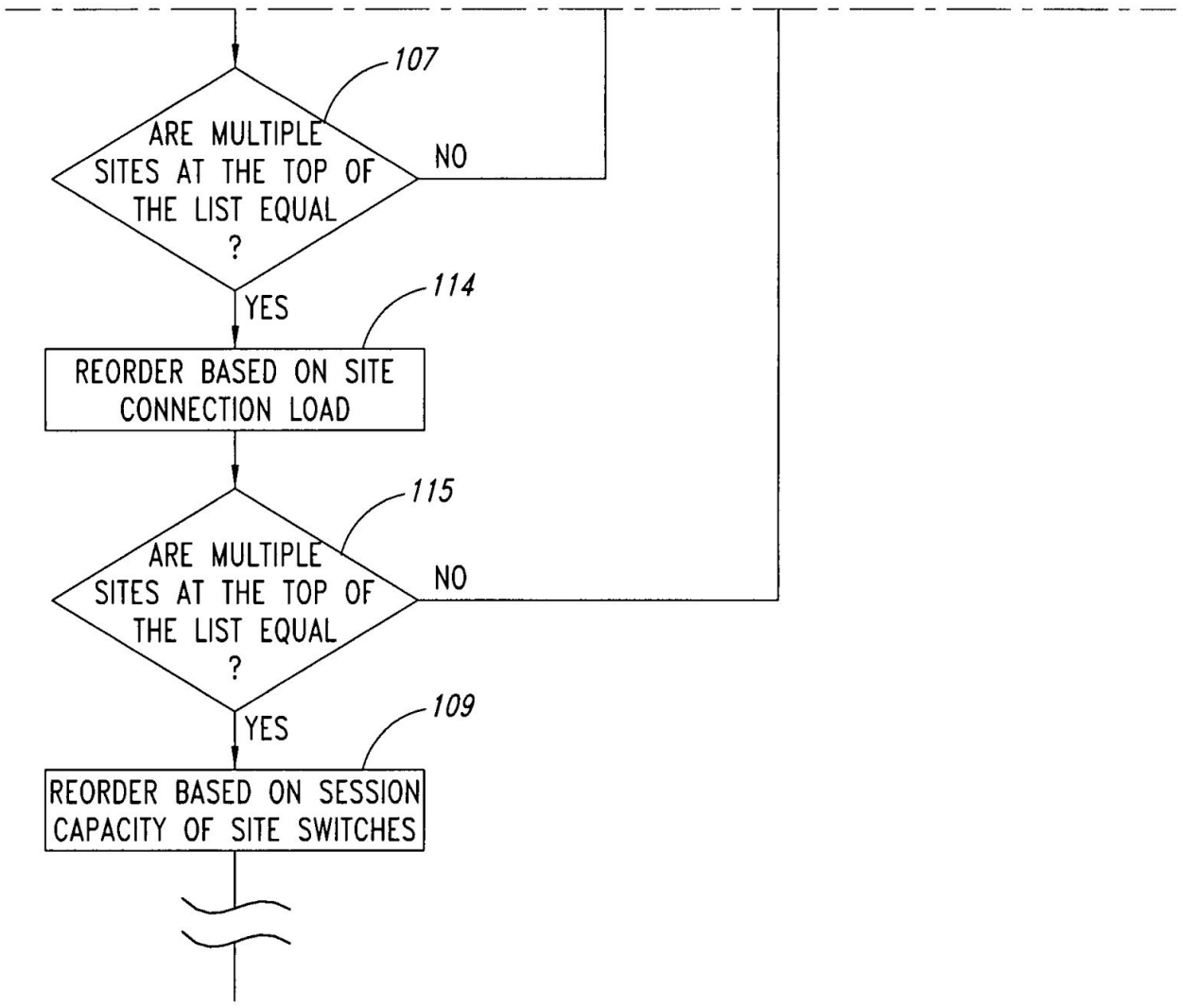


Рисунок 3B.

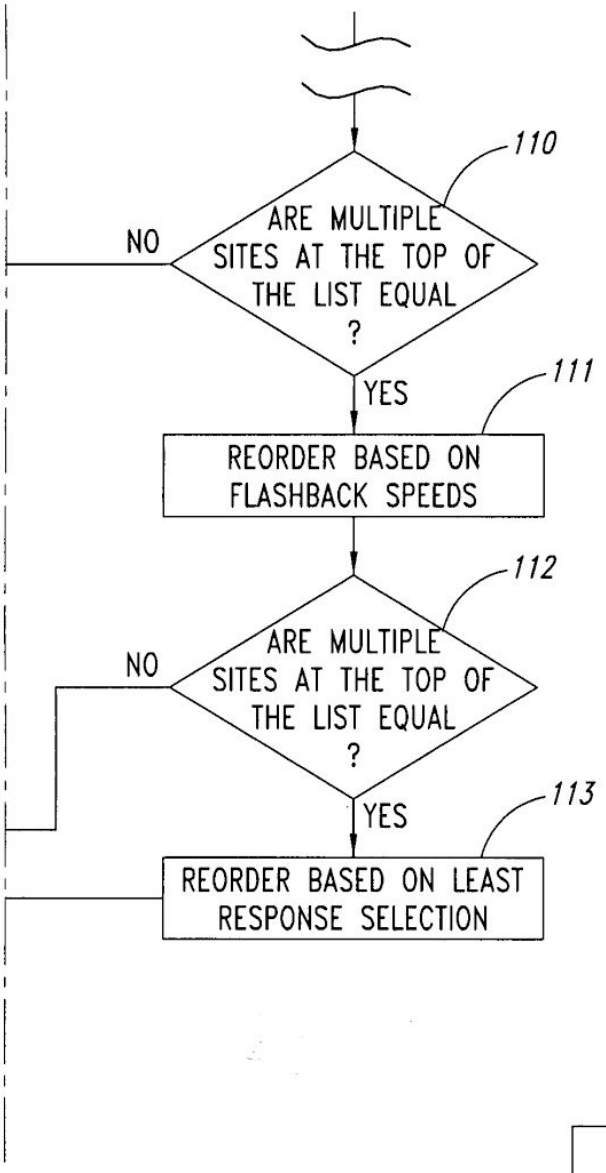
После действия 102, если список IP-адресов имеет несколько IP-адресов (действие 103), оставшиеся IP-адреса в списке могут быть переупорядочены в действии 104 на основе времени RTT между коммутатором узла для IP-адреса (коммутатор 18B сайта) и клиент (например, клиент 28). RTT вычисляется для интервала между временем, когда клиентская машина запрашивает TCP-соединение с прокси-сервером, сконфигурированным на коммутаторе сайта, отправляя прокси-сервер TCP SYN-пакет и временем, когда коммутатор сайта получает от клиентской программы TCP ACK-пакет. В ответ на пакет TCP SYN хост-сервер отправляет пакет TCP SYN ACK, чтобы указать прием

