УДК 519.2:004.7

И.Н. Иванисенко, Л.О. Кириченко, Т.А. Радивилова

ОБ ОДНОМ МЕТОДЕ РАСПРЕДЕЛЕНИЯ НАГРУЗКИ С УЧЕТОМ ФРАКТАЛЬНЫХ СВОЙСТВ ТРАФИКА

Аннотация. Предложено математическое описание распределенной системы балансировки нагрузки и динамический метод распределения нагрузки, учитывающий мультифрактальные свойства сетевого трафика, на основании которых пересчитывается распределение потоков.

Ключевые слова: самоподобный и мультифрактальный трафик, балансировка нагрузки, распределенные системы

Постановка задачи

Современные сети связи характеризуются значительными слабо предсказуемыми колебаниями нагрузки. Трафик современной глобальной сети обнаруживает экспоненциальный рост, происходят значительные структурные изменения, все более ощутимыми становятся плохо предсказуемые колебания нагрузок. Процессы конвергенции сетей связи привели к доминированию протокола IP в качестве универсального для всех видов передаваемых данных. Однако отсутствие встроенных механизмов инжиниринга трафика ставит вопрос о необходимости разработки методов, позволяющих более эффективно использовать возможности существующей сетевой инфраструктуры, но не требующих изменения основ функционирования глобальной сети. Одним из наиболее перспективных вариантов решения указанных проблем на сегодняшний день являются динамические механизмы балансировки трафика, вызывающие пристальный интерес научного сообщества. Применение данных методов, реагирующих на изменения сетевых нагрузок в режиме, близком к реальному времени, для локального смягчения временных перегрузок в глобальной сети позволит избавиться от недостатков присущих существующим сетям [1-4].

Экспериментальные и численные исследования, проведенные в последние десятилетия, свидетельствуют, что трафик во многих му-

льтисервисных сетях имеет самоподобные свойства. Самоподобный трафик вызывает значительные задержки и потери пакетов, даже если суммарная интенсивность всех потоков далека от максимально допустимых значений. Самоподобные свойства информационных потоков обнаружены во многих локальных и глобальных телекоммуникационных сетях [5,6]. В связи с вышеизложенным начали активно исследоваться механизмы повышения качества обслуживания и методов управления трафиком в мультисервисных сетях, функционирующих в условиях самоподобного и мультифрактального трафика [7].

Целью данной работы является математическое описание распределенной системы балансировки нагрузки и разработка динамического метода распределения нагрузки в компьютерных системах на основе мониторинга загруженности серверов, с учетом самоподобной структуры трафика.

Самоподобный и мультифрактальный трафик

Самоподобие случайных процессов заключается в сохранении статистических характеристик при изменении масштаба времени и характеризуется показателем Херста H, который является степенью самоподобия. Стохастический процесс X(t) является статистически самоподобным если процесс $a^{-H}X(at)$ обладает теми же статистическими характеристиками второго порядка, что и X(t). Параметр H, называемый параметром Херста, представляет собой меру самоподобия стохастического процесса. Начальные моменты самоподобного случайного процесса можно выразить как $M\left[|X(t)|^q\right] = C(q)\cdot t^{qH}$, где величина $C(q) = M\left[|X(1)|^q\right]$. Для мультифрактальных процессов выполняется отношение $M\left[|X(t)|^q\right] = c(q)\cdot t^{qh(q)}$, где $c(q)$ – некоторая детерминированная функция; $h(q)$ – обобщенный показатель Херста, являющийся в общем случае нелинейной функций. Значение $h(q)$ при $q = 2$ совпадает со значением степени самоподобия H [6,8].

Самоподобный трафик имеет особую структуру, которая сохраняется на многих масштабах – в реализации всегда присутствует некоторое количество очень больших выбросов при относительно небольшом среднем уровне трафика. Мультифрактальный трафик определяется как расширение самоподобного трафика за счет учета масшта-

 ISSN 1562-9945

бируемых свойств статистических характеристик второго и выше порядков.

