УДК 004.75

ИССЛЕДОВАНИЕ И РАЗРАБОТКА РАСПРЕДЕЛЕННЫХ СЕРВЕРНЫХ СИСТЕМ

Чернобай Ю.А., Теплинский С.В.
Донецкий национальный технический университет

Общая постановка проблемы

Стремительное разрастание Всемирной паутины и связанное с этим увеличения объемов трафика продолжают тревожить специалистов. Web-сервера стали не только хранилищем текстовой и графической информации, но и местом гигантских залежей видео- и аудиоматериалов, а также средством проведения масштабных коммерческих операций. На первый план выходит задание обслуживания запросов за гарантированное время, которое неминуемо требует усовершенствованных технических, алгоритмических и программных средств [7].

Одним из наиболее удачных способов увеличения производительности веб-сервера есть создание распределенной системы. В этом случае вместо одного компьютера запросы пользователей обрабатывают сразу несколько машин [6].

Решение поставленных задач в статье

Для реализации высокопродуктивного и надежного веб-сервера используют распределенный серверные системы. Распределенные серверные системы являются набором серверов – это продублированные ресурсы для одновременного предоставления сервисов многим клиентам. Входные запросы могут быть распределены по серверам согласно определенным стратегиям распределения загрузки и потому эти запросы могут быть быстро обработаны. Распределенные серверные системы могут быть организованни разными способами:

- Они могут быть интегрированы в кластер веб-серверов, соединенных через локальную вычислительную сеть (ЛВС) чтобы действовать как один мощный веб-сервер.
- Они могут использоваться в различных географических местах через глобальную вычислительную сеть (ГВС).

Распределенные серверные системы обладают высокой степенью масштабируемости. Количество их может быть увеличено простым добавлением в нового сервера в локальную сеть.

Между тем, организация распределенного Web-сервера остаточно сложная задача, требующая детальной проработки ряда вопросов:

- анализа сетевого трафика и ожидаемой загрузки сервера;
- выбора архитектуры распределенной системы;
- маршрутизации пакетов внутри системы;
- функциональности распределителей нагрузки;
- дисциплины обслуживания запросов и выбора сервера;
- распределения хранимой информации [7].

При проектировании распределенных систем, которое имеет много общего с проектированием ПО в общем, все же следует учитывать некоторые специфические особенности.

Существует шесть основных характеристик распределенных систем:

1. ***Совместное использование ресурсов***. Распределенные системы допускают совместное использование как аппаратных (жестких дисков, принтеров), так и программных (файлов, компиляторов) ресурсов.
2. ***Открытость***. Это возможность расширения системы путем добавления новых ресурсов.
3. ***Параллельность***. В распределенных системах несколько процессов могут одновременно выполнятся на разных компьютерах в сети. Эти процессы могут взаимодействовать во время их выполнения.
4. ***Масштабируемость***. Под масштабируемостью понимается возможность добавления новых свойств и методов.
5. ***Отказоустойчивость***. Наличие нескольких компьютеров позволяет дублирование информации и устойчивость к некоторым аппаратным и программным ошибкам. Распределенные системы в случае ошибки могут поддерживать частичную функциональность. Полный сбой в работе системы происходит только при сетевых ошибках.
6. ***Прозрачность***. Пользователям предоставляется полный доступ к ресурсам в системе, в то же время от них скрыта информация о распределении ресурсов по системе.

Однако распределенные системы обладают и рядом недостатков.

1. ***Сложность***. Намного труднее понять и оценить свойства распределенных систем в целом, их сложнее проектировать, тестировать и обслуживать. Также производительность системы зависит от скорости работы сети, а не отдельных процессоров. Перераспределение ресурсов может существенно изменить скорость работы системы.
2. ***Безопасность***. Обычно доступ к системе можно получить с нескольких разных машин, сообщения в сети могут просматриваться и перехватываться. Поэтому в распределенной системе намного труднее поддерживать безопасность.
3. ***Управляемость***. Система может состоять из разнотипных компьютеров, на которых могут быть установлены различные версии операционных систем. Ошибки на одной машине могут распространиться непредсказуемым образом на другие машины.
4. ***Непредсказуемость***. Реакция распределенных систем на некоторые события непредсказуема и зависит от полной загрузки системы, ее организации и сетевой нагрузки. Так как эти параметры могут постоянно изменятся, поэтому время ответа на запрос может существенно отличаться от времени.

