УДК 004.72
DOI: 10.17586/0021-3454-2023-66-4-285-296

МЕТОД ПЕРЕДАЧИ ДАННЫХ В ПРОГРАММНО-КОНФИГУРИРУЕМЫХ СЕТЯХ С ИСПОЛЬЗОВАНИЕМ ПЛАНИРОВЩИКА И КОНТРОЛЯ ДОСТАВКИ

К. И. Никишин

Пензенский государственный университет, Пенза, Россия,
nkipnz@mail.ru

Аннотация. Предложен метод передачи данных с использованием планировщика и функции контроля доставки в программно-конфигурируемых сетях, разработан алгоритм его работы. Рассмотрен возможный вариант аппаратной реализации коммутатора OpenFlow, выполняющего обработку и передачу разнородного трафика. Описаны особенности функционирования основных узлов коммутатора — планировщика расписания и валидатора времени доставки. На основе предложенного метода проведено моделирование программно-конфигурируемых сетей с помощью аппарата сетей Петри. Исследованы вероятностно-временные характеристики модели, проведена верификация метода с использованием сетей Петри. Эффективность предложенного метода заключается в том, что загрузка коммутатора остается постоянной за счет более гибкой настройки расписания планировщика (вместо использования жестких тайм-аутов), а также возможности досрочной передачи стандартных данных.

Ключевые слова: *программно-конфигурируемые сети, OpenFlow, коммутатор, Ethernet, трафик, режим реального времени, планировщик расписания, контроль доставки кадров, сети Петри*

Ссылка для цитирования: *Никишин К. И.* Метод передачи данных в программно-конфигурируемых сетях с использованием планировщика и контроля доставки // Изв. вузов. Приборостроение. 2023. Т. 66, № 4. С. 285—296. DOI: 10.17586/0021-3454-2023-66-4-285-296.

METHOD OF DATA TRANSMISSION IN SOFTWARE-DEFINED NETWORKS USING A SCHEDULER AND RATE CONTROL

K. I. Nikishin

Penza State University, Penza, Russia
nkipnz@mail.ru

Abstract. A method of data transmission using a scheduler and a delivery control function in software-defined networks is proposed, and an algorithm for its operation is developed. A possible variant of the hardware implementation of the OpenFlow switch, which performs the processing and transmission of heterogeneous traffic, is considered. The features of the functioning of the main nodes of the switch - the scheduler and the delivery time validator - are described. On the basis of the proposed method, modeling of software-defined networks using the apparatus of Petri nets is carried out. The probabilistic-temporal characteristics of the model are studied, and the method is verified using Petri nets. The effectiveness of the proposed method lies in the fact that the switch load remains constant due to more flexible scheduler settings (instead of using hard timeouts), as well as the possibility of early transmission of standard data.

Keywords: software-defined networks, OpenFlow, switch, Ethernet, traffic, real-time mode, scheduler, frames delivery control, Petri nets

For citation: Nikishin K. I. Method of data transmission in software-defined networks using a scheduler and rate control. *Journal of Instrument Engineering.* 2023. Vol. 66, N 4. P. 285—296 (in Russian). DOI: 10.17586/0021-3454-2023-66-4-285-296.

В современном мире информационные технологии играют ключевую роль. Совершенствуются и создаются новые программные продукты, улучшая ту или иную сферу современного общества. Главной транспортной составляющей в работе любого программного продукта, автоматизированной системы, Интернета является компьютерная сеть.

Исторически первой сетью является классическая Ethernet [1]. Для приоритизации разнородного трафика в сети используется технология Quality of Service, QoS [2—4]. Однако, как и любой программный продукт, компьютерные сети стремительно развиваются. Возникают прорывные технологии, которые будут играть значимую роль, к такой категории могут относиться распределенные сети. Причиной широкого применения и развития распределенных сетей стало ограничение временных характеристик классической компьютерной сети Ethernet согласно стандарту IEEE 802.1*. Кроме этого, в современных условиях накладываются новые ограничения по передаче данных в режиме реального времени, разбросу средней задержки на выходе коммутатора (джиттер) и отказоустойчивости сети.

К распределенным сетям относятся Time-Triggered Ethernet [5, 6], программно-конфигурируемые сети (ПКС), облачные вычисления. Сетевое оборудование, используемое для обработки потоков данных, должно обладать свойствами мобильности, быстроты, простоты администрирования. Всем этим требованиям отвечают ПКС [7, 8]. В ПКС выделяют уровни: приложений, управления и инфраструктурный. Основным элементом ПКС является контроллер, ведущий обработку данных и рассчитывающий оптимальные маршруты для их передачи в коммутаторы. Обмен данными производится через протокол OpenFlow [7—9].

Входящий поток поступает в работающий по протоколу OpenFlow коммутатор ПКС, в одной из таблиц потоков которого выполняется поиск соответствующих наборов значений потока. Таблица потоков включает поля стандарта IEEE 802.1, приоритета, счетчики, инструкции, тайм-ауты.

Однако передача классическим методом в ПКС разнородных потоков данных — в реальном масштабе времени и стандартных данных (иначе — эластичного, или стохастического, трафика) [10—12] характеризуется рядом недостатков:

1) аппаратные ресурсы распределенной компьютерной сети тратятся на сбор, вычисление и установку тайм-аутов (предложенный в настоящей статье метод лишен этого недостатка);

2) значительное время поиска правила в таблицах потоков протокола OpenFlow. Из-за увеличения времени поиска о необходимости удаления потока из ПКС контроллер информируется на более позднем этапе. Таким образом, необходимо выполнять поиск тайм-аутов для потока в таблицах потоков, пока не будет найдено нужное правило или будет удален поток из ПКС.