Как характеристику неоднородности мультифрактального потока данных в работе предложено считать диапазон обобщенного показателя Херста $\Delta h = h(q_{\min}) - h(q_{\max})$. Для монофрактальных процессов обобщенный показатель Херста не зависит от параметра q и является прмой линией: $h(q) = H$, $\Delta h = 0$. Чем больше неоднорость процесса, т.е. большее число выбросов присутствует в трафике, тем больше диапазон Δh.

Система балансировки нагрузки

Рассматриваемая информационная система состоит из группы серверов и балансировщика нагрузки. Система балансировки нагрузки, представленная на рис. 1, построена на основе подсистемы балансировки нагрузки и подсистемы управления и мониторинга, которые тесно взаимодействуют друг с другом.

- Подсистема балансировки нагрузки: алгоритм балансировки нагрузки, информация о текущем состоянии системы, гибкие настройки QoS, динамическое распределение трафика по различным каналам связи и узлам в зависимости от их текущего состояния, степени загрузки, административных политик балансировки нагрузки.

- Подсистема управления и мониторинга: сбор и анализ статистики о текущем состоянии системы, нахождение мультифрактальных свойств входящего потока данных, расчет распределения потоков по узлам сети с учетом классификации трафика и загруженности серверов и каналов связи.

В каждый момент $t \in T$ на балансировщик LB поступает трафик интенсивностью $\lambda = \{\lambda_1, \lambda_2, ..., \lambda_N\}$, относящийся к q-му классу обслуживания, который необходимо доставить на сервер $Serv_k$ для обработки, не превышая заданных максимально допустимых значений задержки τ_q и максимально допустимого процента потерь l_q в зависимости от их текущей загрузки и реальной пропускной способности в конкретный момент времени.

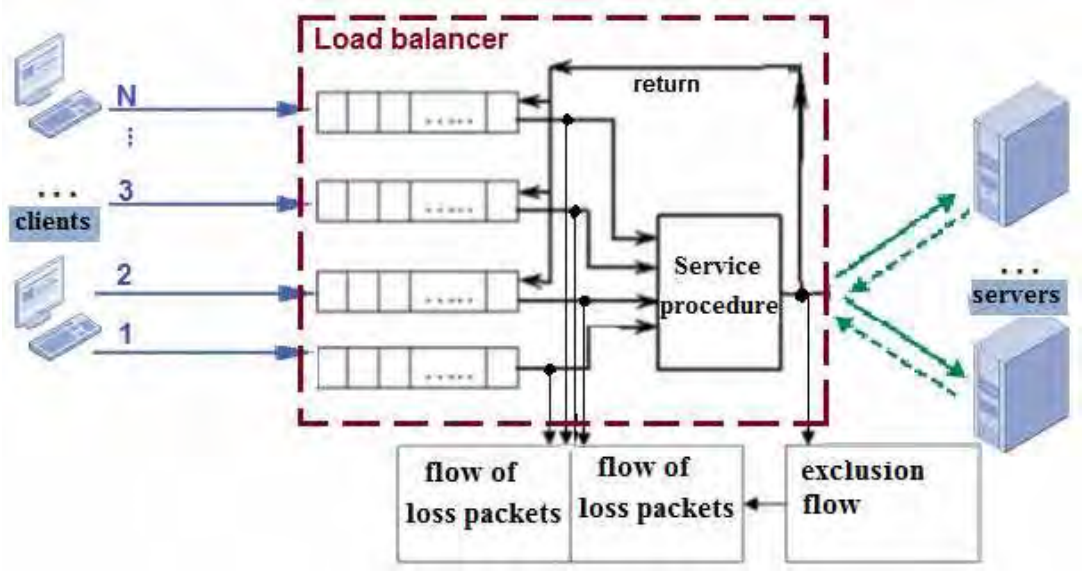

Рисунок 1 – Система балансировки нагрузки

Трафик обладает множеством характеристик $V = \{\lambda, h, \mu\}$, где $\lambda = \{\lambda_1, \lambda_2, ..., \lambda_N\}$ – потоки заявок (пакетов) различной интенсивности; $h = \{H, h(q), \Delta h\}$, где $h(q)$ – выборочное значение функции обобщенного показателя Херста, $H = h(2)$ – значение параметра Херста, $\Delta h = h(q_{\max}) - h(q_{\min})$ – диапазон значений обобщенного показателя Херста для участка трафика, μ –трудоемкость запроса. Трудоемкость запроса определяется как вектор требуемых ресурсов $\mu = \{CPU, Net_i, RAM\}$ для выполнения запроса. Каждому q-му классу обслуживания соответствует набор векторов требуемых ресурсов $\mu_r = \{CPU, Net_i, RAM\}$, $r = 1, 2, ...$.

