Информатика и вычислительная техника

УДК 004.71 **DOI: 10.14529/ctcr200301**

МЕТОДИКА ПОЛУЧЕНИЯ ОБОБЩЕННЫХ СТАТИСТИЧЕСКИХ ХАРАКТЕРИСТИК СЕТЕВОГО ОБОРУДОВАНИЯ ДЛЯ СОЗДАНИЯ ИМИТАЦИОННЫХ МОДЕЛЕЙ

И.П. Болодурина, Ю.А. Ушаков, М.В. Ушакова

Оренбургский государственный университет, г. Оренбург, Россия

Работа посвящена проблеме разработки моделей сетевого оборудования, которые точно воспроизводят работу реального оборудования по временным характеристикам.

Цель исследования. Данное исследование направлено на разработку методики, которая позволяет быстро получать модели, статистически совпадающие по поведению выбранных параметров с реальным измеренным оборудованием на существующих открытых системах моделирования.

Методы. Для получения параметров производительности и временных характеристик реального оборудования создана методика, включающая планирование эксперимента для получения статистических данных в широком диапазоне входных интенсивностей трафика, конфигурацию и схему соединения оборудования, включающую внешнее управление коммутатором для тестирования влияния различных настроек на производительность и время обработки пакета, скрипты для автоматизации создания тестовых наборов трафика с требуемыми параметрами распределения времени между пакетами, схему подключения через контейнерные службы и схему съема дампов трафика. В методику не вошло тестирование работы выходных очередей, целью было исследовать и перенести в модель процессы, происходящие до помещения фреймов в исходящие очереди интерфейсов. В качестве среды моделирования использовался пакет с открытым исходным кодом Omnet++.

Результаты. В работе изучена структура существующего модельного коммутатора Omnet++, выявлены недостатки, влияющие на точность и достоверность результатов, предложен метод реализации линий модельной задержки на основе интерполяции таблиц временных параметров, полученных по результатам реальных данных для требуемого сочетания параметров трафика и входящих интерфейсов. Для апробации разработанных подходов разработан тестовый стенд на реальном оборудовании, сформирован план экспериментальных исследований, реализован скрипт автоматического тестирования по заданному плану, проведено тестирование в режиме обычной коммутации фреймов и в режиме коммутации на основе таблиц Openflow. Также получены временные характеристики промежуточного оборудования для учета их влияния на результаты эксперимента. По полученным результатам проведен эксперимент на модели в среде Omnet++. Поскольку полученные выборки при статистическом анализе показали виды распределений, отличные от нормальных и экспоненциальных, для сравнения средних использовался статистический U-критерий Манна–Уитни.

Заключение. Полученные результаты при уровне значимости 0,05 подтвердили работоспособность данного подхода при создании моделей, более точно описывающих временные характеристики процессов обработки трафика в сетевом оборудовании.

Ключевые слова: моделирование, имитационная модель, коммутация, задержки.

Введение

Современные высокоскоростные сети передачи данных должны обладать предсказуемым поведением при любых сочетаниях параметров проходящего трафика и настроек оборудования. Основной единицей построения высокоскоростных магистралей до конечного оборудования являются коммутаторы, которые постепенно усложнялись, увеличивая функциональность. На сегодняшний день провайдеры начинают использовать оборудование на основе программно-

конфигурируемых сетей, которые предоставляют более гибкие подходы к управлению трафиком. Повсеместное внедрение виртуализации увеличивает требования к оборудованию, так как коммутаторы должны совмещать разнородный трафик виртуальных сервисов, особенно для подхода с использованием виртуализации сетевых функций (NFV, Network Function Virtualization) [1]. Основной проблемой использования совместно всех этих инструментов является прогнозирование задержек, потери пакетов, размеров очередей для обеспечения требуемого качества обслуживания (QoS). Исследование таких сетей часто выполняют на моделях, к которым предъявляются повышенные требования к точности воспроизведения условий реального оборудования.

Имитационное моделирование предоставляет широкие возможности для исследований и позволяет получить результаты, сопоставимые с результатами реального оборудования, но только в случае точного описания всех механизмов работы и их временных и нагрузочных характеристик. Этот подход дает возможность проведения экспериментов без построения реальной сети, что существенно снижает временные и финансовые затраты [2].

На сегодняшний день существует довольно большое количество систем моделирования сети. Они используют в своей работе данные о количестве узлов, конфигурации связей, скоростях передачи данных, используемых протоколах и видах оборудования, а также о выполняемых в сети приложениях [3].

Большинство систем имитационного моделирования позволяют не строить модель с нуля, а использовать готовые модели основных элементов сети, таких как, например, некоторые типы маршрутизаторов, каналы связи, протоколы и т. д. В результате тестовых испытаний реального оборудования и анализа принципов их работы создается библиотека типовых элементов сети, которые можно настраивать с помощью заранее предусмотренных в моделях параметров. Использование библиотек готовых решений существенно упрощает разработку модели сети. Тем не менее высокие темпы развития сетевых технологий приводят к тому, что существующие модели не удовлетворяют потребностям разработчиков.

Моделирование сетевой инфраструктуры на основе ПКС имеет ряд особенностей [4]. Во-первых, время принятия решений по коммутации и маршрутизации в таком оборудовании практически не отличается и зависит только от реализации поиска в таблицах OpenFlow в реактивном режиме (с заранее загруженными данными о топологии). Во-вторых, отличается профиль нагрузки на процессор и память при прохождении трафика. В-третьих, такое оборудование может работать в гибридных режимах, когда часть портов работает в режиме ПКС, а остальные в режиме традиционной коммутации или маршрутизации.

Работа очередей в таких системах не отличается от традиционных коммутаторов и маршрутизаторов, кроме того, что настройка параметров этих очередей может происходить динамически. Создание моделей, в которых используется также проактивный режим работы ПКС, требует введения в процесс коммутации и маршрутизации процесса работы стороннего приложения на внешнем сервере, доступного также через сеть передачи данных с заранее неизвестными параметрами по реакции на нагрузку [5].