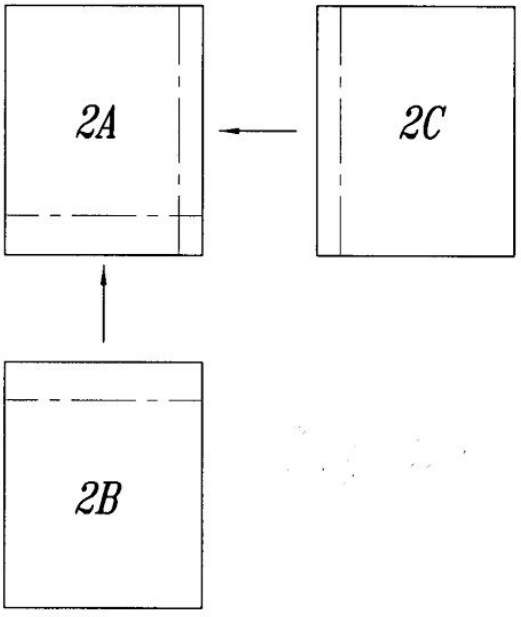
TCP-соединения, клиентский компьютер возвращает пакет TCP ACK, чтобы завершить настройку TCP-соединения. Переключатель GSLB (Например, коммутатор GSLB 12) поддерживает базу данных RTT, которую он создает и обновляет из данных, периодически принимаемых от коммутаторов сайта (в данном примере коммутаторы сайта 18A, 18B, 22A и 22B). Каждый сайт собирает и сохраняет данные RTT для каждого TCP-соединения, установленного с клиентской машиной. В одном варианте осуществления коммутатор GSLB поддерживает один сервер хоста по другому только в том случае, если разница в их RTT с клиентским компьютером превышает указанный процент, например, заданное по умолчанию процентное значение составляет например 10%. Чтобы предотвратить смещение, GSLB-переключатель игнорирует, по умолчанию, значения RTT для 5% клиентских запросов из каждой сети ответа, например. После действия 105, если верхние записи в списке IP-адресов не имеют равных RTT, список IP-адресов возвращается в клиентскую программу 28 на этапе 108.

Если на нескольких сайтах установлены одинаковые RTT (действие 105), то список переупорядочивается в действии 106 на основании географического местоположения хост-сервера. Географическое расположение сервера определяется в зависимости от того, является ли IP-адрес реальным адресом или виртуальным IP-адресом («VIP»). Для реального IP-адреса географический регион для хост-сервера можно определить с самого IP-адреса. В рамках IANA региональным реестрам RIPE (Европа), APNIC (Азиатско-Тихоокеанский регион) и ARIN (Северная и Южная Америка и Африка) назначаются разные блоки префиксов. Предполагается, что IP-адрес, администрируемый одним из этих региональных реестров, соответствует машине, расположенной внутри географического района, находящегося в ведении регионального реестра. Для VIP, географический регион определяется из IP-адреса управления

соответствующего коммутатора сайта. Конечно, для любого IP-адреса можно указать географический регион, определенный в соответствии с приведенной выше процедурой. Коммутатор GSLB предпочитает IP-адрес, который находится в том же географическом регионе, что и клиентский компьютер. На этапе 107, если две верхние записи в списке IP не имеют одинакового ранжирования, список IP отправляется в клиентскую программу 28 на этапе 108. После действия 107, если несколько сайтов имеют равный рейтинг для лучшего сайта, IP-адреса затем могут быть переупорядочены на основе загрузки соединения с сайтом (действие 114). Функция метрики загрузки соединения позволяет сравнивать сайты на основе загрузки соединения с их соответствующим агентом.

Нагрузка соединения является мерой новых соединений в секунду на агенте 407. Администратор может установить предельное пороговое значение для подключения к определенному сайту; может выбирать количество интервалов выборки нагрузки и продолжительность каждого интервала; может выбирать относительный вес для каждого интервала для вычисления средней нагрузки в течение периода времени (то есть новых соединений за период времени).





Рисунки 3C и 3D.

Значение ограничения нагрузки на соединение определяет ограничение нагрузки для любого сайта, передающего метрику. Минимальное значение равно 1, и парсер или другой компонент программного обеспечения в коммутаторе 18A сайта ограничивает максимальное значение - там не обязательно должно быть значение по умолчанию. По умолчанию эта метрика подключения отключается и может включаться при задании ограничения нагрузки. Средняя нагрузка для данного сайта рассчитывается с использованием пользовательских весов и интервалов. Если расчетная средняя нагрузка меньше указанного предела нагрузки, сайт переходит к следующему этапу описанного здесь алгоритма GSLB - в противном случае этот сайт исключается/отбрасывается из набора потенциальных кандидатов.

В одном варианте осуществления может быть сконфигурировано количество «интервалов выборки нагрузки», а также «частота выборки». Частота дискретизации определяет продолжительность каждого интервала выборки в кратном размере от начальной скорости. Например, если выбраны 6 интервалов выборки и частота выборки в 5 секунд, сайт будет отбирать среднюю нагрузку по 5, 10, 15, 20, 25 и 30. В любой момент времени у сайта будет средняя нагрузка для предыдущих 5 секунд, 10 секунд, 15 секунд, 20 секунд, 25 секунд и 30 секунд. Это «скользящее среднее» в том, что на 35-й секунде, например, вычисляется среднее значение с 5-й по 35-ю секунды. Скользящее среднее/точность ограничена начальной частотой дискретизации, а это значит, что, поскольку выборки берутся через каждые 5 секунд, на седьмой секунде, доступно среднее значение для первой и пятой секунд, а не среднее значение от 2 до 7 секунд. Частота дискретизации также определяет интервал обновления для сайта (агент 407 сайта) для загрузки средних значений нагрузки в метрический коллектор 406 на GSLB-переключателе 12. Данный сайт способен поддерживать средние нагрузки для любого числа коллекторов за

один раз. Каждый коллектор периодически обновляется информацией о нагрузке, а интервал обновления также специфичен.

Минимальное количество интервалов равно 1, а максимальное значение равно 8 в данной реализации. По умолчанию используется номер 5, который устанавливается при настройке ограничения нагрузки подключения. Понятно, что это просто иллюстративные примеры и могут отличаться в зависимости от конкретной реализации. Для интервала выборки нагрузки минимальное значение составляет 1 секунду, а максимальное значение - 60 секунд. Значение по умолчанию - 5 секунд. Следовательно, максимальный диапазон для расчета средней нагрузки составляет 60 * 8 секунд = 480 секунд = 8 минут. Таким образом, можно рассмотреть до среднего 8-минутного среднего для анализа нагрузки.

Каждому интервалу можно назначить весовые коэффициенты для расчета средней нагрузки. По умолчанию в одном варианте осуществления каждому интервалу присваивается равный вес 1. Средняя нагрузка для сайта может быть рассчитана по следующей формуле:

$$\frac{\sum_{i=0}^{N} (AvgLoad \text{ of interval } i) * (\text{Weight of interval } i)}{\sum_{i=0}^{N} (\text{Weight of interval } i)}$$

где N - количество интервалов, AvgLoad of interval i - новые соединения за интервал i.

Вклад любого интервала может быть аннулирован, если его вес равен нулю. Если каждому интервалу задан вес, равный нулю, средняя нагрузка равна нулю. В одном варианте осуществления метрический агент 407 для конкретной

местности может вычислять эту среднюю нагрузку и предоставлять ее в метрический коллектор 406 на переключателе 12 GSLB. В других вариантах осуществления метрический коллектор 406 и/или контроллер 401 переключателя могут выполнять вычислений средней нагрузки, основанных на значениях, собранных и представленных агентом 407 метрик конкретного сайта. По умолчанию метрика подключения нагрузки не включена в алгоритме GSLB. Метрика автоматически включается, когда пользователь задает предел нагрузки на соединение. Конкретные требования к конфигурации для выборки и расчета нагрузки на соединение могут быть сконфигурированы на контроллере 401 переключателя, независимо от того, используется ли переключатель 12 для GSLB или как переключатель для конкретного участка.