Кроме этого, вследствие различия архитектур и топологий ПКС контроллер также может медленно реагировать на сигнал удаления потока из ПКС. В таком случае время на принятие решения для повторной передачи данных может быть увеличено, поскольку в передаче участвуют уже несколько контроллеров;

3) отсутствует контроль функций доставки данных на входных портах коммутаторов OpenFlow. Из-за этого затруднительно определить возможность доставки данных в необходимые моменты времени;

4) возникает необходимость прерывания потока стандартных данных в случае временного конфликта между различными видами трафика в коммутаторе. Это способствует уменьшению пропускной способности сети из-за повторной передачи прерванного потока стандартных данных.

* https://ru.wikipedia.org/wiki/IEEE_802.1Q.

По результатам анализа классического метода разработан метод передачи данных с использованием планировщика и функцией контроля доставки в ПКС. Предложенный метод устраняет недостатки, аппаратно упрощаются таблицы потоков и их структура.

Согласно разработанному методу, в каждый коммутатор ПКС вводится специальный аппаратный планировщик трафика, который обеспечивает своевременную доставку данных пользователю или конечному оборудованию. За счет досрочной передачи стандартных данных усовершенствуется алгоритм диспетчера очередей коммутатора.

Диспетчеры очередей в классических коммутаторах Ethernet и ПКС не могут в один момент времени передавать различные виды данных. Предложенный метод, если позволяет длина кадра, дает возможность передавать стандартные данные досрочно — во время блокировки кадром реального времени.

Аппаратный планировщик трафика (рис. 1) состоит из таблицы расписания планировщика, усовершенствованного узла классификации разнородных данных и узла валидации времени доставки кадров реального времени.

Рис. 1

Классический классификатор коммутатора в зависимости от приоритета направляет кадр в соответствующую очередь, в то время как модифицированный классификатор совместно с остальными узлами планировщика трафика позволяет передавать кадр в выходной порт коммутатора, в очереди кадров или удалить его.

Планировщик расписания добавляется в каждый коммутатор OpenFlow, он управляет входящим трафиком. Рассылка данных и настройка расписания выполняются на уровне приложения через API и функции ПКС. Таким образом, в статье используется заранее подготовленное синхронизированное расписание для каждого коммутатора ПКС.

Таблицы потоков упрощаются за счет удаления функций контроля тайм-аутов. Трафик направляется через входящий порт коммутатора к таблицам потоков, где выполняется поиск правила для потока, при этом некоторые поля найденного правила могут быть недоступны для чтения (замаскированы).

Если в заданной таблице для потока не найдено правило, то он переходит в следующую подсеть Петри таблицы потоков, где происходят дальнейший поиск и сравнение с правилами. Входящий поток, для которого после поиска по всем таблицам не будет обнаружено правило, удаляется из сети и об этом информируется контроллер.

Если правило для потока найдено, в работу включается планировщик расписания трафика. Планировщик расписания классифицирует принятый поток в зависимости от типа трафика: реального времени или стандартных данных. При этом для каждого потока реального времени фиксируется время прибытия в коммутатор OpenFlow.

После этого поток реального времени направляется в узел валидатора времени доставки потока, где время прибытия потока сравнивается с указанным в расписании. При этом валидатор времени доставки также может удалить поток реального времени, если время его прибытия превысит заранее известное время из расписания.

Узел дешифратора инструкций может выполнить одну из команд с потоком реального времени: направить непосредственно в выходной порт (канал) коммутатора OpenFlow или поместить поток в специально отведенную для него очередь.

Если совпадают время прибытия кадра реального времени и время доставки из расписания, выполняется передача принятого кадра в выходной порт коммутатора. Если время

прибытия кадра реального времени меньше времени доставки из расписания, он помещается в отдельную очередь.

Поток стандартных данных распределяется в отдельные очереди с помощью узла классификации. Валидатор времени доставки с помощью таймера определяет наступление очередного момента времени доставки для потока реального времени. Если время доставки не наступило, то управление передается диспетчеру очередей, который начинает работать только при наличии свободного выходного канала и при загруженных потоками очередях.

В коммутаторах Ethernet основным элементом управления является диспетчер очередей, что обусловливает необходимость разработки эффективных алгоритмов диспетчеризации с учетом многокритериальных параметров управления диспетчерами коммутатора [13]. К наиболее эффективным относятся различные циклические алгоритмы (Round Robin, RR) и взвешенные справедливые (Weighted Fair Queuing, WFQ).

К эффективным относится новый алгоритм с временной селекцией кадров (Time Selection Service, TSS), показатели эффективности при управлении очередями которого подробно описаны в статьях [14, 15]. Его основное отличие заключается в учете времени ожидания кадров в очередях и ограничении времени обслуживания очередей, что, в свою очередь, приводит к уменьшению разброса среднего значения задержки в сети.

Предложенный в настоящей статье метод использует другой алгоритм диспетчеризации (рис. 2) в коммутаторе OpenFlow, разработанный ранее для распределенной сетевой технологии Time-Triggered Ethernet [16], но в ином временном масштабе. Диспетчер проверяет возможность передачи потока стандартных данных, прежде чем наступит момент доставки очередного потока реального времени:

$$T_{\text{тек}} + T_{\text{ст}} \le T_{\text{бл}} + T_{\text{рв}},$$

где $T_{\text{тек}}$ — текущее время в системе, $T_{\text{ст}}$ — время передачи потока стандартных данных с учетом длины кадров, $T_{\text{бл}}$ — время блокировки, $T_{\text{рв}}$ — время передачи потока реального времени.