Балансировщик нагрузки и сервера соединены между собой двусторонними сетевыми связями с максимальной пропускной способностью $Link_i = \{L_i\}$, $i = 1, 2, ..., N$, которые имеют доступную пропускную способность $Net_i(t) = \{Net_i\}$ в момент времени t. Каждый сервер $Serv_k$ характеризуется следующими параметрами: $CPU_i(t)$ – объем свободного ЦПУ i-го сервера в момент времени t, который вычисляется как $CPU_i(t) = 1 - \sum_1^j LoadM_j / j$, где $LoadM_j$ – загрузка каждого ядра $j = 1, 2, ...$ – в каждом процессоре $M = 1, 2, ...$ в сервере в момент времени t, $RAM_k(t)$ – объем свободной оперативной памяти i-го сервера в момент времени t [1,2].

То, как ядра распределены по процессорам будем считать несущественнным. Два четырехядерных соответствуют четырем двуядерным и соответствуют восьми одноядерных процессоров. Имеет значение лишь общее число ядер. Если среднее значение загрузки постоянно превышает 0.70, следует выяснить причину такого поведения системы во избежание проблем в будущем; Если средняя загрузка системы превышает 1.00, необходимо срочно найти причину и устранить ее.

На вход балансировщика нагрузки LB поступают несколько независимых мультифрактальных потоков пакетов с различной интенсивностью $\lambda = \{\lambda_1, \lambda_2, ..., \lambda_N\}$, каждый из которых отправляется в очередь Q_i ограниченной емкости. Время обслуживания заявок зависит от класса обслуживания q, то есть учитывается приоритетность заявок (наивысший приоритет – первым). Пока все приоритетные запросы на обслуживание не будут обработаны, пакеты других типов остаются в очереди до истечения их времени жизни. Вновь поступившие приоритетные запросы прерывают обработку неприоритетных и с вероятностью равной единице вытесняют их в накопитель (если есть свободные места ожидания), либо за пределы системы (если свободных мест нет). Вытесненные с обслуживания пакеты присоединяются к очереди неприоритетных требований и могут быть дообслужены после всех приоритетных. Накопители являются раздельными для каждого входного потока, свободные места ожидания полнодоступны для любого вновь поступившего запроса.

В отличие от типовых приоритетных СМО рассматриваемая система снабжена вероятностным выталкивающим механизмом. Приоритетный пакет, заставший все места ожидания занятыми в момент обработки другого приоритетного пакета, с заданной вероятностью вытесняет из накопителя один из менее приоритетных пакетов и занимает его место. Вытесненный пакет теряется либо отправляется обратно в очередь. Подсистема балансировки нагрузки LB, в соответствии с заложенным в нее алгоритмом осуществляет извлечение задач из очередей Q_i и назначение их на свободные вычислительные ядра подходящих серверов.

Для описания механизма высвобождения занятых трафиком сетевых ресурсов при окончании передачи трафика q-го класса обслуживания, (это происходит на основе данных, поступающих от

протокола маршрутизации, поддерживающего сообщения о доступной полосе пропускания и доступных ресурсах на сервере, например, CSPF, SNMP), введем переменную $\varepsilon_{C_{Li}}^{q,t_0}(t) = \{0,1\}$, указывающую, что в момент t на сервер перестал поступать трафик класса $q(\varepsilon = 1)$, который был принят на обслуживание в момент t_0 и должен был передаваться по пути $Net_i(t)$ на сервер $Serv_k$. Данная переменная содержит все необходимые данные для определения сетевых ресурсов, подлежащих высвобождению.

Балансировщик LB в t-й момент характеризуется коэффициентом потерь $X_{LB}^q(t)$ и средним временем ожидания пакета в очереди $T_{LB}^q(t)$. Переменная $X_{LB}^q(t)$ равна проценту потерь на балансировщике трафика с классом обслуживания q, передаваемого по пути $Net_i(t)$ на сервер $Serv_k$ в момент t. Предполагается, что вероятностью искажения пакета в тракте можно пренебречь и потери происходят исключительно в балансировщике из-за переполнения буфера.