Все веса для всех интервалов не нужно настраивать, если они не рассматриваются за пределами определенной точки. Сконфигурированные веса будут назначаться на интервалы, начинающиеся с первого, и любому не настроенному интервалу будет присвоен нулевой вес.

Таким образом, несмотря на то, что 6 интервалов сконфигурированы в приведенном выше примере, все остальные сведены к нулю из-за нулевых весов. По умолчанию метрика подключения нагрузки может быть не включена в алгоритм GSLB. После настройки предела нагрузки соединения метрика включается после метрики географического местоположения в метрической последовательности согласно одному варианту осуществления, например, как показано на фиг. 2B. Понятно, что порядок метрик может быть изменен или настроен.

В действии 115, если нет нескольких кандидатов в верхней части списка IP, которые прошли метрику загрузки соединения (или их нет равного ранга), список IP-адресов отправляется в клиентскую программу 28 на этапе 108. После действия 115, если несколько сайтов имеют равный рейтинг для лучшего сайта, IP-адреса затем могут быть переупорядочены на основе доступной

емкости сеанса (действие 109). Например, в одном варианте осуществления, если коммутатор 18A имеет 1 000 000 сеансов, а коммутатор 22B имеет 800 000 сеансов, тогда предпочтительным является переключатель 18A, если предел допуска, представляющий разницу в доступных сеансах, выраженный как процент от емкости в более крупном коммутаторе, является приоритетным. Например, если предел допуска равен 10%, переключатель 18A должен будет иметь как минимум 100 000 дополнительных сеансов, доступных, чем переключатель 22B, чтобы быть предпочтительным. Если предпочтительным является IP-адрес (действие 110), IP-адрес будет помещен в верхнюю часть списка IP-адресов и затем возвращен запрашивающему объекту на этапе 108. В противном случае, если емкость сеанса не разрешает лучший IP-адрес, действие 111 затем предпринимает попытку разрешения, основанного на скорости «обратного хода». Скорость ретроспективного анализа - это время, требуемое коммутатору сайта для реагирования на проверки состояния слоев 4 и 7 с помощью переключателя GSLB. Скорость ретроспективного анализа, таким образом, является мерой нагрузки на хост-сервер. Опять же, предпочтительный IP-адрес будет соответствовать скорости повтора обратного вызова, превышающей следующую, на заданный предел допуска.

Скорости ретроспективного отражения измеряются для хорошо известных приложений (уровень 7) и соответствующих им портов TCP (уровень 4). Для других приложений скорость флешбека измеряется для выбранных пользователем портов TCP. Сначала сравниваются скорости ретроспективного воспроизведения уровня 7 (уровня приложения), если это применимо. Если флэшбэки приложения не обеспечивают лучший IP-адрес, сравниваются скорости повторения кадров уровня 4. Если хост-сервер связан с несколькими приложениями, переключатель GSLB выбирает самое медленное время отклика среди приложений для сравнения. На этапе 112, если лучший IP-адрес разрешен, список IP-адресов отправляется в клиентскую программу 28 на этапе

108. В противном случае на этапе 113 IP-адрес на сайте, который наименее часто выбирается как «лучший» сайт. Затем список IP-адресов отправляется в клиентскую программу 28 (действие 108). После получения списка IP-адресов программа-клиент 28 использует лучший выбранный IP-адрес (то есть верхнюю часть списка), чтобы установить TCP-соединение с хост-сервером. Даже в том случае, если внезапный всплеск трафика приводит к перегрузке хост-сервера или когда хост-серверы или приложения на сайте становятся недоступными, коммутатор сайта может перенаправить запрос TCP-соединения на другой IP-адрес, используя, например, существующую процедуру перенаправления HTTP.

Чтобы предоставить RTT в соответствии с вариантом реализации, описанной выше, при первом доступе клиента к IP-адресу коммутатор сайта (например, коммутатор сайта 22A на фиг.2) контролирует время RTT - разность времени между получением TCP SYN и TCP ACK для TCP-соединения - и записывает его в запись базы данных кэша. Измеренное таким образом время RTT соответствует естественному потоку трафика между клиентом и указанным хост-сервером, а не искусственным RTT, основанным на «ping» клиентской машины по стандартному сетевому протоколу. Периодически коммутаторы сайта сообщают базу данных RTT переключателю GSLB наряду с условиями нагрузки таких как количество доступных сеансов. Коммутатор GSLB объединяет сообщения RTT, указанные в таблице приближений, проиндексированной сетевым окружением (сетевое окружение - это часть сети, использующая префикс IP-адреса. Таким образом, коммутатор GSLB может искать RTT для клиентской машины на любом конкретном хост-сервере на основе сетевого окружения клиента, указанного на IP-адресе клиента, и, благодаря доступу к хост-серверам, из большого числа сетевых окружений коммутатор GSLB может создать всеобъемлющую базу данных о географической близости, которая обеспечивает более разумный выбор сайта.

Для того чтобы сохранить таблицу приближения полезной и актуальной, коммутатор GSLB управляет таблицей близости с политиками управления кэшем, например, очистка редко используемых записей в пользу недавно полученных RTT. Данные о близости могут использоваться для всех IP-адресов, обслуживаемых каждым коммутатором сайта.

4. Алгоритм балансировки

В результате работы и анализа существующих решений, технологий и потребностей рынка было получен алгоритм балансировки, позволяющий решать ряд проблем, связанных с распределением нагрузки, увеличением скорости доступа и отказоустойчивостью сервиса или приложения, а также с производительностью. Общая принцип, на котором происходит балансировка можно представить следующим образом:

$$S = \sum_i k_i(x_i - m_i)$$

- S - суммарный показатель приоритета при балансировки. Сервер с наибольшим показателем S будет определен как наиболее предпочтительный для данного пользователя.
- k_i - коэффициент, определяющий важность метрики в зависимости от заданных настроек. Принимает положительное значение, если метрика учитывается и 0, если нет. Более подробно о нем будет рассказано ниже для каждой из описанных метрик.
- x_i - текущая свободная мощность рассматриваемого ресурса.
- m_i - пороговое значение для рассматриваемой метрики. Представляет собой константу и устанавливается администратором балансировщика в зависимости от характеристик сервера.

Для каждой из вышеописанных метрик величины k_i и m_i выбираются в зависимости от рассматриваемой метрики и расставления ее приоритетов.

В случае, когда хотя бы одна из разностей по любой из метрик меньше или равна нулю, суммарный показатель приоритета считается равным 0, т.е. данный сервер имеет наименьший приоритет. В зависимости от предпочтений

компании по балансировке, коэффициенты k выбираются минимальными для неприоритетной метрики, и много большего значения для метрики, по которой выставлен приоритет.