Получение характеристик оборудования и программных модулей возможно при помощи прогона стресс-тестирования, причем при использовании ПКС можно генерировать трафик и принимать его с помощью только механизмов протокола OpenFlow в проактивном режиме. Для минимизации влияния времени запроса и ответа к серверу все остальные процессы, кроме генерации приема трафика, должны проходить в реактивном режиме с минимально необходимыми размерами таблиц. Стресс-тестирование на реально работающих сетях связано с трудностями сосуществования синтетического и реального трафиков. С точки зрения получения всех возможных вариантов и сочетаний характеристик оборудования такой режим позволит создать модели только для нагрузки равной или выше существующей. Поскольку при использовании ПКС можно снимать очень подробную статистику с каждой единицы оборудования, есть возможность проводить такие стресс-тесты по дереву связанных коммутаторов и маршрутизаторов, одновременно снимая характеристики со всех устройств на одном и том же паттерне трафика.

В данной работе рассмотрена методика расширения возможностей по моделированию внутренних процессов коммутатора в среде OMNeT++ с использованием фреймворка INET, который обладает широким набором уже готовых компонентов для описания различных элементов сети [6].

1. Снятие характеристик реального сетевого оборудования для целей модельного исследования

При моделировании сетевого оборудования применяют несколько степеней детализации и, соответственно, уровней декомпозиции. Большинство модельных исследований коммутаторов ориентируются на изучение пропускной способности, задержек, потерь пакетов, поведения очередей, что требует детализации до уровня интерфейсов, но не внутренних процессов принятия решений, поиска адресов, классификации, фильтрации и прочих важных процессов. Зависимость задержки обработки пакета от типа настроек и их количества бывает очень существенной и нелинейной. Вероятность потери пакета во время обработки внутренними программными модулями, его фильтрации, изменения существенных полей заголовков, влияющих на попадание в разные очереди, не учитываются в стандартных моделях большей части систем моделирования [7].

Рассмотрим схему прохождения пакета сквозь коммутатор. Для примера возьмем программное обеспечение для коммутатора: Cisco IOS – как наиболее полно отражающее все функции коммутатора в кампусных сетях, Mikrotik – как представителя Linux коммутации и маршрутизации, Huawei – как представителя альтернативного подхода к распределению управления потоком данных. Общая схема прохождения пакета при условии коммутации второго уровня выглядит, как показано на рис. 1. Порядок действий у разных производителей может отличаться, некоторых шагов может не быть или, наоборот, имеются дополнительные, но сам принцип соблюдается. Все необходимые действия на входе в интерфейс выполняются последовательно, внося задержку [8]. Каждое действие внутри состоит из расчетов (например, в профилировщиках и ограничителях потока), множества сравнений со списками переменной длинны (фильтры, модификаторы, классификаторы), доступом к общей ассоциативной программной памяти (быстрая коммутация) или аппаратным способам работы с таблицами TCAM (классическая коммутация) [9].

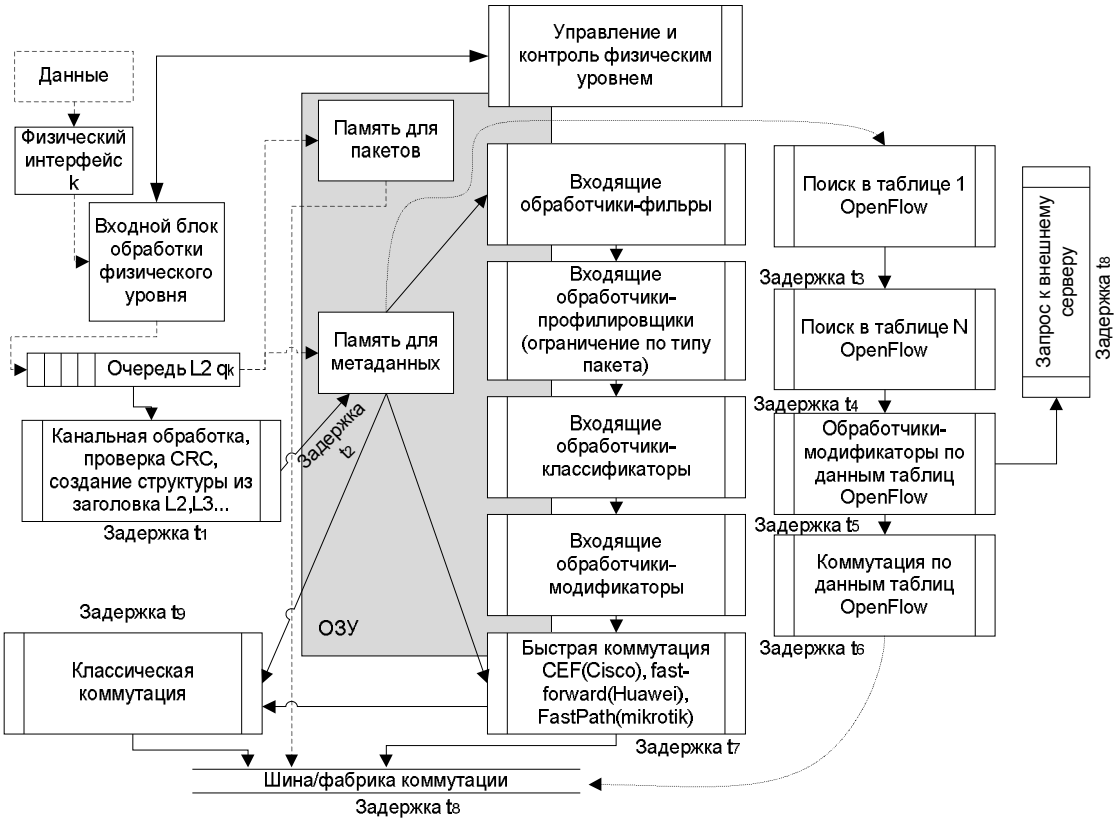


Рис. 1. Схема обработки данных на входе в коммутатор
Fig. 1. Switch input data processing scheme

Большую задержку могут вносить длинные списки в операциях фильтрации, классификации и модификации, а также занятость процессора, памяти и аппаратных модулей при работе с внешними данными (кэшами и таблицами). Влияние нагрузки процессора, количества запросов к па-

мяти и модулям на общую задержку может быть весьма существенным, поэтому в современных коммутаторах разделяют уровень управления и уровень передачи данных и часто выделяют им свои процессоры/ядра и память. Но общее влияние всех этих процессов может быть существенно [10]. Так, Mikrotik заявляет о двукратном снижении производительности при включении фильтрации. В коммутаторах Cisco часть фильтров ACL, Route-map обрабатывается в аппаратных TCAM-таблицах (например, в Catalyst 4500 только по маскам подсетей /24 и /32), остальные фильтры обрабатываются процессором и снижают общую производительность.