Входящими параметрами для диспетчера очередей являются размер (длина) кадров в потоке и время поступления потока в очередь для каждого из видов трафика. Исходя из этих параметров диспетчер проверяет следующее условие: сумма текущего момента времени системы и времени передачи потока стандартных данных с учетом длины кадров должна быть меньше времени доставки очередного кадра реального времени. Учитывается, за какое время коммутатор сможет передать поток на выходной порт с учетом длины кадров.

Если заданное условие выполняется, то поток стандартных данных передается из соответствующей очереди, иначе — ожидается момент доставки очередного потока реального времени, и когда этот момент наступает, осуществляется передача потока из выбранной очереди в выходной канал.

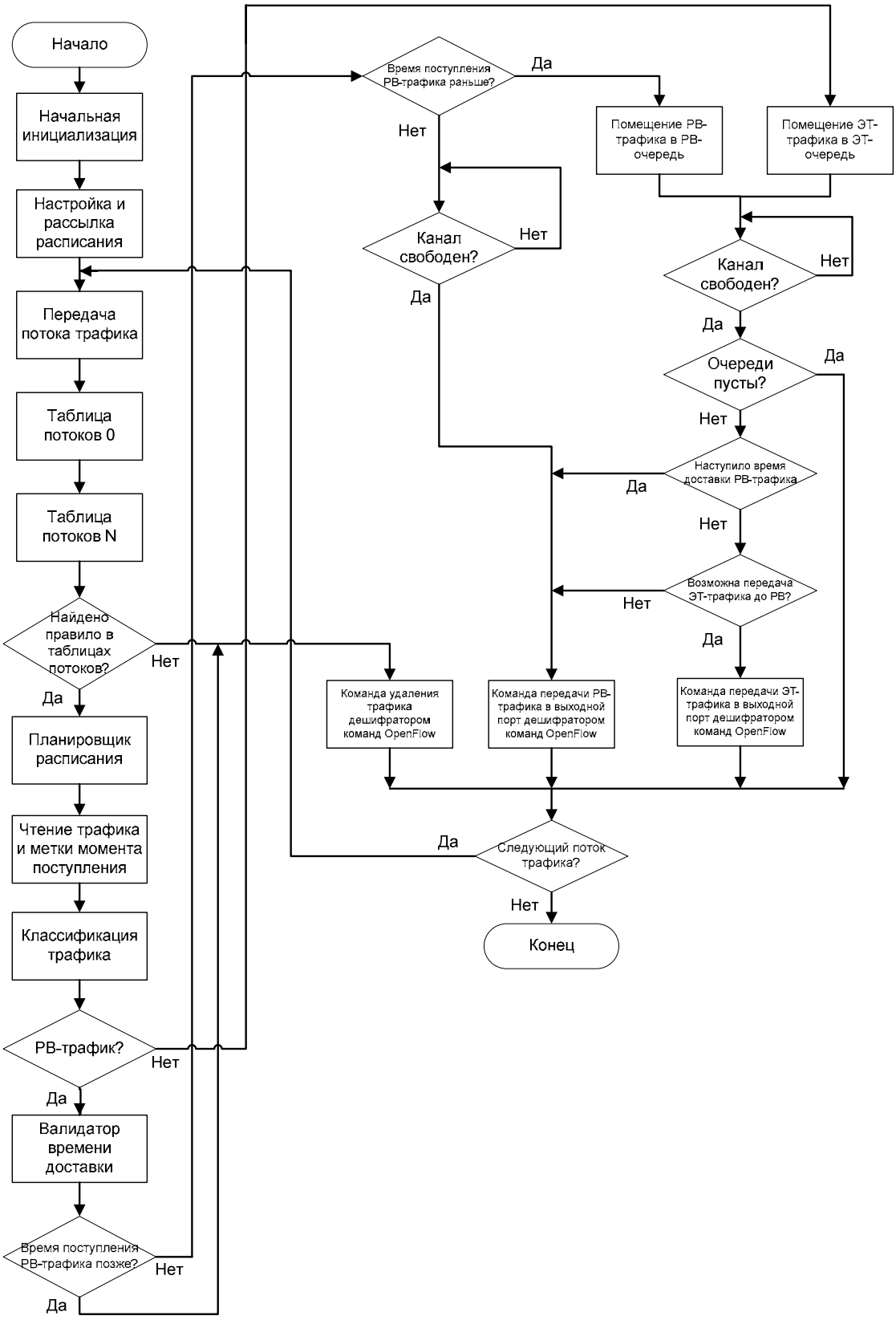

Рис. 2

На рис. 3 показан возможный вариант реализации архитектуры коммутатора на основе OpenFlow, выполняющего обработку и передачу разнородного трафика. Основные узлы аппаратной архитектуры: *1* — шина приема кадров из аппаратуры продвижения и передачи в порт назначения коммутатора; *2* — валидатор времени доставки данных; *3* — планировщик

расписания потока реального времени; *4* — таблица потоков 0; *5* — классификатор поступающего трафика и канал прямой записи в память; *6* — таблица потоков *N*; *7* — шина чтения кадров из очередей; *8* — шина передачи кадров в очередь или на выходной порт; *9* — процессор диспетчеризации; *10* — сигнал „очереди свободны“; *11* — память типа FIFO для приема стандартных данных; *12* — память типа FIFO для приема потока кадров реального времени; *13* — дешифратор инструкций коммутатора OpenFlow; *14* — канал прямого чтения кадров из очередей или шины передачи кадров на выходной порт; *15* — физический интерфейс выходного порта; *16* — сигнал „выходной порт свободен“; *17* — шина выходного канала; *18* — контроллер ПКС.