Для метрик сервера x_i и m_i представляют собой относительные значения ресурсов сервера. Для памяти они будут обозначаться как текущая доля свободной памяти и минимально возможная доля свободной памяти. Аналогичный подход применяется для CPU и дискового пространства - текущая свободная мощность и ее минимальное пороговое значение. Стоит отметить, что существует множество алгоритмов динамического распределения ресурсов серверов в пределах одного дата-центра, однако, данный вид балансировки не рассматривается в данной работе, поскольку более подробно описывает зависимости между техническими ресурсами облака, и как следствие, представляет собой несколько иную область исследования. С целью упрощения принятие решения в пределах данной работы, данная метрика будет считаться настраиваемой вручную.

Метрики трафика сервера и пользовательские настройки, фактически, являются эквивалентными друг другу. Однако, метрики сервера устанавливаются провайдером облачных услуг и не могут быть изменены администратором балансировщика, тогда как для последнего эти значения могут регулироваться отдельно и для отдельного клиента, и для конкретного сервера. Кроме того, существует необходимость разграничивать пользователей приложения или ресурса в зависимости от их важности. Например, если сервис предоставляет клиенту свой функционал на платной основе, то ресурсы для него должно быть предоставлены в первую очередь, в то время как услуги для пользователей, пользующихся сервисом бесплатно, будут предоставляются вторично.

Метрики времени представляют собой временные задержки, который текущий клиент может испытывать при доступе к ресурсу. Данные показания могут варьироваться в зависимости от текущей загрузки того или иного сервера, предоставляющего доступ к ресурсу, объема обрабатываемого трафика, а так же, что не менее важно, от его географической дислокации. По мере увеличения дальности расположения от клиента, время доступа к ресурсу на удаленном сервере может существенно возрастать, и как следствие, временный задержки по этому показателю могут быть существенно выше, нежели в зависимости от других метрик. Кроме того, существует возможность автоматического перенаправления клиента на наиболее близко расположенный к нему центр предоставления услуг с целью снижения задержек при доступе и работе с ресурсом.

Как упоминалось выше, расчет итогового значения приоритета может быть упрощен за счет установления порядка нахождения произведений по метрикам с целью выявить наименее приоритетные серверы. Так, например, расчет времени доступа должен быть получен в первую очередь, поскольку сервер может быть недоступен и находить остальные значения не имеет смысла. Далее следует вычислить метрики трафика и сессий, так как сервер может быть не способен принимать данные или большее число клиентов чем есть сейчас. После этого производится расчет метрик производительности сервера для выявления способности обеспечить работу запрашиваемого ресурса с учетом потенциального возрастания нагрузки. В последнюю очередь происходит расчет задержек при использовании выбранного сервера на основании уже обрабатываемых клиентских сессий.

Существенным плюсом данного алгоритма является его гибкость в настройках в зависимости от функционала предоставляемого сервиса. Так, например, в случае необходимости быстрого доступа и обработки запросов клиент может быть перенаправлен на более загруженный, но более близкий к клиенту сервер. В другом случае, когда производительность имеет более существенное значение, клиент, напротив, будет перенаправлен на более удаленный, но менее загруженный сервер.

Стоит отметить, что при доступе одному и тому же ресурсу сервера, потребности клиентов в вычислительных мощностях и трафике могут варьироваться, поэтому необходимо прогнозировать потенциальное возрастание нагрузки по той или иной метрике с целью избежания достижения пороговых значений по свободным мощностям и полного отказа в предоставлении услуг.

В реальных условиях не всегда удается обеспечить однородность в ресурсах имеющихся серверов. Так, например, компания может располагать набором серверов, отличающихся по мощности в несколько раз. Данная неоднородность при определенных условиях может существенно влиять на окончательный выбор при балансировке. Например, один из серверов может иметь относительно большие мощности по объему предоставляемой памяти, дисковому пространству и процессоров, но иметь существенную загрузку по одной из этих метрик. Напротив, второй сервер может иметь обладать значительно меньшими мощностями, но быть более свободным от загрузки. В результате, более выгодным решением будет перенаправить клиента на второй сервер во избежании достижения полной загрузки ресурсов на первом сервере.

Базовые алгоритмы балансировки не имеют существенного недостатков в производительности, по сравнению с GSLB, в случае, когда объем доступных ресурсов серверов достаточно высокий и количество клиентов незначительное. Однако, по мере возрастания числа пользователей и, как следствие, возникающих ограничений в мощностей на единичном сервере данная проблема распределения ресурсов становится более существенной. Далее будут рассмотрены несколько крайних частных случаев, когда стандартные алгоритмы балансировки менее эффективны, чем GSLB.

4.1 Приоритет по свободным ресурсам

В данном примере будет рассмотрен частный случай при выставлении приоритетов балансировки, в зависимости от доступных ресурсов серверов.

Допустим, компания располагает в своих ресурсах три отличающихся по мощности сервера, географически расположенных в разных частях земного шара и в разных дата-центрах. Первый сервер считаем наиболее приоритетным и обладающим наибольшим количеством ресурсов, второй и третий считаем в 2 раза слабее. Каждый из серверов курируется одним GSLB контроллером. Зададим начальные условия

- Сервер 1: текущая загрузка ресурсов 70%, загрузка трафика 40%, условное задержка времени доступа 1 ед., пороговое значение m равное 10%
- Сервер 2: текущая загрузка ресурсов 10%, загрузка трафика 5%, условная задержка времени доступа в 2 ед., пороговое значение m равное 30%
- Сервер 3: текущая загрузка ресурсов 5%, загрузка трафика 1%, условная задержка времени доступа в 3 ед., пороговое значение m равное 20%

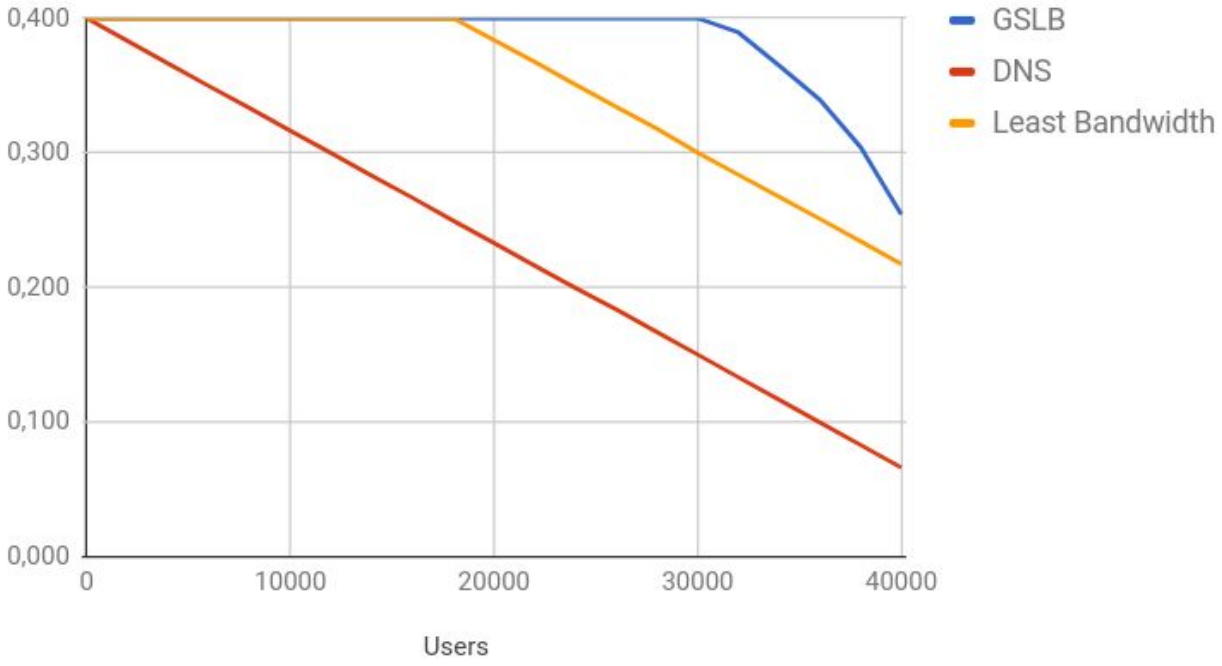
Пороговые значения по времени доступа будем считать равным 5 ед. времени для каждого из серверов. Пороговые значения для остальных метрик будем считать равными 0, максимальное количество пользовательских сессий считаем равным 15000 для каждого из серверов. Считается, что увеличение задержек при использовании ресурсов растет прямо пропорционально загрузки ресурсов и трафика, по 1 ед. времени на каждые 40% доступной мощности.