Общую задержку прохождения пакета через коммутатор можно принять за сумму задержек обработки, ожидания в очереди, отправки.

$$t_{switch} = t_{process} + t_{queue} + t_{transmit}, \quad (1)$$

где $t_{process}$ – суммарное время обработки пакета от получения последнего бита на входе до помещения пакета в очередь; t_{queue} – время ожидания в очереди; $t_{transmit}$ – время на передачу данных в линию связи, для полнодуплексных линий $t_{transmit} = L / C$, где L – длина передаваемых данных, вместе с преамбулой и окончанием, в бит; C – скорость линии, в бит/с.

Рассмотрим состав задержки обработки

$$t_{process} = \sum_i (t_i n_i + t'_i n'_i + t_i^{wait} + t_i'^{wait}), \quad (2)$$

где t_i – время выполнения одной операции i-го блока обработки в программном режиме; n_i – количество операций i-го блока обработки, выполняемого программным способом, для каждого пакета может быть разным; t'_i – время выполнения одной операции i-го блока обработки в аппаратном режиме; n'_i – количество операций i-го блока обработки, выполняемого аппаратными модулями, для каждого пакета может быть разным; t_i^{wait} – суммарное время ожидания доступа к разделяемым ресурсам процессора при выполнении i-го блока в программном режиме; $t_i'^{wait}$ – суммарное время ожидания доступа к разделяемым ресурсам процессора при выполнении i-го блока в аппаратном режиме.

Поскольку современный коммутатор – устройство с несколькими вычислительными ядрами, десятками интерфейсов, современной операционной системой и несколькими шинами данных, задержка выполнения каждой операции может варьироваться в широких пределах, но не будет менее $t_i n_i + t'_i n'_i$. Поскольку максимальное значение суммарной задержки зависит от ожидания доступа к ресурсам при повышении нагрузки на коммутатор в целом, необходимо оценить вероятности возникновения доступа к разделяемым ресурсам [11].

Наложение множества независимых случайных потоков приводит к простейшему потоку с экспоненциальным законом распределения [12]. Функция распределения $F(t)$ времени t между поступлениями двух задач обработки одного класса (задач работы с одними разделяемыми ресурсами) определяется как

$$F(t) = 1 - e^{-t \cdot \lambda_i}, \quad (3)$$

где λ_i – интенсивность поступления задач одного класса.

Получение данных о задержке обработки пакета в реальном коммутаторе возможно из длины входной очереди q_i (для коммутаторов с аппаратными обработчиками портов, например в Cisco – командой *show buffers input-interface*, в Linux длины очереди драйвера через утилиту *ethtool* с ключом *–S*, для OpenvSwitch – командой *ovs-ofctl queue-stats*).

Из теории систем массового обслуживания известно, что время ожидания в очереди связано с длинной очереди соотношением

$$\overline{T}_k = \frac{\overline{L}_k}{\lambda_k}, \quad (4)$$

где $\overline{L}_k$ – средняя длина очереди в данный временной интервал на интерфейсе k; λ_k – интенсивность поступления новых пакетов на интерфейсе k, $\lambda_k = 1/\tau_k$ для экспоненциального закона распределения интервалов времени между пакетами τ_r.

Для аппаратных коммутаторов $\overline{T_k} = t_{process}$ для порта k, поскольку пакет находится во входящей очереди до момента коммутации. Для задания параметров модели, адекватной реальному коммутатору в плане задержек по обработке пакетов до коммутации, необходимо создать модуль очереди с обрабатывающим устройством, вносящим задержки в соответствии с функцией задержки реального коммутатора. В общем случае для этого требуется распределение (3).

Для снятия показаний задержки во входящей очереди реального коммутатора достаточно создать план экспериментальных исследований, постепенно увеличивающий количество пакетов на одном интерфейсе и общее количество задействованных интерфейсов на коммутаторе. Для этого удобнее всего использовать второй коммутатор в режиме моста (для Cisco это включение всех портов в *bridge-group* с установленным *protocol ieee*), соединённый всеми портами 1:1 с исследуемым коммутатором, а также Linux компьютер с несколькими сетевыми картами с аппаратной обработкой трафика (для снижения использования процессора при обработке трафика).

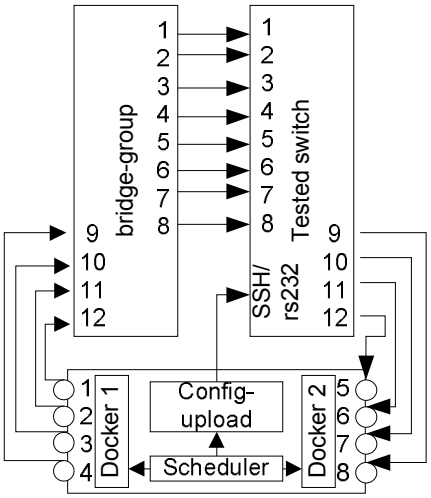


Рис. 2. Схема стендового съема данных для модели
Fig. 2. Scheme of bench data acquisition for the model

На рис. 2 показан пример схемы снятия данных о производительности и задержке передачи коммутатора. Трафик генерируется в контейнере 1, принимается в контейнере 2, дамп трафика записывается с временными метками точностью до микросекунд. Планировщик запускает различные схемы распространения трафика через включение новых портов (через прямое управление коммутатором), варьирует интенсивность, запрашивает данные о состоянии очередей, потерянных пакетов, загрузки памяти, процессора, прерываний, срабатывания кэша. Крестиками показаны временно отключенные каналы. Эта схема позволяет провести полное исследование всех аспектов работы коммутатора.