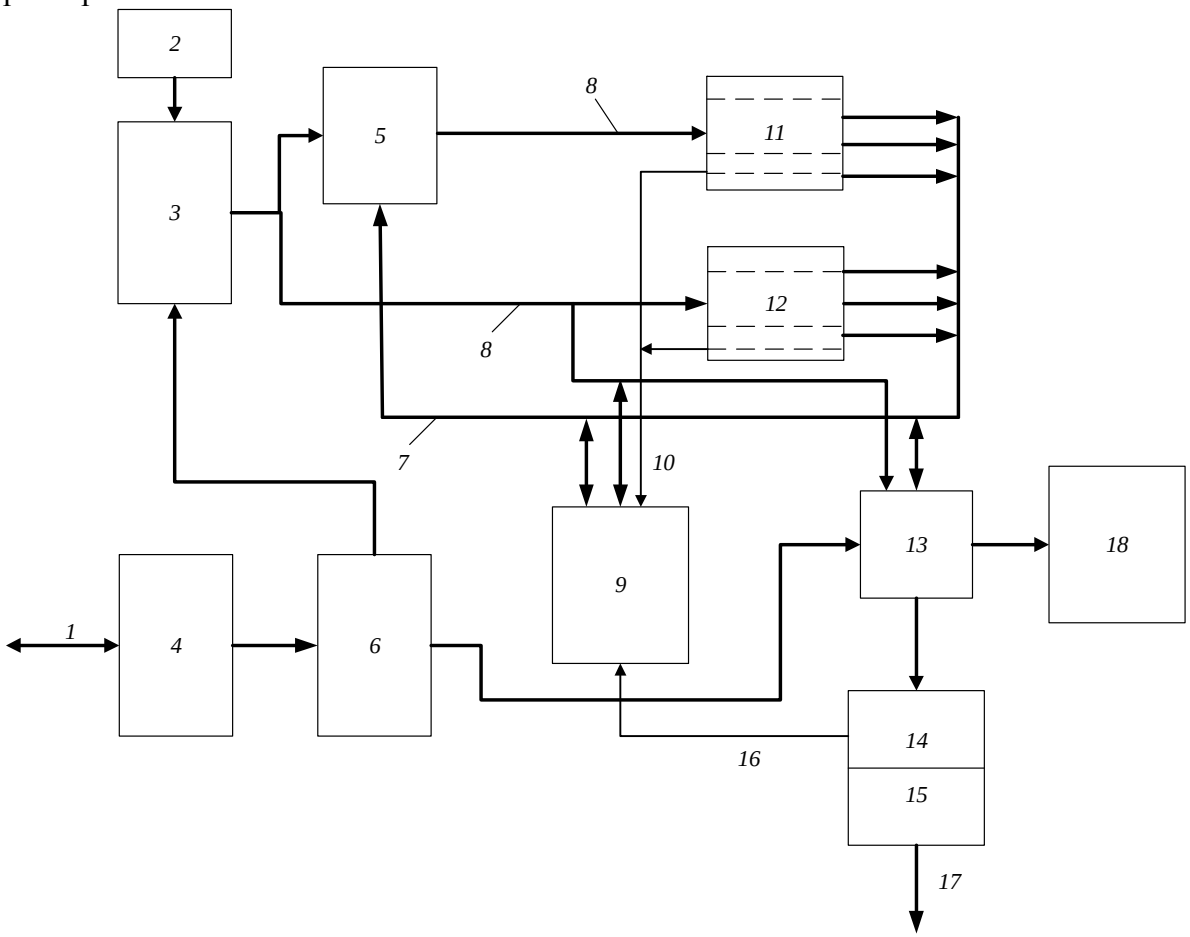

Рис. 3

Пример функционирования коммутатора, реализующего предложенный метод, представлен ниже.

В порт назначения коммутатора OpenFlow по шине *1* поступают потоки (кадры) из аппаратуры продвижения кадров, которые могут быть направлены в данный порт.

Выполняется поиск правила для входящего потока во всех таблицах потоков. Таблиц потоков может быть несколько — от 0 до *N*. Вначале выполняется поиск правила в таблице потоков 0 (на рис. 3 узел *4*), затем — в таблице потоков *N* (на рис. 3 узел *6*). Если для входящего потока не будет обнаружено правило, то информация передается дешифратору инструкций коммутатора OpenFlow *13*. Дешифратор инструкций совместно с контроллером ПКС *18* принимает решение об удалении потока из ПКС. В случае успешного поиска правила включается в работу планировщик расписания трафика *3*.

Текущий момент поступления потока кадров реального времени сравнивается с моментом доставки в планировщике расписания *3*. Валидатор времени доставки потока *2* сравнивает время прибытия потока с заранее установленным значением момента доставки.

Управляющие логические элементы и таймер сравнения времени являются основными составляющими узла *2*. Планировщик расписания представляет собой регистровую память (набор регистров), в которой хранятся таблицы расписаний со значениями моментов доставки кадров реального времени. Каждый коммутатор OpenFlow должен содержать свой планировщик расписания.

Планировщик расписания может: передавать поток стандартных данных в классификатор *5*; поместить кадры реального времени в очередь для данного трафика (текущий момент прибытия этого потока меньше момента его доставки); направить кадры реального времени сразу же в выходной канал *17* (текущий момент прибытия трафика меньше равен моменту его доставки). За выполнение инструкций отвечает дешифратор инструкций коммутатора *13* согласно протоколу OpenFlow.

Классификатор *5* разделяет потоки стандартных данных в зависимости от типа трафика и помещает в соответствующие очереди.