Предположим, что количество клиентов использующих данный ресурс будет возрастать от 0 до 40000. Каждый из клиентов потенциально может потреблять от 0.0001 до 0.0010% мощности сервера и трафика, а также освобождать потребляемые ресурсы с течением времени и отключаться. Количество потребляемых клиентов ресурсов сервера варьируется в зависимости от максимальной мощности сервера.

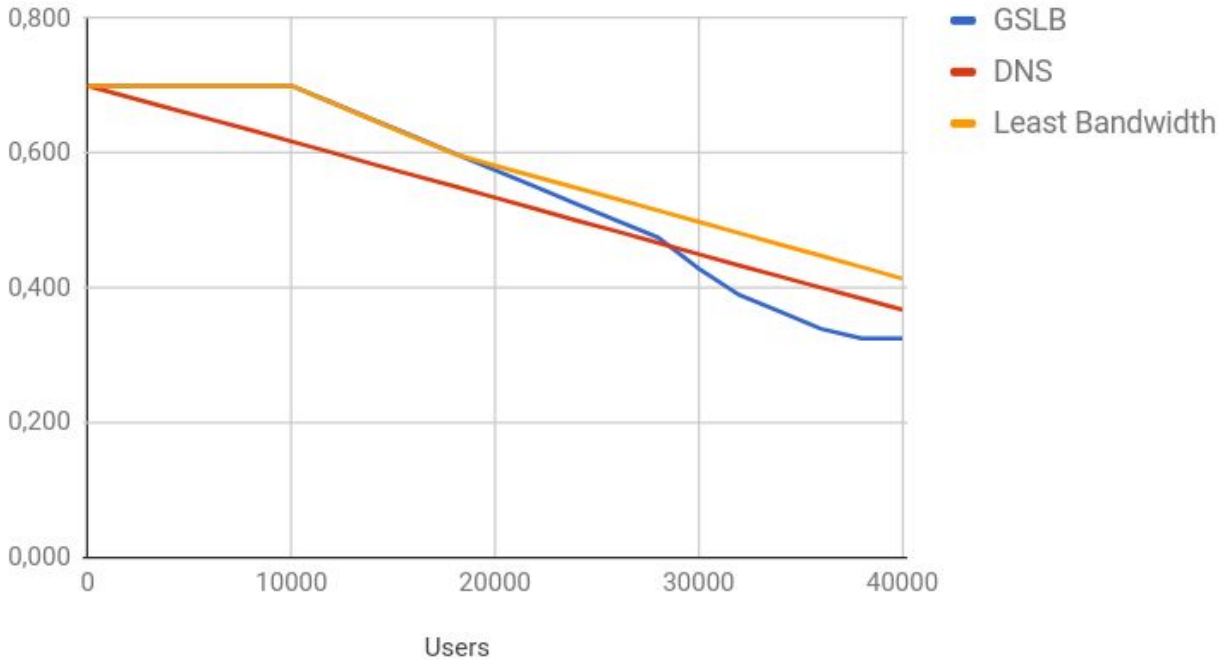
Целью данного частного случая будет достижение наиболее равномерного распределения нагрузки между доступными серверами и избежание ситуации полного отказа дата-центра в предоставлении услуг. В качестве объектов сравнения будут выступать альтернативные алгоритмы балансировки, такие как DNS Round-Robin и Least Bandwidth.

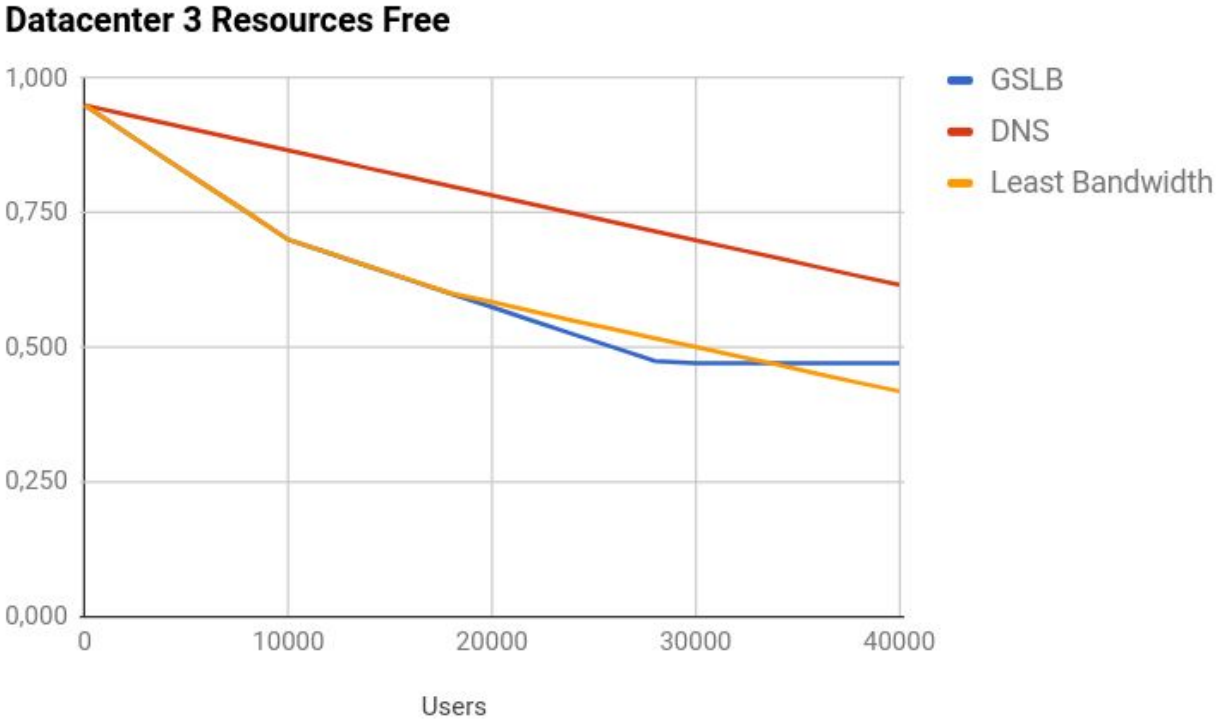
На основании заданных параметров ниже приведены графики загрузки серверов с течением времени, трафика и задержек при использовании.