Мониторинг состояния серверов и свободной пропускной способности можно осуществить тремя способами: после каждого поступившего запроса; в фиксированные промежутки времени, определяемые статическим алгоритмом; в нефиксированные промежутки времени, определяемые динамическим алгоритмом [3].

Информация, полученная первым способом, является наибольшей по объему, т.к. измерения проводятся после каждого поступившего запроса. При втором способе количество информации постоянно, но необходимо определить интервал съема информации, чтобы объем информации не был избыточным и недостаточным. При третьем способе количество информации зависит от частоты интервалов контроля, который должен приспосабливаться к структуре поступающего трафика, учитывая его самоподобную структуру.

Балансировка нагрузки с учетом мультифрактальных свойств трафика

Очереди и потери, порождаемые трафиком с мультифрактальными свойствами, зависят от мультифрактальной характеристики: функции обобщенного показателя Херста. Диапазон значений обобщенного показателя Херста соответствует степени неоднородности трафика, т.е. характеризует разброс данных. Значение обычного па-

раметра Херста, полученное из обобщенного показателя, соответствует степени долгосрочной зависимости реализации и характеризует корреляционные свойства трафика.

Использование динамически изменяющегося алгоритма является наиболее приемлемым с точки зрения уменьшения избыточности данных. При таком способе частота мониторинга будет зависеть от значений обобщенного показателя Херста $h(q)$ входящего потока [8]. Интервал мониторинга должен учащаться, если во входящем потоке диапазон значении обобщенного показателя Херста принимает значения, большие заданной величины $\Delta h = h(q_{\min}) - h(q_{\max}) > \Delta_{MAX}$, и при значении показателя Херста $H = h(q = 2) > H_{MAX} > 0.8$.

Система балансировки нагрузки состоит из компонентов, представленных на рис. 2. Балансировщик нагрузки имеет веб-интерфейс для отслеживания, настройки и администрирования распределения нагрузки [9,10].

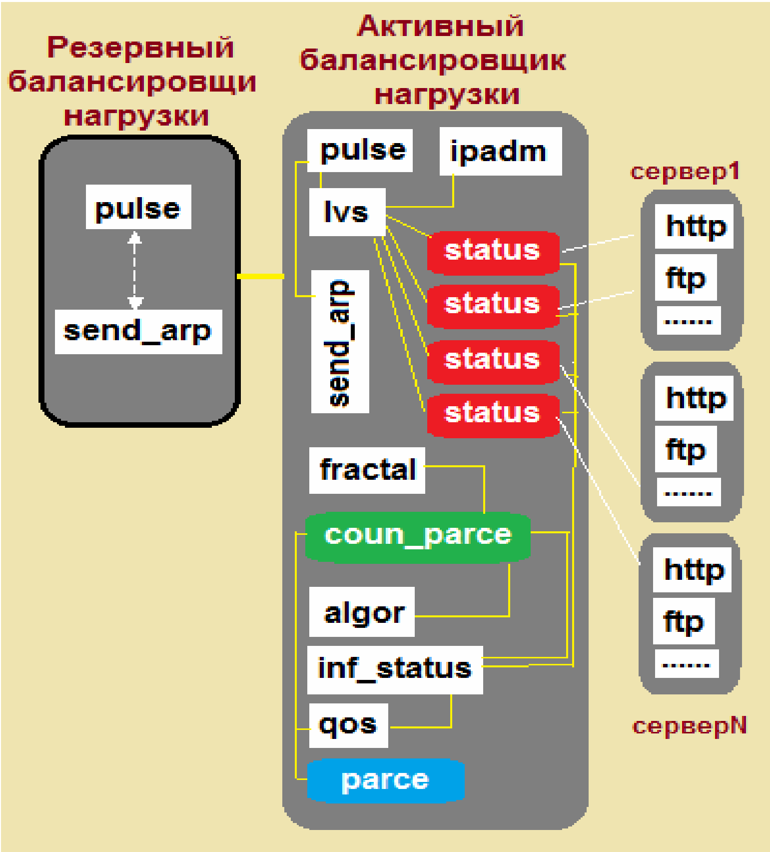

Рисунок 2 – Компоненты распределения нагрузки

Процесс `pulse` – это управляющий процесс, который на активном балансировщике запускает процесс *lvs*, а на резервном баланси-

