УДК 681.3.06

АНАЛИЗ МЕТОДОВ УВЕЛИЧЕНИЯ ПРОИЗВОДИТЕЛЬНОСТИ ВЕБ-ПРИЛОЖЕНИЙ

А.Н. ЩИТЛЯК

Белорусский государственный университет информатики и радиоэлектроники, Республика Беларусь

Поступила в редакцию 31 октября 2018

Аннотация. Исследованы методы увеличения производительности клиент-серверных приложений. Проведены программные эксперименты, направленные на оценку объема трафика, создаваемого приложениями, количества запросов, обрабатываемых сервером, а также требуемого объема оперативной памяти. Обосновано, что наиболее предпочтительным методом увеличения производительности приложений является вариант, связанный с кэшированием динамических страниц и обработкой статических файлов на стороне клиента.

Ключевые слова: веб-приложение, производительность, быстродействие, кэширование.

Введение

За последние годы Интернет стал важной платформой для получения данных и запуска веб-приложений. Веб-приложения – клиент-серверные приложения, в которых клиент взаимодействует с сервером при помощи браузера. Такие приложения легкодоступны независимо от места, с которого к ним обращаются. Поэтому в настоящее время пользователи все чаще отдают предпочтение веб-приложениям. Это обуславливает повышение требований к этим приложениям, а также к их платформам и средам.

Среда веб-приложений обладает рядом особенностей, среди которых можо выделить следующие:

1. Взаимодействие пользователя и сервера происходит посредством кратковременных и часто повторяющихся запросов.

2. По глобальным телекоммуникационным сетям передаются дополнительные файлы, содержащие в себе медиаконтент, исполняемый код, справочную информацию, что способствует возникновению множества параллельных запросов, повышающих загрузку веб-серверов и каналов связи.

3. Пропускная способность каналов связи между пользователями и сервером может быть ограничена. Этот факт значительно увеличивает время обслуживания каждого клиента по сравнению со временем выполнения программных сценариев.

4. Количество одновременно работающих пользователей может быть очень большим, что требует принятия специфических мер при реализации серверных приложений.

Многие компании затрачивают большое количество времени, усилий и ресурсов на повышение производительности разрабатываемых или используемых ими веб-приложений. Сюда относятся сокращение общего объема страницы и количества объектов на ней, оптимизация кода или увеличение пропускной способности Интернет-соединения. Тем не менее, производительность веб-приложения зависит от полной цепи его доставки, где должно быть оптимизировано каждое звено, начиная с веб-сервера и сетей, соединяющих его с конечным пользователем, и заканчивая различными браузерами и клиентским ПО конечных пользователей.

Таким образом, актуально исследование способов и методов рационального использования аппаратных ресурсов путем разработки и внедрения дополнительных программных средств, модификаций существующего программного кода, улучшающих качество

информационно-вычислительной среды (производительность, требуемый объем оперативной памяти, загруженность каналов связи и т. д.).

Методы увеличения производительности веб-приложений

Объем оперативной памяти – основной ограничивающий фактор для увеличения количества одновременно обслуживаемых пользователей. Это обусловлено тем, что все процессы делят общее ограниченное адресное пространство. При этом скорость выполнения других процессов существенно замедляется, поскольку для хранения swap-файла, эмулирующего недостающую оперативную память, используется жесткий диск.

Значительное влияние на производительность веб-приложений оказывают пропускная способность и время отклика канала связи. Низкая пропускная способность канала связи ведет к увеличению времени доставки страниц пользователям. Как следствие, наблюдается увеличение количества процессов, одновременно находящихся в оперативной памяти и занятых передачей сформированных данных.

Существует множество способов, позволяющих улучшить производительность серверных приложений. В ходе работы исследовались следующие методы увеличения производительности:

1) кэширование данных на стороне сервера;

2) предварительная генерация содержимого веб-страниц в статические файлы;

3) использование многоуровневой архитектуры клиент-сервер [1].

Схема проведения исследований

В ходе экспериментов были задействованы следующие программные средства: Apache 2.4 [2] в качестве веб-сервера, PHP 5.2.5 [3] как язык выполнения серверных сценариев, MySQL 5.6 [4] в качестве сервера системы управления базой данных.

Характеристики тестовой страницы, сформированной PHP-сценарием:

1) 2 js-файла, пять графических файлов, ссылки на два ccs-файла;

2) 30 произвольных записей из таблицы базы данных, содержащей 10000 записей; каждая запись включала в себя ссылку на графический файл, заголовок и случайный текст из 2000 символов;

3) при каждой загрузке страницы создавалась или изменялась одна запись из таблицы базы данных, имеющая ссылку на графический файл и текст.

Технические характеристики: процессор Intel Core i3 2,5 ГГц, объем оперативной памяти 4 ГБ, жесткий диск объемом 500 ГБ (частота вращения шпинделя – 7200 об/мин).