Потоки записываются по шине *8*, а считываются из очередей по шине *7*. Валидатор времени доставки потока *2* постоянно проверяет наступление момента доставки кадров реального времени и передает управление процессору диспетчеризации *9*. Он запускается сигналами *8* и *13*, когда очереди не пусты и выходной канал свободен. Осуществляется передача кадров в выходной канал через дешифратор инструкций *13*.

С помощью предложенного метода проведено моделирование компьютерной сети. Для этих целей выбран математический аппарат сетей Петри, выбраны цветные временные иерархические сети Петри [17], а в качестве инструмента построения таких сетей — пакет CPN Tools. Данный пакет хорошо зарекомендовал себя при исследовании телекоммуникационных и компьютерных сетей, их алгоритмов, протоколов, различного оборудования [18, 19].

На рис. 4 представлен только один узел коммутатора OpenFlow на основе сетей Петри — планировщик расписаний. В подсети Петри выполняется передача входящего потока для дальнейшей обработки в коммутаторе. В частности, поток будет направлен к подсети классификатора коммутатора. В этой подсети генерируется таблица расписаний со своей собственной структурой для входящего потока.

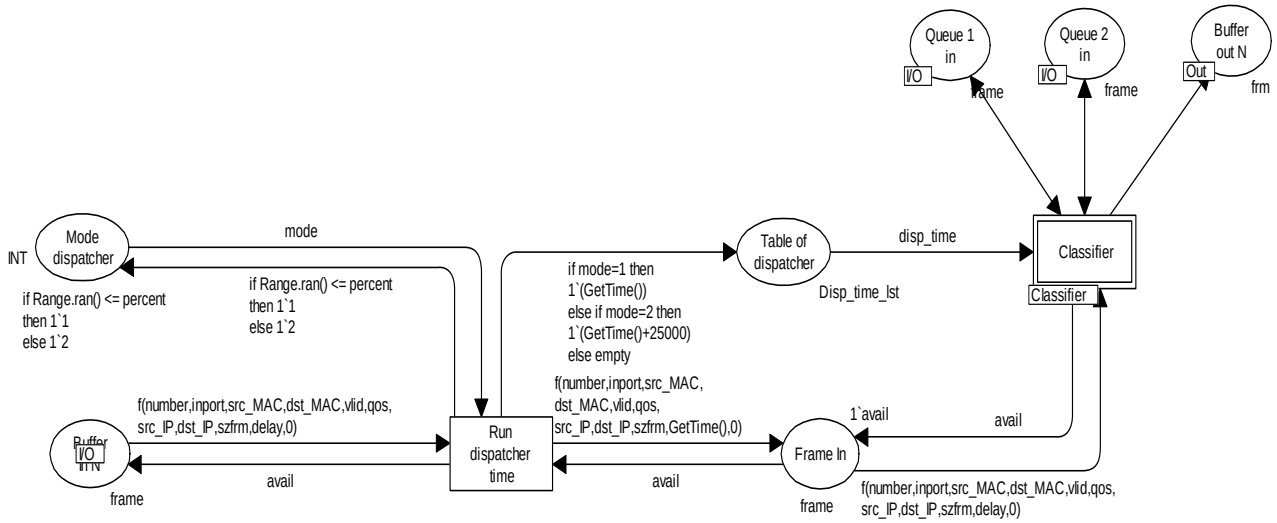

Рис. 4

Таблица расписаний также направляется в подсеть классификатора коммутатора. В дальнейшем в работу включается подсеть валидатора времени доставки потока. Модели классического метода в ПКС на сетях Петри описаны в статьях [20—22]; модель предложенного метода, алгоритмы его функционирования на сетях Петри подробно описаны в статье [23]; в [24] рассмотрена модель ранней диагностики потерь данных в ПКС на основе сетей Петри.

Было проведено экспериментальное исследование моделей классического и предложенного методов в коммутаторе OpenFlow с разной загрузкой, через коммутатор передавались 3436, 5300 и 7500 кадров. Загрузка коммутатора определяется соотношением общей суммы всех переданных битов (байтов) в коммутаторе к итоговому времени передачи всех кадров. Поэтому для загрузки коммутатора 0,4 о.е. требовалось передать 3436 кадров, аналогичным образом вычислялась загрузка коммутатора и с другим количеством кадров. При анализе результатов экспериментов выявлено увеличение загрузки коммутатора, потому что кадры в модели удаляются классическим методом. Потери возникают вследствие использования тайм-аутов по протоколу OpenFlow, когда время прибытия потоков превышает границы тайм-аутов.

В таких случаях происходит удаление потока из ПКС, выполняется инструкция удаления потока дешифратором инструкций OpenFlow и информирование контроллера об удалении потока. В свою очередь, контроллер принимает решение о повторной передаче в ПКС. Это приводит к увеличению загрузки коммутатора за счет повторной передачи: при загрузке коммутатора 0,8 о.е. появляется критическая доля удаленных кадров, приводящая к большей загрузке коммутатора (0,88 о.е.) и к неустойчивости работы коммутатора. Результаты моделирования классического и предложенного методов представлены в табл. 1 и 2.