Использование многоуровневой архитектуры “клиент-сервер” при создании распределенных серверных систем.

Многоуровневая архитектура клиент-сервер — разновидность архитектуры клиент-сервер, 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.[3]

Многоуровневые архитектуры обычно состоят из четырех уровней (рисунок 3), где в сети сервер отвечает за обработку соединение между клиентом браузером и сервером приложений.[5]

При использовании многоуровневой архитектуры клиент-сервер при создании распределенных серверных систем часть нагрузки, которую выполняет главный сервер, можно перенести на промежуточные серверы. Организация очереди запросов между главным и промежуточными серверами не дает сильно нагружать главный сервер, а

переложенная на промежуточные серверы функция шифровки/дешифрации проверки целостности пакетов, а также адекватность пересылаемых данных позволяет разгрузить процессорные ресурсы главного сервера, тем самым сократив время обработки каждого запроса, и увеличить отказоустойчивость главного сервера, защитив его от сетевых атак.

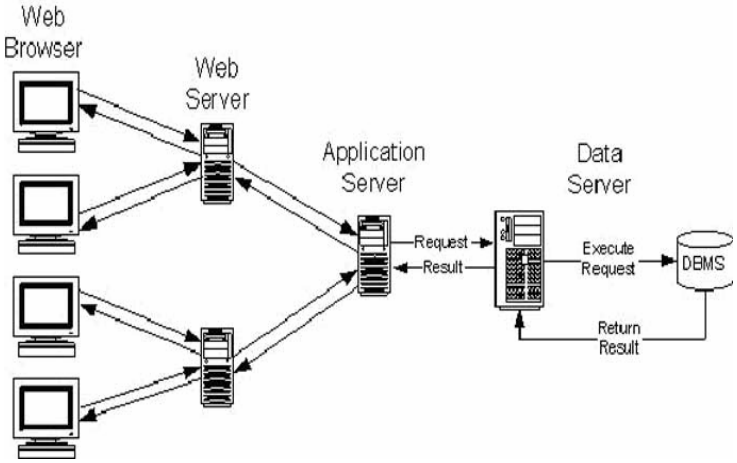

Рисунок 1 – Многоуровневая архитектура «клиент-сервер»

Беря за основу четырехуровневую архитектуру клиент-сервер был разработан комплекс программ составляющих в группе распределенную серверную систему. Главный сервер соединен с базой данных через сервер MsSQL, к главному серверу подключены промежуточные сервера, к которым подключаются клиентские программы. Главным преимуществом такой реализации сервера является его масштабируемость: в случае большой загруженности системы её производительность может быть увеличена подключением новых промежуточных серверов.

Однако в отличие от стандартной организации многоуровневой клиент серверной архитектуры оставлена возможность подключения клиентов к главному серверу. Если запущен только главный сервер, то клиентские программы соединяются непосредственно с ним. Этот способ подключения целесообразно использовать в случае, если невелико количество одновременно подключенных клиентов. Также, когда клиент соединен напрямую с главным сервером, сокращается общее время обмена пакетами между клиентом и сервером.

Если к главному серверу присоединяются промежуточные серверы, то новые клиенты, распределяются равномерно между этими промежуточными серверами. Промежуточные серверы выполняют только проверки целостности пакетов, адекватность пакетов и операции шифровки дешифрации. А операции обращения к базе данных для всех клиентов выполняется все равно на одном главном сервере. В случае если отправляемые пакеты будут иметь одинаковую длину, и пакеты от каждого клиента будут поступать периодически, как это происходит в онлайн играх, то манипуляции, которые выполняются над пакетами на промежуточных серверах будут занимать одинаковое количество времени. А это значит, что для расчета загруженности каждого промежуточного сервера необходимо знать текущее количество подключенных клиентов к этому серверу.

Схема соединения клиентов с сервером с использованием промежуточных серверов изображена на рис. 2.