В качестве программы для имитации клиентских обращений была использована программа SIEGE. Она предоставляет разработчикам возможность проверить ресурсоемкость своего кода в условиях, максимально приближенных к реальным. Также SIEGE позволяет имитировать обращения к сайту сразу нескольких пользователей. Количество запросов, отправленных к ресурсу, рассчитывается из общего количества пользователей и количества их обращений к серверу. Например, 20 пользователей, обратившись по 50 раз, создают в общей сложности 1000 запросов. Результат, выводимый программой после тестирования, включает в себя время, затраченное на проверку, общее количество переданной информации (включая заголовки), среднее время ответа сервера, его пропускную способность и число запросов, на которые пришел ответ с кодом 200. Эти данные формируются и выдаются при каждой проверке [5]. Также программа SIEGE позволяет имитировать одновременное обращение к серверу нескольких пользователей.

Варианты повышения производительности

Кэширование данных на стороне сервера. Кэширование позволяет увеличивать производительность веб-приложений за счет использования сохраненных ранее данных: ответов на сетевые запросы или результатов вычислений. Благодаря кэшу, при очередном обращении клиента за одними и теми же данными, сервер может обслуживать запросы быстрее.

Кэширование – эффективный архитектурный паттерн, т. к. большинство программ часто обращаются к одним и тем же данным и инструкциям.

Суть кэширования на стороне сервера заключается в том, чтобы записать все, что происходит на сервере, в файл, сохранить его и при последующем обращении этого либо любого другого пользователя к этой странице предоставить ему статичную копию. В результате происходит не только ускорение загрузки страниц, но и снижение нагрузки на сервер и базу данных.

В ходе эксперимента осуществлялось кэширование результатов запросов к базе данных в файлах. В результате исследований количество обращений к базе данных сократилось на 70 %. Данные кэша уничтожались при изменении записей базы данных и не имели времени актуальности. Весь кэш очищался при добавлении или удалении одной или нескольких записей. При изменении одной или нескольких записей БД удалялись только страницы, содержащие изменяемые записи. Если серверный скрипт при обращении в кэш не находил нужной ему записи, то производился запрос к базе данных и его результаты сохранялись в кэше.

Предварительная генерация содержимого в статические файлы. Веб-запросы пользователей к статическим страницам обрабатываются быстрее и требуют меньше накладных расходов (необходимость использования базы данных, объем памяти), чем запросы к динамически формируемым серверными скриптами страницам. В ходе эксперимента реализовывалось обращение к статическим документам *.js. Если такие файлы отсутствовали, то они формировались с помощью специального обработчика страницы ошибок (HTTP-код 404 – Документ не найден).

Использование многоуровневой архитектуры клиент-сервер. Многоуровневая архитектура клиент-сервер – разновидность архитектуры, в которой функция обработки данных вынесена на один или несколько отдельных серверов. Это позволяет разделить функции хранения, обработки и представления данных для более эффективного использования возможностей серверов и клиентов. Достоинствами такой архитектуры являются масштабируемость, конфигурируемость, высокая безопасность и надежность.

При использовании многоуровневой архитектуры клиент-сервер все запросы пользователя принимает клиент. Если пользователь обращается к статическому файлу, то запрос клиентом обрабатывается самостоятельно. При использовании динамически формируемых страниц, клиент формирует запросы к Apache-серверу и, получив от него данные веб-страниц, возвращает их пользователю. Этот подход значительно экономит процессорные ресурсы, поскольку клиент большую часть трудоемких операций (отправка файлов, считывание данных с диска) осуществляет асинхронно с помощью функций ядра операционной системы, получая только сигналы об их завершении.

Подход, в ходе которого осуществлялось кэширование данных на стороне сервера, оказался самым медленным из-за высокой нагрузки на базу данных. Наилучшим оказался подход, заключающийся в предварительной генерации содержимого в статические файлы. Это обусловлено тем, что при использовании данного метода обеспечивается значительная экономия передаваемого трафика и высокое быстродействие. При использовании многоуровневой архитектуры клиент-сервер были достигнуты хорошие показатели при небольшом количестве пользователей, однако при повышении нагрузки быстродействие снизилось из-за частых операций перезаписи файлов [6].

Заключение

Проведены исследования быстродействия веб-приложений при различных вариантах организации взаимодействия его пользователя с клиентом, клиента с сервером и статическими файлами. Показано, что для практического применения наиболее предпочтительным является вариант реализации веб-приложений с кэшированием динамических страниц на стороне клиента и обработкой статических файлов на стороне клиента.

ANALYSIS OF METHODS TO INCREASE PERFORMANCE OF WEB-APPLICATIONS

A.N. SHCHITLYAK

Abstract. The methods for increasing of server applications performance are researched. Program experiments for estimation of traffic volumes formed by applications, quantity of requests processed by the server and demanded volume of operative memory are carried out. It is shown that the most preferable variant, from the point of view of practical application, is the variant with processing of static files on the client side.

Keywords: web-application, performance, speed, caching.

Список литературы

1. Веллинг С., Томсон Л. Разработка клиент-серверного приложения. М., 2012.
2. Уэнрайт П. Apache для профессионалов. СПб, 2010.
3. PHP: Hypertext preprocessor. [Электронный ресурс]. http://php.net/releases/5_2_5.php (дата обращения: 31.10.2018).
4. Дюбау П. MySQL. М., 2014.
5. Siege – утилита для нагрузочного тестирования веб-серверов. [Электронный ресурс]. https://habr.com/post/65128/ (дата обращения: 31.10.2018).
6. Ботыгин И.А., Каликин К.А. Исследование методов увеличения производительности Web-приложений. Томск, 2008.