Datacenter 1 Resource Free



Datacenter 2 Resources Free



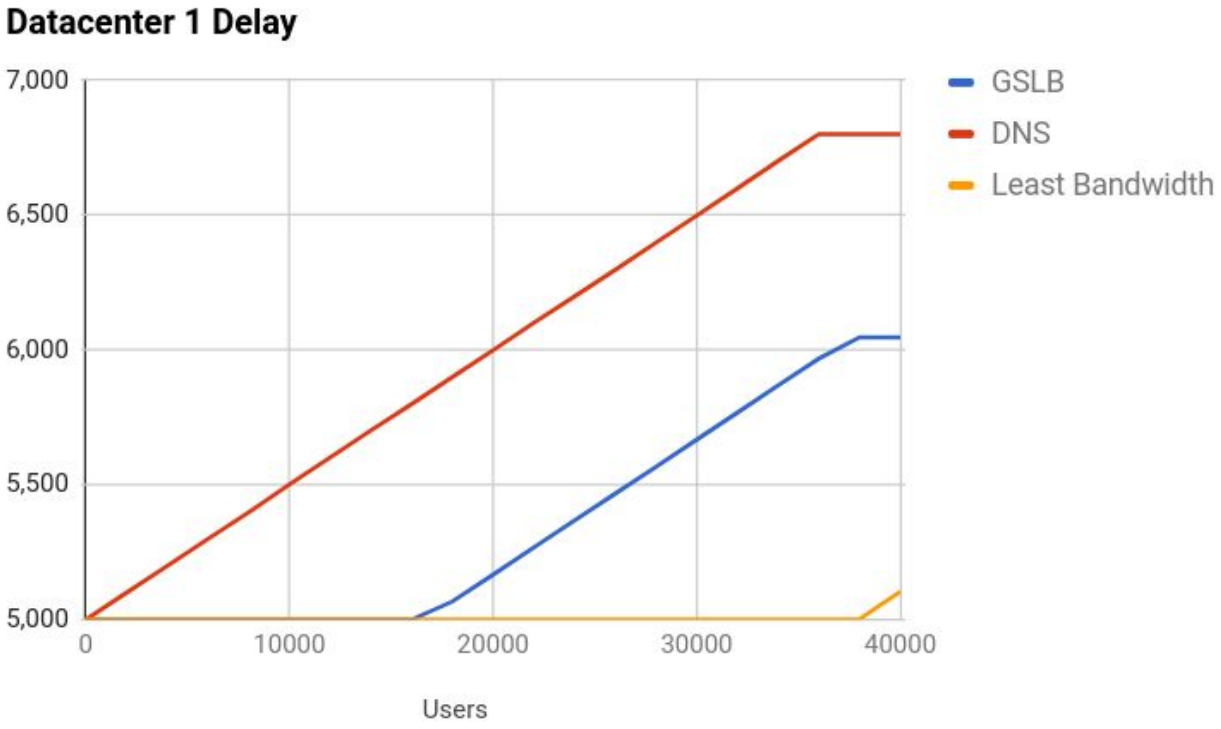


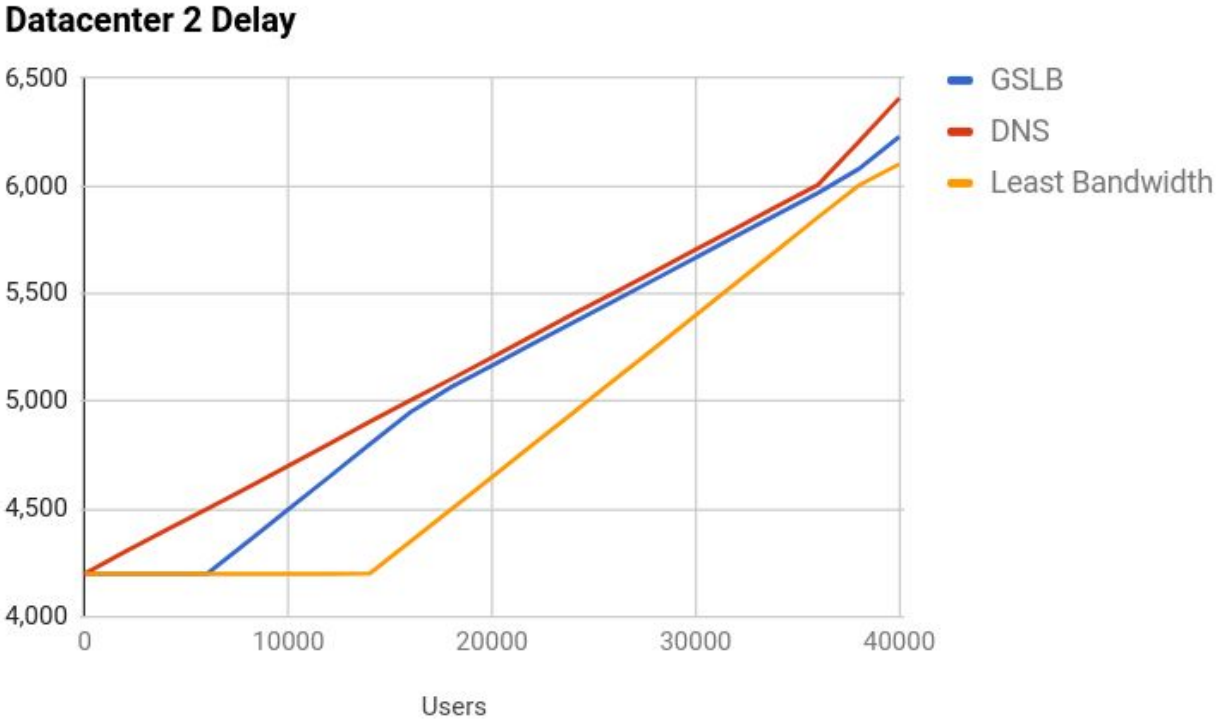
Согласно полученным результатам, DNS балансировка распределяет клиентов равномерно по всем трем дата-центрам, независимо от их текущей нагрузки. Least Bandwidth и GSLB методы распределяют нагрузку более равномерно, ориентируясь на текущую загрузку по трафику и ресурсам соответственно. Однако в случае Least Bandwidth заполнение наиболее загруженного и приоритетного первого дата-центра происходит быстрее, в то время как GSLB балансировка распределяет нагрузку по вторичным дата-центрам, используя мощности приоритетного дата-центра в последнюю очередь.