Таблица 1

Результаты моделирования классического метода

Характеристика	Исходная загрузка коммутатора, о.е.		
	0,4	0,6	0,8
Общее число поступивших кадров	3436	5300	7500
Число удаленных кадров	231	429	738
Размер переданных кадров, бит	24824385	36986102	55221301
Размер удаленных кадров, бит	1434880	2812698	4644378
Загрузка коммутатора с учетом удаленных кадров, о.е.	0,43	0,65	0,88

Таблица 2

Результаты моделирования предложенного метода

Характеристика	Исходная загрузка коммутатора, о.е.		
	0,4	0,6	0,8
Общее число кадров	3436	5300	7500
Число досрочно переданных кадров	342	687	1227
Размер переданных кадров, бит	24824385	36986102	55221301
Размер досрочно переданных кадров, бит	2158091	4204640	6738654
Результирующая загрузка коммутатора, о.е.	0,4	0,6	0,8

Таблицы строились на основе статистических данных экспериментов в работе с моделями. В статистику включались следующие показатели: итоговое количество кадров, размер передаваемых кадров, количество удаленных кадров, размер удаленных кадров, входящее время поступления кадров в коммутатор и выходящее время передачи кадров из коммутатора.

Проводилась серия опытов с разным количеством передаваемых кадров и требуемой загрузкой. На вход коммутатора передавался трафик с различными характеристиками, такими как длина межкадрового интервала, интенсивность поступления потока стандартных данных, соотношение длины кадров и приоритетов, величина постоянного периода.

Данная статистика собиралась с помощью мониторов — встроенного инструмента в пакете CPN Tools. Показатели сети сохранялись в текстовые файлы, которые далее были обработаны с помощью Excel.

На рис. 5 представлен график распределения количества удаленных кадров из ПКС ($K_у$) при увеличении интенсивности поступления кадров с K = 3436 до 7500 [18]. На рис. 6 пред-

ставлена гистограмма сравнения загрузки коммутатора OpenFlow двумя методами процесса передачи разнородного трафика ($Z_н$ — номинальная загрузка; $Z_р$ — рабочая загрузка).

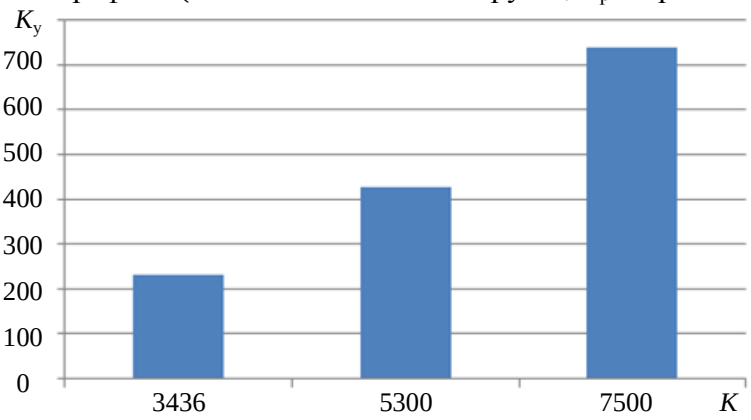

Рис. 5

При этом эффективность предложенного метода заключается в том, что загрузка коммутатора не повышается благодаря гибкому расписанию планировщика, а не наличию жестких тайм-аутов. Кроме этого, возможна передача стандартных данных до того, пока не будет передан кадр реального времени. Как можно увидеть из табл. 1 и рис. 6, в классическом методе увеличивается загрузка коммутатора на 11 %. При дальнейшем увеличении числа кадров в ПКС могут возникнуть перебои в работе коммутатора, приводя к пиковой загрузке на уровне 0,9—0,95.

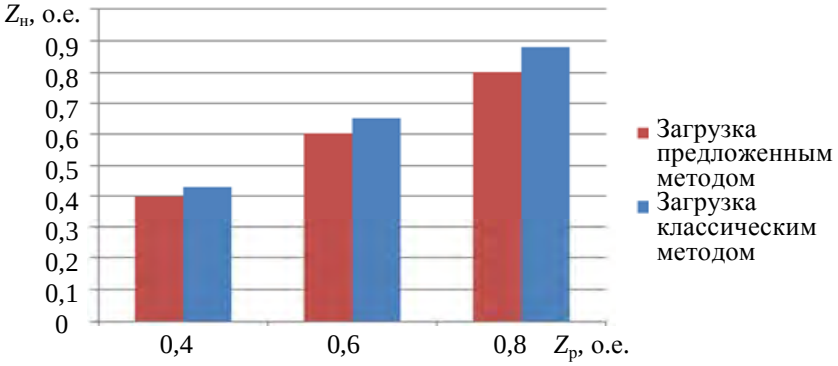

Рис. 6

На рис. 7 показаны общее число $K_д$ переданных досрочно стандартных кадров, а также доля $D_д$ таких кадров, наибольшая эффективность предложенного метода достигается при рабочей загрузке коммутатора 0,8 о.е. Таким образом, на 17 %, по сравнению с классическим методом, увеличивается возможность досрочной передачи стандартных данных.

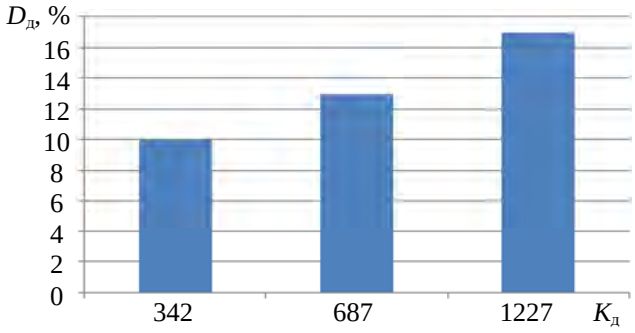

Рис. 7

Поэтому эффективность предложенного метода заключается не только в том, что загрузка коммутатора остается постоянной, но также обеспечивается возможность досрочной передачи стандартных данных во время блокировки коммутатора потоком кадров реального времени. Таким образом, коммутатор может передавать кадры стандартных данных во время ожидания

доставки кадров реального времени. Это позволяет повысить пропускную способность сети и снизить задержку, все эти показатели играют решающую роль при передаче данных по сети для конечного пользователя.