План эксперимента должен включать различные сочетания как паттернов трафика, так и количества задействованных портов на вход и выход. Несколько выходящих портов и, соответственно, адресов назначения пакетов используются для минимизации воздействия выходящей очереди.

Простейший план эксперимента включает в себя генерацию трафика по экспоненциальному закону распределения времени между пакетами, с постепенным увеличением интенсивности и количества задействованных портов.

$$\Lambda = \begin{bmatrix} \lambda_{11} & \lambda_{12} & \lambda_{1k} \\ \lambda_{21} & \dots & \dots \\ \lambda_{m1} & \dots & \lambda_{mk} \end{bmatrix}, \quad (5)$$

где k – количество интерфейсов для тестирования; m – количество планов эксперимента.

$$\lambda_{i,j} = h(i,j) \cdot \lambda_{i,j-1}, i \in (1,k], j \in (2,m], \quad (6)$$

где функция $h(i,j) = (i \cdot m + j) \cdot 2^{2R}$ предоставляет прогрессивный коэффициент для показательного увеличения трафика, коэффициент R используется для выбора такого шага, чтобы $R : \lambda_{m,k} \le \sum_{i=1}^{k} \mu_i$, где μ_i – производительность обслуживания i-го интерфейса.

Каждый элемент матрицы Λ используется для генерации трафика интенсивностью $\lambda_{i,j}$ при количестве задействованных входных интерфейсов, равном j. Результаты записываются в выходные матрицы D (средняя полная задержка передачи), Q_i (средний размер входящей очереди), Q_o (средний размер исходящей очереди – только если используется один исходящий в генератор трафика порт), L (количество потерянных пакетов).

Если уже существующая матрица результатов эксперимента слишком подробная для использования в модели, ее аппроксимируют матрицей меньшего ранга любым выбранным методом, например методом наименьших квадратов или сплайнами.

2. Использование полученных данных в модели коммутатора

Рассмотрим базовую модель коммутатора, представленного в фреймворке INET Omnet++. Коммутатор представлен набором модулей, взаимодействующих через систему передачи сообщений Omnet [13]. Наиболее подробно реализованы модули интерфейсов Ethernet, в частности имеются реализации входящей очереди типа Tail-Drop, исходящих очередей типов FIFO, RED, WRED, PQ, CBFQ, также возможна реализация задержки передачи на линии. Однако в этой базовой модели отсутствует реализация или имитация процессов внутренней обработки пакетов, вся коммутация представлена модулем MacAddressTable с поиском адресов, модулем MacRelayUnit, просто передающим пакеты на исходящий интерфейс без задержек. Более подробная реализация коммутатора от фреймворка ANSA Omnet++ имеет множество протоколов L3-L4 в составе коммутатора, реализации OpenFlow-коммутатора наиболее полно описывают процесс работы, но только в режиме Openflow, а не в обычном режиме коммутации [14]. При этом в Openflow режиме существует возможность генерации реальных запросов к настоящему внешнему контроллеру и интерпретации ответов. Моделирование всех внутренних процессов обработки пакета в общем случае не требуется, поскольку изучаются поведение задержки, очередей, потери пакетов и приоритезации. Для целей получения адекватной модели на уровне первых моментов распределения времени между пакетами достаточно симулировать поведение функции обработки по задержке с теми же первыми моментами распределений.

Рассмотрим создание блока функции задержки обработки на примере выходной матрицы средней длины входящих очередей Q_i, имеющей ту же размерность, что и Λ, т. е. каждому значению $\lambda_{i,j}$ однозначно соответствует значение $q_{i,j}$. Для того чтобы определить среднее значение очереди $F(\lambda_0, j)$ для произвольного λ_0 при заданном количестве интерфейсов j, необходимо найти такое значение i, при котором $\lambda_{i,j} \le \lambda_0 < \lambda_{i+1,j}$.

Тогда в случае использования линейной интерполяции отрезков между узлами, образованными столбцом j матрицы Λ, значение длины очереди для матрицы результатов (например, матрицы Qi) вычисляется по формуле

$$F(\lambda_0, j) = q_{i,j} + u \cdot (q_{i+1,j} - q_{i,j}), \tag{7}$$

где коэффициент $u = \dfrac{\lambda_0 - \lambda_{i,j}}{\lambda_{i+1,j} - \lambda_{i,j}}$.

Как правило, функции зависимости задержки от интенсивности поступления пакетов имеют вид степенных функций, погрешность кусочно-линейной интерполяции по сравнению с использованием более высоких порядков интерполяции зависит от числа строк матрицы Λ и шага изменения параметров. Она может быть сравнима с погрешностью экспериментальных измерений при коротком шаге и вносить существенные изменения в функционирование сети на длинных шагах. Для шага между измерениями более 50 пакетов лучше использовать интерполяцию более высокими степенями.

3. Постановка эксперимента

Схема для снятия показаний выполнена по схеме рис. 2. В качестве коммутатора для распределения использовался Cisco 3750x, в качестве исследуемого коммутатора – HP 3500yl-24. Станция тестирования –Ubuntu 18.04 на базе Xeon E5 с 4 встроенными сетевыми картами 10Гб/с с поддержкой аппаратной обработки пакетов.