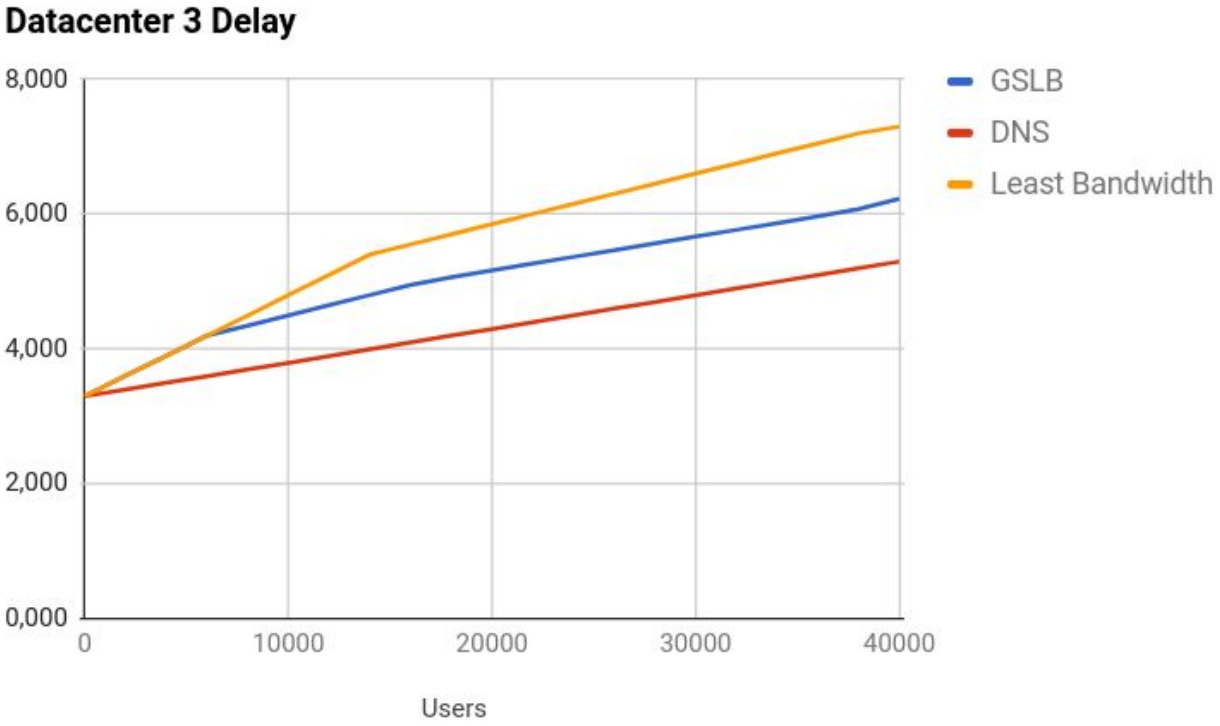
4.2 Приоритет по наименьшим задержкам использования

В данном примере будет рассмотрен частный случай при выставлении приоритета балансировки, в зависимости от задержки времени доступа и использования ресурса сервера клиентами. Начальные условия зададим аналогичными, представленными в примере 1.

Целью данного частного случая будет наиболее оптимальное распределение клиентов с целью снизить задержки при доступе и использовании ресурса. Сравнение также будет происходить с альтернативными алгоритмами балансировки, указанными выше.







Полученные результаты показывают, что при GSLB балансировке удалось добиться приблизительно одинаковых задержек при работе со всеми дата-центрами, в то время как при DNS и Least Bandwidth балансировке присутствует значительный разброс. В конечном счете, данный разброс может

привести к ситуации, когда при доступе к одному дата-центру клиенты не будут испытывать никаких проблем в работе, в то время работа клиентов, направленных на другой дата-центр, может быть полностью невозможна в силу существенных задержек.

5. Реализация

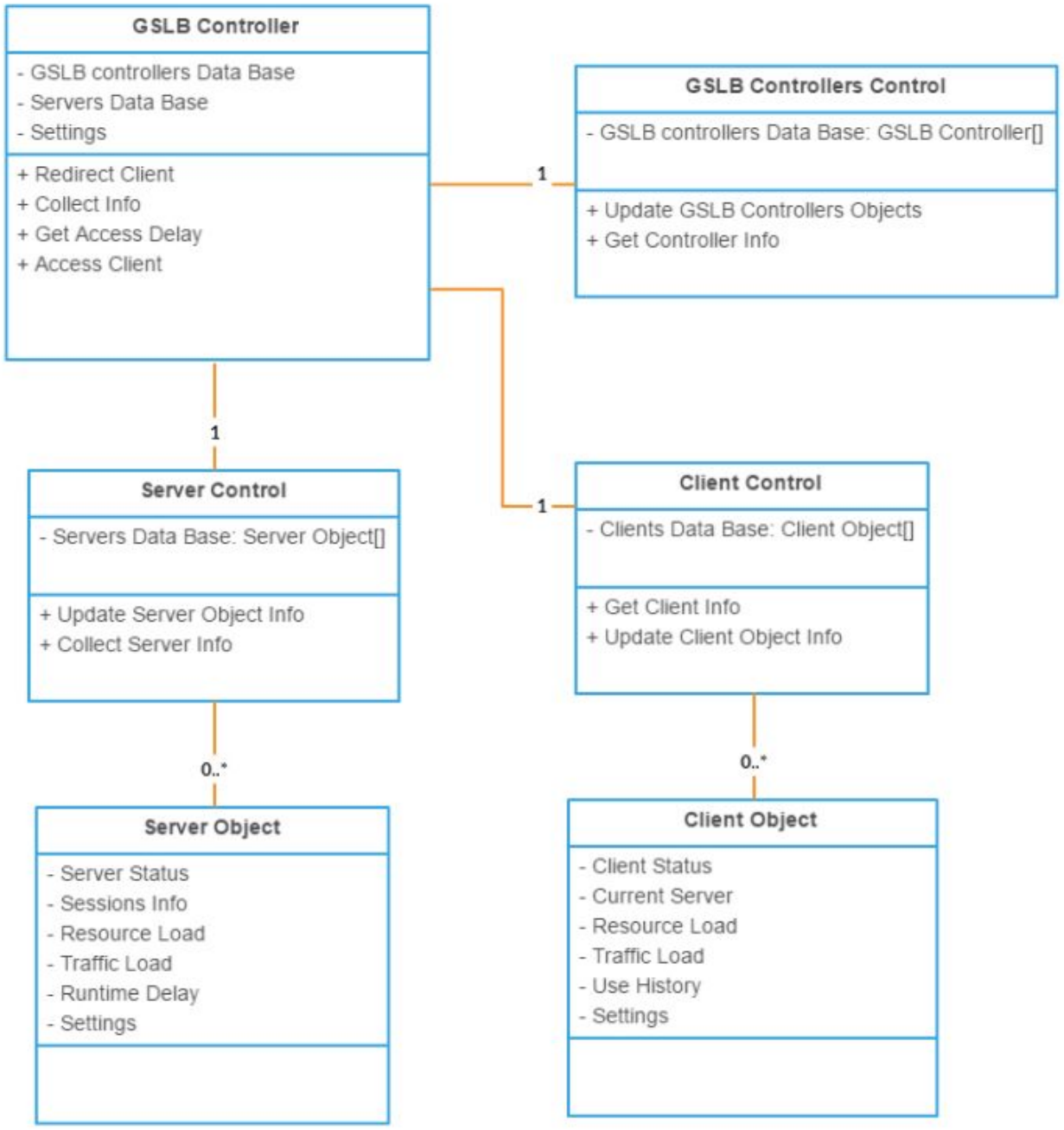
Подробное описание низкоуровневых операций по принятию решений и перенаправлению клиентов было приведено в главе **Принципы балансировки GSLB**. В данной же главе будет описана общая архитектура реализации и описание основных компонентов модуля балансировки.