Согласно результатам моделирования предложенного метода, при увеличении интенсивности поступления кадров вероятность их потери ниже, поскольку осуществляется гибкая настройка расписания планировщика. Однако большая интенсивность поступления кадров может привести к перегрузке коммутатора, поэтому в статье рассмотрены рабочий и оптимальный варианты загрузки коммутатора.

Предложен метод передачи потоков данных с использованием планировщика и функцией контроля доставки в ПКС, разработан алгоритм его работы. Детально описаны особенности функционирования планировщика расписания и валидатора времени доставки.

Рассмотрен возможный вариант аппаратной реализации коммутатора на основе OpenFlow, выполняющего обработку и передачу разнородного трафика.

На основе предложенного метода проведено моделирование ПКС с помощью аппарата сетей Петри. Исследованы вероятностно-временные характеристики модели, проведена верификация метода с использованием сетей Петри.

Преимущества предложенного метода заключаются в том, что (1) загрузка коммутатора остается постоянной благодаря более гибкой настройке расписания планировщика, а не за счет жестких тайм-аутов; (2) обеспечивается возможность досрочной передачи потока стандартных данных, пока не наступило время доставки кадров реального времени.

СПИСОК ЛИТЕРАТУРЫ

1. *Олифер В. Г., Олифер Н. А.* Компьютерные сети. Принципы, технологии, протоколы. 4-е изд. СПб: Питер, 2010. 943 с.

2. *Karakus M., Durresi A.* Quality of service (QoS) in software defined networking (SDN): A survey // Journal of Network and Computer Applications. 2017. Vol. 80. P. 200—218.

3. *Перепелкин Д. А., Бышов В. С.* Балансировка потоков данных в программно-конфигурируемых сетях с обеспечением качества обслуживания сетевых сервисов // Радиотехника. 2016. № 11. С. 111—119.

4. *Никишин К. И.* Механизм управления трафиком реального времени в коммутаторе Ethernet // Вестник компьютерных и информационных технологий. 2015. № 10. С. 32—37.

5. *Nikishin K., Konnov N.* Schedule Time-Triggered Ethernet // 2020 International Conference on Engineering Management of Communication and Technology (EMCTECH). Vienna, Austria, 2020. P. 1—5. DOI: 10.1109/EMCTECH49634.2020.9261540.

6. *Nikishin K., Konnov N., Pashchenko D.* Modelling of systems using Time-Triggered Ethernet // Springer Information Technologies and Mathematical Modelling — Queueing Theory and Applications. 2017. Vol. 638. Ser. Communications in Computer and Information Science. P. 303—314.

7. *McKeown N., Anderson T., Balakrishnan H.* et al. Openflow: enabling innovation in campus networks // ACM SIGCOMM Computer Communication Review. 2008. Vol. 38, N 2. P. 69—74.

8. *Shalimov A.* et al. Advanced study of SDN/OpenFlow controllers // Proc. of the 9th Central & Eastern European Software Engineering Conf. in Russia. ACM, 2013.

9. *Kobayashi M., Seetharaman S., Parulkar G., Appenzeller G., Little J., Van Reijendam J., McKeown N.* Maturing of OpenFlow and Software-Defined Networking Through Deployments // Computer Networks. 2014. Vol. 61. P. 151—175.

10. *Maniu R. and Dumitru L. A.* Exploring the possibilities of a self‑ regulating SDN controller // Scientific Bulletin „Mircea cel Batran“ Naval Academy. 2015. Vol. 18, N 1. P. 58—61.

11. *Перепелкин Д. А.* Концептуальный подход динамического формирования трафика программно-конфигурируемых телекоммуникационных сетей с балансировкой нагрузки // Информационные технологии. 2015. Т. 21, № 8. С. 602—610.

12. *Ren H., Li X., Geng J.* A SDN‑based dynamic traffic scheduling algorithm // IEEE Intern. Conf. on Cyber‑Enabled Distributed Computing and Knowledge Discovery (CyberC). Chengdu, China, 2016. DOI: 10.1109/CyberC.2016.103.

13. *Механов В. Б., Кизилов Е. А.* Моделирование цветными сетями Петри обслуживания очередей алгоритмом WRR // Тр. IX. Междунар. науч.-техн. конф. „Новые информационные технологии и системы“. Ч. 1. Пенза: Изд-во ПГУ, 2010. С. 67—73.

14. *Kizilov E., Konnov N., Nikishin K., Pashchenko D., Trokoz D.* Scheduling queues in the Ethernet switch, considering the waiting time of frames // MATEC Web of Conferences. 2016. Vol. 44. P. 01011-p.1—01011-p. 5.

15. *Кизилов Е. А., Коннов Н. Н., Механов В. Б., Никишин К. И.* Учет времени поступления кадров для управления очередями в коммутаторе // Телематика-2014: тр. XXI Всерос. науч.-метод. конф. СПб: СПбГУ ИТМО, 2014. С. 134—136.

16. *Никишин К. И., Коннов Н. Н., Гурин Е. И.* Усовершенствованный механизм передачи трафика жесткого реального времени в сети Ethernet // Изв. вузов. Поволжский регион. Технические науки. 2018. № 4. С. 28—38.

17. *Jensen K., Kristensen L. M.* Coloured Petri Nets. Modelling and Validation of Concurrent Systems. Berlin: Springer, 2009. 384 p.

18. *Никишин К. И., Коннов Н. Н.* Генератор трафика Ethernet на основе цветных сетей Петри // Модели, системы, сети в экономике, технике, природе и обществе. 2016. № 1(17). С. 299—307.