Спецификация изучаемого коммутатора содержит сведения о наличии 3000 записей в TCAM-таблице для различных правил, включая классификацию, ACL. Кроме этого, утверждается одновременная коммутация на скорости линии через все порты. Но при включении Openflow-режима все может измениться даже при реактивном способе задания правил. Первую часть эксперимента проведем с загрузкой портов 1–20 трафиком 64 байта (как наиболее ресурсозатратного). Максимальный теоретический максимум передачи с учетом преамбулы и интервала между фреймами составляет 1,488 млн фреймов/с с полезным использованием полосы пропускания 761 Мбит/с. 20 портов теоретически могут передать 28,960 млн фреймов/с. В табл. 1 показан прогрессивный план эксперимента (матрица Λ).

Таблица 1

План эксперимента – матрица Λ , в тыс. пакетов/с

Table 1

Experiment plan – matrix Λ , thousands of packets per second

№	Кол-во задействованных портов					
	1	2	5	10	15	20
1	2	5	38	313	1352	4145
2	38	62	219	1043	3380	8679
3	219	313	793	2731	7291	16464
4	793	1043	2185	6084	14139	28959

Трафик UDP будет генерироваться утилитой *mgen* по сгенерированному скрипту посылки пакетов с требуемым законом распределения. Съем посланного и принятого трафика осуществляется утилитой *tcpdump* с записью на *tmpfs*-диск и опцией записи временных меток с микросекундами. Каждый пакет имеет в полезной нагрузке порядковый номер в ASCII, что позволит отследить его задержку простыми утилитами сравнения *linux*.

4. Эксперимент

Тестирование проводилось на стенде, состоящем из 2 коммутаторов и одного сервера (см. рис. 2). В качестве коммутатора для распределения использовался Cisco 3750x, в качестве исследуемого коммутатора – HP 3500yl-24. Станция тестирования – Ubuntu 18.04 на базе Xeon E5/32Гб JPE/4NVe SSD RAIN 10, материнская плата Supermicro с 4 сетевыми картами 10Гб/с на базе модуля AOC-STG-b4S с поддержкой аппаратной обработки пакетов.

Для автоматизации анализа параметров коммутатора создан скрипт, который по плану эксперимента генерирует трафик на требуемое количество портов с требуемыми параметрами распределения, собирает дампы трафика (использовалось ограничение – записывались только заголовки и время фрейма). Каждый пакет посылается на следующий по порядку порт UDP, для того чтобы быстро найти этот пакет на приемной стороне в случае разного порядка получения. Запись происходит на файловую систему TMPFS с заранее выделенным участком памяти для снижения времени выделения новой памяти, включенной опцией «noatime» с высокой скоростью записи и почти нулевым временем поиска. После каждого прогона из записи дампов извлекалось время и номер UDP порта по каждому интерфейсу, затем рассчитывались разности времени в микросекундах. Потом файлы записи трафика очищались для освобождения места для нового прогона.

При тестировании выяснилось, что имеющаяся материнская плата имеет ограничение в 40 Гбит/с на всю шину с сетевыми картами, поэтому максимальная полоса пропускания на выходе из двух сетевых карт составила не более 23,5 Гбит/с с учетом накладных расходов.

Для измерения задержки на сетевом интерфейсе проведен эксперимент с соединением напрямую входов и выходов 10 Гбит/с (табл. 2, 3).

Таблица 2

Прямое соединение сетевых карт

Table 2

Direct network card connection

Параметр	Задержка пакетов		
Размер пакета	64	512	1518
Задержка, мкс	2,04	2,19	2,21

Таблица 3

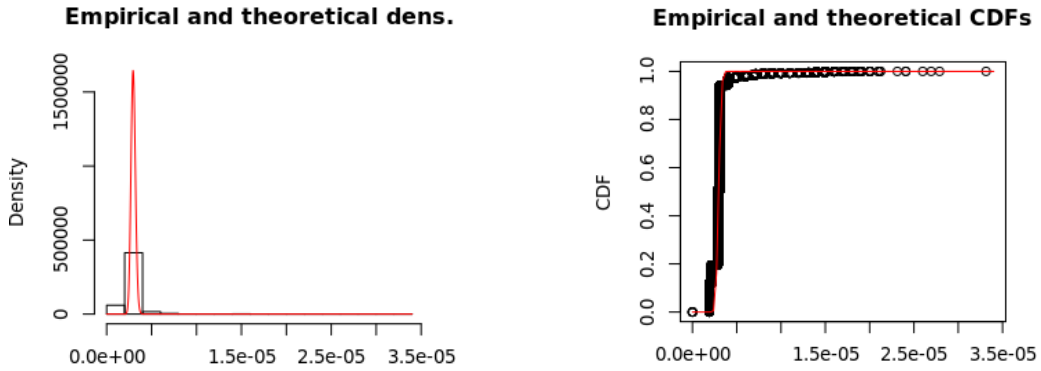
Задержки в служебном коммутаторе Cisco 3750x

Table 3

Latency of intermediate switch Cisco 3750x

Параметр	Задержка пакетов		
Размер пакета	64	512	1518
Задержка на передаче 10→10 Гбит/с, мкс	4,27	4,89	5,96
Задержка на передаче 10→1 Гбит/с, мкс	7,21	12,1	18,41

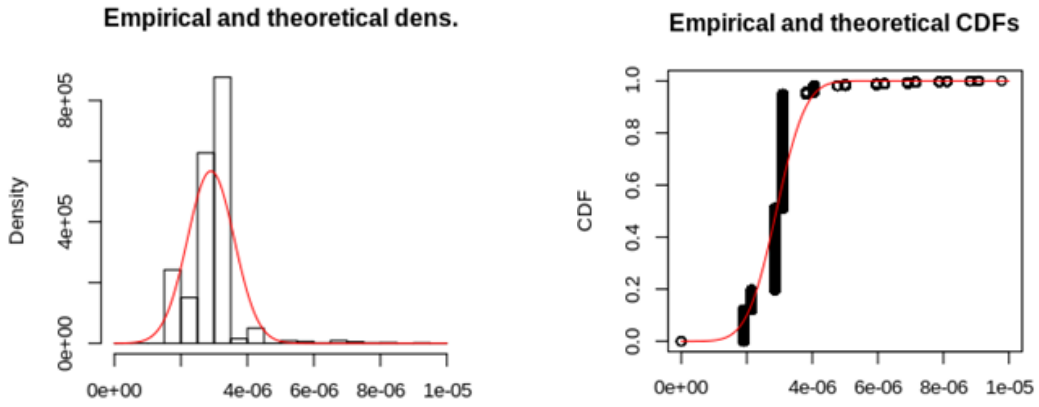
Анализ трафика проводился средствами языка R, имеющего широкий выбор библиотек статистического анализа. На рис. 3 показано распределение задержки прохождения пакета в коммутаторе при максимальной нагрузке 28960 пакетов/с размером 64 байт.