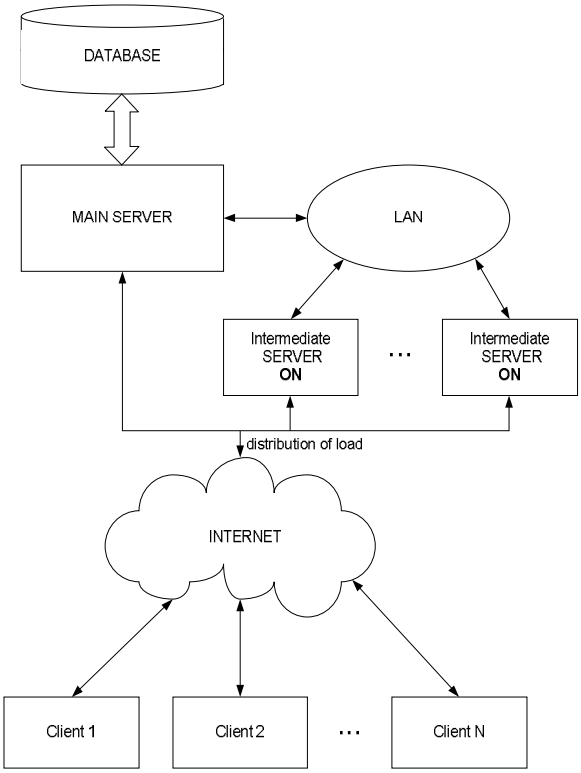

Рисунок 2 – Взаимодействие главного сервера с клиентами с участием промежуточных серверов

Исходя из этого, был разработан алгоритм балансировки, который заключается в том, что каждый промежуточный сервер отправляет на главный сервер информацию о количестве клиентов каждый раз, когда происходит подключение или разъединение клиентской программы с этим сервером. Главный сервер получает и хранит эту информацию для каждого промежуточного сервера.

При подключении новый клиент отправляет запрос к главному серверу. Главный сервер проверяет загруженность всех присоединенных промежуточных серверов и отправляет информацию о наименее нагруженном сервере клиенту. После получения этой информации клиент соединяется с этим сервером.

Такой алгоритм балансировки имеет как преимущества, так и недостатки. Алгоритм не учитывает разность вычислительной мощности и пропускной способности сети серверов, однако при использовании данного алгоритма нет необходимости постоянно проводить мониторинг входящего, исходящего трафиков и загруженности процессора промежуточных серверов, что экономит время обмена информацией. При равномерном подключении клиентов ко всем промежуточным серверам можно предположить, что нагрузка на них будет равномерно распределена даже, если все клиенты одновременно обратятся к серверу. Еще одно преимущество данного алгоритма балансировки его простота реализации.

Предложенная распределенная система повышает уровень отказоустойчивости системы. Если промежуточный сервер выключается, его клиенты отправляют запрос главному серверу, а он перенаправляет их на других работающие промежуточные сервера. Однако в случае отключения главного сервера система перестает функционировать. Также промежуточные серверы выполняют функцию защиты главного сервера от сетевых атак, так как все запросы клиентов поступают сначала к ним.

Выводы

Использование многоуровневой архитектуры клиент-сервер в распределенных серверных системах позволит создавать быстрые, защищенные от атак системы, которые легко масштабируются.

Используя для своей работы локальную сеть из обычных компьютеров пользователей, она позволяет экономить средства на приобретении серверов большой мощности и использование дорогого лицензионного программного обеспечения.

Эти системы могут применятся в разных сферах Интернет деятельности, которые нуждаются в клиент-серверной организации взаимодействия с клиентами, и охватывают развлекательные и социальные сферы работы Интернета.

Литература

[1] Маркин А.В. Построение запросов и програмирование на SQL / А.В. Маркин // Рязань 2008. 312с.

[2] Шаньгин В. Компьютерная безопасность информационных систем. / В. Шаньгин // 2008 416с.

[3] Клиент-сервер. Википедия свободная энциклопедия [Электронный ресурс] Режим доступа: http://ru.wikipedia.org/wiki/Клиент-сервер.

[4] Шокин Ю.И., Федотов А.М. Распределенные информационные системы [Электронный ресурс] Режим доступа: http://old.ict.nsc.ru/win/gis/lib/publ/inf_sys.html.

[5] Крамек Э. Перевод: Чернобай Ю.А. Введение в архитектуру клиент-серверных систем [Электронный ресурс] Режим доступа: http://masters.donntu.edu.ua/2010/fknt/chernobay/library/translate.htm

[6] Распределенный сервер и круговой DNS [Электронный ресурс] Режим доступа: http://www.compdoc.ru/network/dns/circular_dns/

[7] Труб И., Алгоритмическое обеспечение распределенных Web-серверов [Электронный ресурс] Режим доступа: http://citforum.ru/cfin/alg_ob_web_ser/

[8] Балансировка нагрузки в распределенных системах [Электронный ресурс] Режим доступа: http://www.intuit.ru/department/algorithms/distrsa/9/1.html