В качестве основного инструмента в создании реализации был использован Windows Communication Foundation. WCF являет собой software development kit (SDK) для разработки и развертывания услуг на Windows и предоставляет среду для выполнения услуг, значительно упрощающая построение инфраструктуры. Microsoft WCF предоставляет набор стандартов, определяющих взаимодействия служб, преобразование типов, маршалинг и управление различными протоколами. WCF обладает множеством полезных качеств, такими как поддержка хостинга, возможность управление экземплярами служб, асинхронные вызовы, надежность, управления транзакциями, безопасность, поддержка работы с большим количеством протоколов. WCF также является расширяемым, тем самым обеспечивая возможность улучшения базового функционала. Ниже будет представлена диаграмма классов для данной реализации.

На каждого пользователя приходится по одному объекту типа Client Object. В нем содержится базовое описание данного клиента и информация о текущем статусе клиента, занимаемых ресурсах, трафике, принадлежности к серверу на данном дата-центре, история использования и настройки для выбранного клиента. К истории клиента могут относится дата последнего использования и текущее время активности сеанса. В качестве настроек выступают установленное время жизни данного объекта, т.е. как долго данные

пользователя будут хранится со времени последнего использования, и параметры использования ресурсов.

На каждый сервер приходится по одному объекту типа Server Object. Данный класс содержит в качестве полей текущие характеристики сервиса, информацию о настройках (лимиты использования, кол-во сессий и др.), текущую задержку в использовании, текущее кол-во сессий, как активных, так и пассивных, и статус сервера.



В качестве основного класса выступает класс GSLB контроллера. Параметрами в нем выступают объект клиентской базы данных, работающих с данным дата-центром, объект серверной базы данных для данного дата-центра, база данных альтернативных GSLB контроллеров и настройки использования. Данный класс отвечает за сбор информации, по результатам которого принимается решение о подключении пользователя к текущему дата-центру и к какому серверу на этой ферме, либо о перенаправлении пользователя на другую.

Классы баз данных отвечают за мониторинг текущего состояния соответствующих объектов, обновлением самих баз данных, и сборе информации для GSLB контроллера с целью дальнейшей балансировки. Класс управления альтернативными GSLB контроллерами отвечает за добавление и удаление дополнительных GSLB контроллеров в облако, сбор метрик с этих GSLB контроллеров и проверку их статуса и поиск альтернативного центра предоставления услуг. Полученная информация используется в случае невозможности предоставления услуг пользователю на этой ф

Класс управления серверными объектами отвечает за базу данных серверов и выполняет сбор информации о их текущем состоянии в пределах фермы, а также добавление, удаление и настройку этих серверов. Полученная информация предоставляется GSLB контроллеру для дальнейшей обработки.

Класс управления клиентами содержит в себе базу данных о текущих клиентах и отвечает за добавление, удаление и обновление клиентских объектов.

Каждый из экземпляров классов управления серверами и GSLB контроллерами производит сбор метрик самостоятельно с выбранным интервалом. В случае, если неудачи операции сбора на каком-либо объекте, либо, если достигнут предел использования по тому или иному ресурсу этого объекта, данный объект считается недоступным для последующего

использования до момента устранения неисправности или до момента снижения загрузки. Обновление информации о клиентских объектах происходит посредством уведомления при подключении/отключении пользователя, а также в процессе использования с целью проверки валидности клиентского объекта: недоступен или перегружен.

6. Результаты

В результате данной работы был произведен анализ существующих проблем и рассмотрено множество как частных решений, так и общедоступной информации по глобальной балансировки нагрузки серверов, включая патенты. По результатам анализа было предложен обобщенный алгоритм, описан принцип его работы и смоделировано множество случаев его поведения при различных условиях. При аналогичных условиях была также смоделировано поведения базовых алгоритмов балансировки и произведено сравнение с предложенным алгоритмом, по результатам которого удалось добиться более умного распределения нагрузки на сервера в зависимости от различных условий и приоритетов балансировки. На базе теоретического описания предложенного алгоритма был создан модуль балансировки и представлена его базовая архитектура, по которой создавалась реализация.

7. Перспективы

К дальнейшим планам развития полученного алгоритма относится построение его строгого математического обоснования эффективности, а также включения в эту модель принципов балансировки на уровне отдельных компонентов, таких как балансировка ресурсов сервера в пределах одного кластера/сервера и поиск оптимального пути соединения пользователя с конечным сервисом в сети Интернет, а также совершенствование модели поведения алгоритма при рассмотрение относительного недостатка ресурсом при доступе к конечному сервису или службе. Практическая реализация полученного алгоритма, в дальнейшем, может развиваться как самостоятельное решение.

Източници

1. Global Server Load Balancing, US 8024441 B2, 20.09.2011, Sunanda Lakshmi Kommula, Ivy Pei-Shan Hsu, Rajkumar Jalan, David Chun Ying Cheung
2. DNS Load Balancing, www.radware.com/resources/dns_load_balancing.aspx
3. DNS Load Balancing – Comparison of 4 services, June 6, 2014, Dmitriy Akulov, www.maxcdn.com/blog/dns-load-balancing-comparison-4-services/
4. Manage GSLB via the API, https://wiki.appnexus.com/display/documentation/Managing+Global+Server+Load+Balancing
5. Global Server Load Balancing Scalability, High Availability and Performance for Distributed Networks http://www.arraynetworks.co.jp/ufiles/resources/WP-APV-Global-Server-Load-Balancing-Apr-2011-Rev-A.pdf
6. Citrix NetScaler… The basics continued, part five. Global Server Load Balancing! October 20, 2015, Bas van Kaam, http://www.basvankaam.com/2015/10/20/citrix-netscaler-the-basics-continued-part-five-global-server-load-balancing/
7. Global Server Load Balancing (GSLB) Concepts, http://alpha64.subrigo.net/a10/A10%20GSLB%20Concepts%20and%20Policy_092214.pdf
8. Global Server Load Balancing & Domain Name System (DNS) Management, https://www.veber.co.uk/global-server-load-balancing-domain-name-system-dns-management/
9. Incapsula, DNS Load Balancing and Failover, https://www.incapsula.com/load-balancing/dns-load-balancing-failover.html
10. DNS Load Balancing and Using Multiple Load Balancers in the Cloud, October 23, 2012, Patrick McClory,

http://www.rightscale.com/blog/enterprise-cloud-strategies/dns-load-balancing-and-using-multiple-load-balancers-cloud

11. Eukhost, Global Server Load Balancing, January 5, 2011 http://www.eukhost.com/kb/global-server-load-balancing/
12. Cisco GSS CLI-Based Global Server Load-Balancing Configuration Guide, http://www.cisco.com/c/en/us/td/docs/app_ntwk_services/data_center_app_services/gss4400series/v1-3/configuration/cli/gslb/guide/cli_gslb/Intro.html
13. Global Server Load Balancing (GSLB), http://www.carlstalhood.com/global-server-load-balancing/
14. Global Server Load Balancing (GSLB) for Enterprise, August 12, 2015, Chris Yoo http://www.netmanias.com/en/post/blog/7637/dns-gslb-network-protocol/global-server-load-balancing-gslb-for-enterprise-part-1-concept-and-service-logic
15. Infoblox Introduces the First Solution Adding Global Server Load Balancing to Enterprise-grade DNS Appliances January 7, 2015, https://www.infoblox.com/company/news-events/press-releases/infoblox-introduces-first-solution-adding-global-server-load-balancing-enterprise-grade-dns-appliances/