а) б)

Рис. 3. Гистограмма(а) и функция распределения (б) бета-распределения (линия) и выборки времени задержки на всем диапазоне данных
Fig. 3. Histogram (a) and distribution function (b) beta distribution (line) and sampling for full-range latency

Длинный хвост распределения образуется из-за попадания отдельных пакетов в исходящую очередь и, соответственно, существенно большего времени ожидания посылки. При увеличении детализации в диапазоне $[0;10^{-5}]$ видны явные неоднородности (ступени), хотя объем выборки более 1 млн (рис. 4). Эти задержки соответствуют работе систем кэширования и аппаратных очередей.



а) б)

Рис. 4. Гистограмма (а) и функция распределения (б) нормального распределения (линяя) и выборки времени задержки в диапазоне до 1e-5
Fig. 4. Histogram (a) and distribution function (b) of the normal distribution (line) and sampling latency for range up to 1e-5

Затем прогон эксперимента по плану (см. табл. 1) показал следующие значения (табл. 4).

Таблица 4

Задержки в исследуемом коммутаторе HP 3500yl в обычном режиме, мкс

Table 4

Latency in the test switch HP 3500yl in legacy mode, μs

№	Кол-во задействованных портов					
	1	2	5	10	15	20
1	6,91	6,91	6,94	7,19	7,40	7,55
2	6,94	7,05	7,08	7,33	7,55	7,70
3	6,98	7,19	7,22	7,48	7,70	7,86
4	7,01	7,33	7,36	7,63	7,85	8,51

Средняя длина входящей очереди во всех экспериментах, кроме последнего (20 портов, 28959 тыс. пакетов/с, очередь – 3 пакета), составила 1 пакет. В табл. 5 показаны результаты эксперимента по 4-й строке матрицы Λ для Openflow в реактивном режиме с разрешением 0,5 мкс. По графику поведения задержки (рис. 5) видны определенные (практически дискретные), наиболее часто встречающиеся уровни задержки – около 4, 5, 10, 11 мкс. Значения выше 10 мкс – ожидания в выходной очереди.

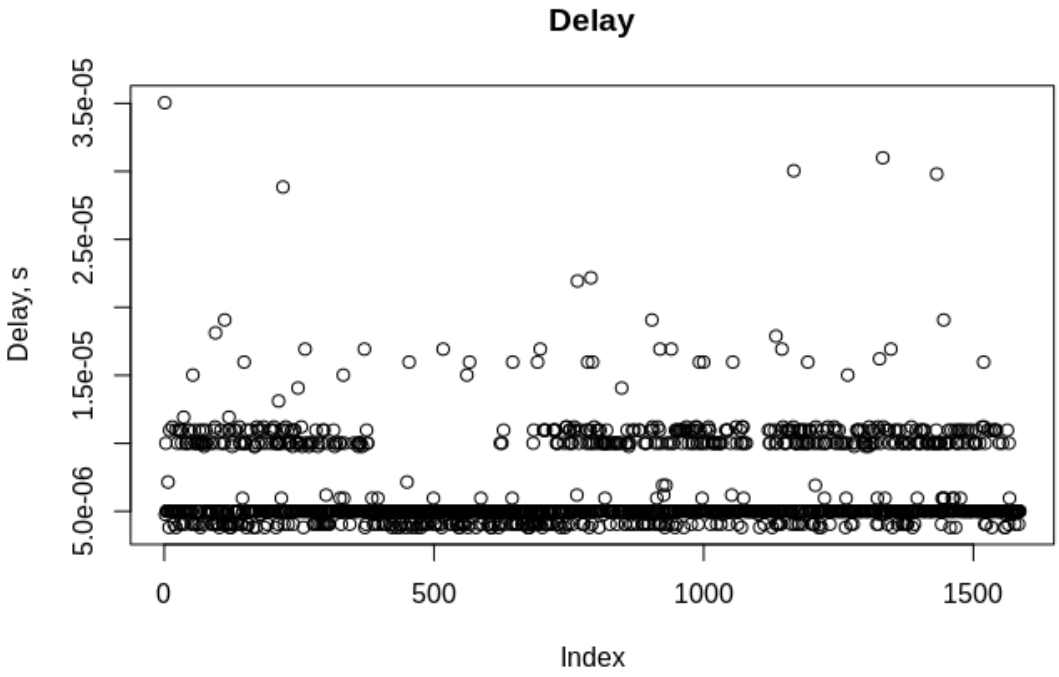


Рис. 5. График задержки внутри коммутатора
Fig. 5. Latency graph for the switch

Таблица 5
Задержки в исследуемом коммутаторе HP 3500yl в Openflow-режиме, мкс
Table 5
Latency in the test switch HP 3500yl in the Openflow mode, μs

№	Кол-во задействованных портов					
	1	2	5	10	15	20
4	271,3	284,9	293,4	299,3	305,3	308,3

Как видим, оптимизация работы с Openflow на этом коммутаторе отсутствует, и латентность поиска в таблице повышается в несколько раз.

Для сравнения данных модель коммутатора OMNET тестировалась на 4-й строке плана эксперимента. Было задействовано столько генераторов экспоненциального трафика, сколько портов требовалось в плане эксперимента. Статистика средней задержки рассчитывалась по дампам событий пакетов, сохраненных в скалярный файл. Результаты показаны в табл. 6.