ровщике будет отслеживать состояние активного, периодически опрашивая его. Если активный балансировщик не отвечает в течение заданного периода времени, будет инициирован процесс передачи его функций резервному балансировщику. При этом `pulse` на резервном балансировщике отправляет процессу `pulse` на активном балансировщике команду остановки всех служб *lvs*, запускает `send_arp` для присвоения виртуальных IP-адресов MAC-адресу резервного балансировщика и запускает процесс `lvs`. Процесс `send_arp` рассылает широковещательные пакеты ARP при переходе виртуального IP-адреса от одного узла другому.

Процесс `lvs` запускается на активном балансировщике по вызову `pulse`. Он вызывает службу `ipadm` для создания, добавления, изменения и удаления записей в таблице маршрутизации IP. Процесс `lvs` запускает процесс `status` для каждой настроенной службы распределения нагрузки. Если `status` сообщает о том, что реальный сервер отключен, `lvs` заставит утилиту `ipadm` удалить этот сервер из таблицы IP. Основным назначением процесса *status*, работающего на активном балансировщике, является наблюдение за нагрузкой серверов, сбор и анализ статистики о текущем состоянии системы и интенсивности трафика. Процесс *status* передает обработанную и проанализированную информацию в подсистему балансировки нагрузки процессу *inf_status*. Процесс *inf_status* в свою очередь передает информацию о текущем состоянии системы процессу *coun_parce*.

Процесс *fractal* проводит расчет мультифрактальных свойств каждого входящего потока данных и передает эту информацию процессу *coun_parce*. Процесс *algor* выбирает алгоритм балансировки нагрузки и также передает эту информацию процессу *coun_parce*. Процесс *coun_parce* проводит расчет распределения потоков по узлам сети с учетом классификации трафика и загруженности серверов и каналов связи. Результаты расчета передаются процессу *parce*, который осуществляет динамическое распределение трафика по различным каналам связи и узлам в зависимости от их текущего состояния. Также результаты расчета процесса coun_parce передаются процессу *qos* для гибкой настройки обеспечения качества обслуживания в соответствии с методами управления трафиком (управление кеш-памятью, пропускной способностью памяти и каналов, производительностью процессоров), если это необходимо. В случае применения методов

управления трафиком процессом *qos*, информация об изменениях передается процессу *inf_status*.

На основании оценивания мультифрактальных свойств входящего трафика предлагается динамический алгоритм балансировки трафика. Приведем пошаговое описание динамического алгоритма балансировки нагрузки с модифицированной обратной связью.

1. В трафике, поступающем на вход коммутатора, выделяем окно X, фиксированной длины T.

2. Находим выборочное значение функции обобщенного показателя Херста $h(q)$, значение параметра Херста $H = h(2)$ и диапазон значений обобщенного показателя Херста $\Delta h = h(q_{\min}) - h(q_{\max})$ для участка трафика в выделенном окне.

3. Проводим сбор и анализ статистической информации: интенсивности входящих потоков $\lambda_1, \lambda_2, ..., \lambda_N$, доступной пропускной способности $Net_i(t)$, состояния серверов $CPU_i(t)$, $RAM_k(t)$ – объем свободного ЦПУ и объем свободной оперативной памяти i-го сервера в момент времени t соответственно.

3. На основе мультифрактальных свойств трафика (значения из п.2), интенсивности трафика вычисляем необходимое количество ресурсов для каждого q-го класса трафика.

4. Проводим расчет распределения потоков по узлам сети с учетом классификации трафика и загруженности серверов и каналов связи. На основе полученных данных расчитывается загруженность серверов на следующем шаге.

5. Распределяем трафик по серверам, согласно алгоритму балансировки в пределах каждого класса потока.