Таблица 6
Задержки в модельном коммутаторе в обычном режиме, мкс
Table 6
Latency in the model switch in legacy mode, μs

№	Кол-во задействованных портов					
	1	2	5	10	15	20
4	7,00	7,15	7,25	7,38	7,5	8,25

Средний размер очереди при k = 20 составил 3 пакета, данные модели согласуются с данными эксперимента в пределах 5 %.

Для оценки различий между выборкой с экспериментальными данными о задержке и выборкой, полученной в эксперименте с моделью по каждой позиции плана эксперимента, использовался статистический U-критерий Манна–Уитни как менее подверженный влиянию вида статистического распределения. Для уровня значимости 0,05 значение U-критерия по каждой позиции превысило критическое, а это означает, что модель адекватно описывает реальный коммутатор по критерию задержки, возникающей при обработке пакетов.

Заключение

Рассмотренный подход позволяет быстро получать модели, статистически совпадающие по поведению выбранных параметров с реальным измеренным оборудованием на существующих открытых системах моделирования. Данный подход будет развиваться в дальнейшем в сторону более детального описания внутренних процессов и их статистического описания на основе экспериментальных данных.

Рассмотрена структура моделей Omnet++, предложены методы модификации моделей для повышения уровня адекватности внутренней обработки пакетов.

Полученные результаты при уровне значимости 0,05 подтвердили работоспособность данного подхода при создании моделей, более точно описывающих временные характеристики процессов обработки трафика в сетевом оборудовании.

Разработанная методика исследования оборудования позволяет быстро исследовать статистические параметры производительности и будет развиваться дальше в сторону получения данных без отключения от сети.

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 20-07-01065, а также гранта Президента Российской Федерации для государственной поддержки ведущих научных школ Российской Федерации (НШ-2502.2020.9).

Литература

1. Network function virtualization enablement within SDN data plane / H. Mekky, F. Hao, S. Mukherjee et al. // IEEE INFOCOM 2017-IEEE Conference on Computer Communications. – IEEE, 2017. – P. 1–9.

2. Кутузов, О.И. К анализу парадигм имитационного моделирования / О.И. Кутузов, Т.М. Татарникова // Научно-технический вестник информационных технологий, механики и оптики. – 2017. – № 3

3. Акатов, Д.В. Характеристики основных средств для анализа и оптимизации корпоративных сетей / Д.В. Акатов, А.Г. Юрочкин // Моделирование, оптимизация и информационные технологии. – 2015. – № 2. – С. 8.

4. Model-Based Performance Predictions for SDN-Based Networks: A Case Study / S. Herrnleben, P. Rygielski, J. Grohmann et al. // International Conference on Measurement, Modelling and Evaluation of Computing Systems. – Springer, Cham, 2020. – P. 82–98.

5. Моисеев, В.И. Детектирование дисциплины обслуживания очередей в Ethernet-коммутаторах / В.И. Моисеев // Математика и междисциплинарные исследования 2018. – 2018. – С. 189–192.

6. Хабаров, С.П. Моделирование Ethernet сетей в среде OMNeT++ INET framework // Научно-технический вестник информационных технологий, механики и оптики / С.П. Хабаров. – 2018. – Т. 18. – № 3. – С. 462–472.

7. Nayyar, A.A Comprehensive Review of Simulation Tools for Wireless Sensor Networks (WSNs) / A. Nayyar, R. Singh // Journal of Wireless Networking and Communications. – 2015. – V. 5(1).

8. Темлюков, Д.А. Моделирование времени доставки пакетов данных с учетом очередей / Д.А. Темлюков, А.М. Андреев // Научное сообщество студентов XXI столетия. Технические науки. – 2018. – С. 124.

9. PacketScope: Monitoring the Packet Lifecycle Inside a Switch / R. Teixeira, R. Harrison, A. Gupta, J. Rexford //Proceedings of the Symposium on SDN Research. – 2020. – P. 76–82.

10. Моисеев, В.И. Экспериментальное исследование структуры пакетного буфера Ethernet коммутатора / В.И. Моисеев // T-Comm: Телекоммуникации и транспорт. – 2020. – Т. 14. – № 1. – С. 18–24.

11. Модель функционирования телекоммуникационного оборудования программно-конфигурируемых сетей / К.Е. Самуйлов, И.А. Шалимов, И.Г. Бужин, Ю.Б. Миронов // Современные информационные технологии и ИТ-образование. – 2018. – Т. 14. – № 1.

12. Ларкин, Е.В. О приближении потока событий к пуассоновскому / Е.В. Ларкин, Д.В. Горбачев, А.Н. Привалов // Чебышевский сборник. – 2017. – № 18(2). – С. 222–234.

13. An extension of the OMNeT++ INET framework for simulating real-time ethernet with high accuracy / T. Steinbach, H.D. Kenfack, F. Korf, T.C. Schmidt // Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques. – 2011. – P. 375–382.

14. Veselý, V. Multicast simulation and modeling in OMNeT++ / V. Veselý, P. Matousek, M. Svéda // SimuTools. – 2013. – P. 142–145

Болодурина Ирина Павловна, д-р техн. наук, профессор, заведующий кафедрой прикладной математики, Оренбургский государственный университет, г. Оребург; prmat@mail.osu.ru.

Ушаков Юрий Александрович, канд. техн. наук, доцент кафедры геометрии и компьютерных наук, Оренбургский государственный университет; unpk@mail.ru.

Ушакова Маргарита Викторовна, старший преподаватель кафедры геометрии и компьютерных наук, Оренбургский государственный университет; m.v.ushakova@mail.ru.

Поступила в редакцию 7 марта 2020 г.

DOI: 10.14529/ctcr200301

METHOD OF OBTAINING GENERALIZED STATISTICAL CHARACTERISTICS OF NETWORK EQUIPMENT FOR CREATION OF SIMULATION MODELS

***I.P. Bolodurina**, prmat@mail.osu.ru,*
***Yu.A. Ushakov**, unpk@mail.ru,*
***M.V. Ushakova**, m.v.ushakova@mail.ru*
Orenburg State University, Orenburg, Russian Federation

The research is devoted to the problem of the development models of network equipment that precisely reproduce the operation of real equipment by the time characteristics.

Aim. This research is aimed at developing a methodology that allows you to quickly obtain models that statistically coincide in the behavior of the selected parameters with real measured equipment on existing open modeling systems.

Methods. To obtain the performance parameters and time characteristics of the real equipment, an approach was proposed, which includes experiment planning to compute the characteristics for a wide range of input traffic intensities. In addition, the approach describes the connection scheme and equipment configuration including external switch control to test the effect of various settings on the performance and packet processing times. It also includes the scripts to automate the creation of traf-

fic test sets with the required parameters of time distribution between the packets as well as the scheme of connection through the container services and receiving traffic dumps. The approach does not include the testing of the output queues operation because the goal was to investigate and transfer into the model processes, which occur before the frames are placed in the outgoing interface queues. An open-source Omnet++ software was used as a modeling environment.

Results. We studied the structure of its existing switch model and identified disadvantages that affect the accuracy and reliability of the results. In addition, we proposed an approach for implementing model delay lines based on the interpolation of time parameter tables obtained from real data for the required combination of traffic parameters and inbound interfaces. To test the developed approaches, a test bench on real equipment was assembled, and an experimental research plan was developed. Finally, an automated testing script was implemented according to a given plan. Testing was carried out in the conventional frame switching mode as well as in the switching mode based on OpenFlow tables. In addition, the time characteristics of the intermediate equipment were also obtained in order to take into account their influence on the experimental results. Based on them, a table of parameters was computed for the processing delay line in the model, and the experiment was conducted on the model in Omnet++. Since the obtained samples in the statistical analysis showed the types of distributions that are different from the 3 normal and exponential, the statistical Mann–Whitney U-test was used to compare the mean values.

Conclusion. The results obtained at a significance level of 0.05 confirmed the efficiency of this approach for models that more accurately describe the temporal characteristics of the traffic processing in network equipment.

Keywords: modeling, simulation model, switching, delays.

References

1. Mekky H., Hao F., Mukherjee S., Lakshman T.V., Zhang Z. Network Function Virtualization Enablement within SDN Data Plane. *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, 2017, pp. 1–9.

2. Kutuzov O.I., Tatarnikova T.M. [On the Simulation Paradigm Analysis]. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2017, vol. 17, no. 3, pp. 552–558. (in Russ.)

3. Akatov D.V., Yurochkin A.G. [The Characteristics of the Main Tools for Analysis and Optimization of Enterprise Networks]. *Modeling, Optimization and Information Technology*, 2015, no. 2, pp. 8-8. (in Russ.)

4. Herrnleben S., Rygielski P., Grohmann J., Eismann S., Hobfeld T., Kounev S. Model-Based Performance Predictions for SDN-Based Networks: A Case Study. *International Conference on Measurement, Modelling and Evaluation of Computing Systems*, 2020, pp. 82–98.

5. Moiseev V.I. [Detecting Ethernet Switch Buffer Scheduling Discipline]. *Mathematics and Interdisciplinary Studies-2018*, 2018, pp. 189–192. (in Russ.)

6. Khabarov S.P. [Modeling of Ethernet Networks in OMNeT ++ INET Framework Medium]. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2018, vol. 18, no. 3, pp. 462–472 (in Russ.)

7. Nayyar A.A., Singh R. Comprehensive Review of Simulation Tools for Wireless Sensor Networks (WSNs). *Journal of Wireless Networking and Communications*, 2015, vol. 5(1), pp. 19–47.

8. Temlukov D.A., Andreev A.M. [Queuing Simulation of Data Packet Delivery Time]. *The scientific community of students of the XXI century. Technical science,* 2018, pp. 124–124 (in Russ.)

9. Teixeira R., Harrison R., Gupta A., Rexford J. PacketScope: Monitoring the Packet Lifecycle Inside a Switch. *Proceedings of the Symposium on SDN Research*, 2020, pp. 76–82.

10. Moiseev V.I. [Experimental Evaluation of Ethernet Switch Packet Buffer Structures]. *T-Comm*, 2020, vol. 14, no. 1, pp.18–24. (in Russ.)

11. Samouylov K.E, Shalimov I.A., Buzhin I.G., Mironov Yu.B. [Model of Functioning of Telecommunication Equipment for Software-Configurated Networks]. *Modern Information Technologies and IT-Education*, 2018, vol. 14, no. 1, pp. 13–26. (in Russ.)

12. Larkin E.V., Gorbachev D.V., Privalov A.N. [On the Approximation of the Flow of Events for a Poisson]. *Chebyshevskii Sbornik*, 2017, vol. 18(2), pp. 222–234. (in Russ.)

13. Steinbach T., Kenfack H.D., Korf F., Schmidt T.C. An Extension of the OMNeT++ INET Framework for Simulating Real-Time Ethernet with High Accuracy. *Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques*, 2011, pp. 375–382.

14. Veselý V., Matousek P., Svéda M. Multicast Simulation and Modeling in OMNeT++. *SimuTools*, 2013, pp. 142–145.

Received 7 March 2020

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Болодурина, И.П. Методика получения обобщенных статистических характеристик сетевого оборудования для создания имитационных моделей / И.П. Болодурина, Ю.А. Ушаков, М.В. Ушакова // Вестник ЮУрГУ. Серия «Компьютерные технологии, управление, радиоэлектроника». – 2020. – Т. 20, № 3. – С. 5–17. DOI: 10.14529/ctcr200301

FOR CITATION

Bolodurina I.P., Ushakov Yu.A., Ushakova M.V., Method of Obtaining Generalized Statistical Characteristics of Network Equipment for Creation of Simulation Models. *Bulletin of the South Ural State University. Ser. Computer Technologies, Automatic Control, Radio Electronics*, 2020, vol. 20, no. 3, pp. 5–17. (in Russ.) DOI: 10.14529/ctcr200301