6. Проводим распределение недооценки прогнозируемого количества ресурсов $Net_i(t)$, $CPU_i(t)$, $RAM_k(t)$. Переоценка не учитывается алгоритмом, т.к. не вносит существенных изменений.

7. Проводим сбор данных о загруженности серверов $Net_i(t)$, $CPU_i(t)$, $RAM_k(t)$ и передачу их в систему балансировки нагрузки для расчета нового распределения потоков.

8. Передвигаем окно X длины T вперед на заданную величину сдвига ΔT; осуществляем анализ трафика и расчет следующего значения загруженности серверов.

Предложенный метод должен обеспечивать статистически равномерное распределение нагрузки на серверах, высокие показатели производительности, пропускной способности, отказоустойчивости (автоматически обнаруживая сбои узлов и перераспределяя поток данных среди оставшихся) и низкое время отклика, количество служебной информации, количество потерянных данных.

Выводы

В работе предложена математическая модель системы балансировки нагрузки, в которой балансировщик нагрузки описывается с помощью системы массового обслуживания. Состояния серверов описываются объемом свободного ЦПУ и объемом свободной оперативной памяти. Все значения параметров модели имеют зависимость от времени. Такая модель позволяет описывать поведение распределенной сети во времени для различных классов обслуживания входящего трафика. Также в работе предложен динамический метод распределения нагрузки, который учитывает мультифрактальные свойства трафика. Предлагаемый метод балансировки нагрузки благодаря анализу и учету мультифрактальных свойств входного потока обеспечивает статистически равномерное распределение нагрузки на серверах, высокие показатели производительности и пропускной способности, а также снижение времени отклика и количества потерянных данных.

ЛИТЕРАТУРА

1. Тарасов В.Н. Математические модели облачного вычислительного центра обработки данных с использованием Openflow / Тарасов В.Н., Полежаев П.Н., Шухман А.Е., Ушаков Ю.А., Коннов А.Л. // ВЕСТНИК ОГУ №9 (145)/сентябрь`2012. С.150-155.
2. Andre Understanding Linux CPU load - when should you be worried? 2009. URL:

http://blog.scoutapp.com/articles/2009/07/31/understanding-load-averages

3. Е.И.Игнатенко. Адаптивный алгоритм мониторинга загруженности сети кластера в системе балансировки нагрузки. / Е.И.Игнатенко, В.И.Бессараб, И.В.Дегтяренко // Наукові праці ДонНТУ. Вип.21(183). 2011. С.95-102.
4. Дорт-Гольц А. А. Разработка и исследование метода балансировки трафика в пакетних сетях святи / Диссертация на к.т.н. // Федеральное агентство связи. Федеральное государственное образова-

тельное бюджетное учреждение высшего профессионального образования «Санкт-Петербургский государственный университет телекоммуникаций им. проф. М.А. Бонч-Бруевича». 2014. С.168

5. Sheluchin O. I. Self-Similar Processes in Telecommunications / O. I. Sheluchin, S. M. Smolskiy, A. V. Osin // New York : John Wiley & Sons. – 2007. – 320 p.
6. Шелухин О. И. Мультифракталы. Инфокоммуникационные приложения / О. И. Шелухин. – М. : Горячая линия – Телеком, 2011. – 576 с.
7. Кириченко, Л.О. Анализ методов повышения QoS в сетях MPLS с учетом самоподобия трафика / Л.О. Кириченко, Э. Кайали, Т.А. Радивилова // Системні технології. – 2011. – Вип. 3. – С. 52–59.
8. Kirichenko, L. Modeling telecommunications traffic using the stochastic multifractal cascade process // L. Kirichenko, T. Radivilova, E. Kayali // Problems of Computer Intellectualization / ed. K. Markov, V. Velychko, O. Voloshin. – Kiev–Sofia: ITHEA. – 2012. – P. 55–63.
9. Red Hat Enterprise Linux. V.6. Администрирование виртуального сервера.

https://access.redhat.com/documentation/ru-RU/Red_Hat_Enterprise_Linux/6/html/Virtual_Server_Administration/ch-lvs-overview-VSA.html

10. Wenhong Tian. Optimized Cloud Resource Management and Scheduling: Theories and Practices. /Wenhong Tian, Yong Zhao // Morgan kaufmann. 2014. P.284.