19. *Никишин К. И.* Моделирование и верификация топологий программно-конфигурируемых сетей // Вестн. Рязанского государственного радиотехнического университета. 2022. № 80. С. 67—74. DOI: 10.21667/1995-4565-2022-80-67-74.

20. *Никишин К. И.* Моделирование контроллера и верификация процесса передачи данных в программно-конфигурируемых сетях // Вестн. Рязанского государственного радиотехнического университета. 2022. № 80. С. 75—83. DOI: 10.21667/1995-4565-2022-80-75-83.

21. *Никишин К. И.* Моделирование процесса передачи трафика в программно-конфигурируемых сетях // Вестн. Рязанского государственного радиотехнического университета. 2022. № 81. С. 32—41. DOI: 10.21667/1995-4565-2022-81-32-41.

22. *Никишин К. И.* Исследование и моделирование таблицы потоков коммутатора Openflow в программно-конфигурируемых сетях // Вестн. Рязанского государственного радиотехнического университета. 2022. № 81. С. 42—50. DOI: 10.21667/1995-4565-2022-81-42-50.

23. *Никишин К. И.* Моделирование процесса передачи трафика реального времени с использованием планировщика и функцией контроля доставки в программно-конфигурируемых сетях // Изв. СПбГЭТУ „ЛЭТИ“. 2023. Т. 16, № 1. С. 53—65. DOI: 10.32603/2071-8985-2023-16-1-53-65.

24. *Никишин К. И.* Моделирование метода ранней диагностики потерь трафика реального времени в программно-конфигурируемых сетях на основе аппарата сетей Петри // Вестн. Поволжского государственного технологического университета. Сер. „Радиотехнические и инфокоммуникационные системы“. 2022. № 2(54). С. 47—60. DOI: 10.25686/2306-2819.2022.2.4.

Сведения об авторе

Кирилл Игоревич Никишин — канд. техн. наук; Пензенский государственный университет, кафедра вычислительной техники; доцент; E-mail: nkipnz@mail.ru

Поступила в редакцию 28.10.22; одобрена после рецензирования 05.12.22; принята к публикации 28.02.23.

REFERENCES

1. Olifer V.G., Olifer N.A. *Komp'yuternyye seti. Printsipy, tekhnologii, protokoly* (Computer Networks. Principles, Technologies, Protocols), St. Petersburg, 2010, 943 p. (in Russ.)
2. Karakus M., Durresi A. *Journal of Network and Computer Applications*, 2017, vol. 80, pp. 200–218.
3. Perepelkin D.A., Byshov V.S. *Radioengineering*, 2016, no. 11, pp. 111–119. (in Russ.)
4. Nikishin K.I. *Vestnik komp'iuternykh i informatsionnykh tekhnologii* (Herald of Computer and Information Technologies), 2015, no. 10, pp. 32–37. (in Russ.)
5. Nikishin K., Konnov N. *2020 International Conference on Engineering Management of Communication and Technology (EMCTECH)*, Vienna, Austria, 2020, pp. 1–5, DOI: 10.1109/EMCTECH49634.2020.9261540.
6. Nikishin K., Konnov N., Pashchenko D. *Springer Information Technologies and Mathematical Modelling – Queueing Theory and Applications*, 2017, vol. 638, ser. Communications in Computer and Information Science, pp. 303–314.

7. McKeown N., Anderson T., Balakrishnan H. et al. *ACM SIGCOMM Computer Communication Review*, 2008, no. 2(38), pp. 69–74.
8. Shalimov A. et al. *Proceedings of the 9th Central & Eastern European Software Engineering Conference in Russia*, ACM, 2013.
9. Kobayashi M., Seetharaman S., Parulkar G., Appenzeller G., Little J., Van Reijendam J., McKeown N. *Computer Networks*, 2014, vol. 61, pp. 151–175.
10. Maniu R. and Dumitru L.A. *Scientific Bulletin „Mircea cel Batran" Naval Academy*, 2015, no. 1(18), pp. 58–61.
11. Perepelkin D.A. *Information Technologies* (Informacionnye Tehnologii), 2015, no. 8(21), pp. 602–610. (in Russ.)
12. Ren H., Li X., Geng J. *IEEE International Conference on Cyber‐Enabled Distributed Computing and Knowledge Discovery (CyberC)*, Chengdu, China, 2016, DOI: 10.1109/CyberC.2016.103.
13. Mekhanov V.B., Kizilov E.A. *Novyye informatsionnyye tekhnologii i sistemy* (New Information Technologies and Systems), Proceedings of the IX International Scientific and Technical Conference, Part 1, Penza, 2010, pp. 67–73. (in Russ.)
14. Kizilov E., Konnov N., Nikishin K., Pashchenko D., Trokoz D. *MATEC Web of Conferences*, 2016, vol. 44, pp. 01011-p.1–01011-p. 5.
15. Kizilov E.A., Konnov N.N., Mekhanov V.B., Nikishin K.I. *Telematika-2014* (Telematics-2014), Proceedings of the XXI All-Russian Scientific and Methodological Conference, St. Petersburg, 2014, pp. 134–136. (in Russ.)
16. Nikishin K.I., Konnov N.N., Gurin E.I. *University proceedings. Volga region. Technical sciences*, 2018, no. 4, pp. 28–38. (in Russ.)
17. Jensen K., Kristensen L.M. *Coloured Petri Nets. Modelling and Validation of Concurrent Systems*, Berlin, Springer, 2009, 384 p.
18. Nikishin K.I., Konnov N.N. *Models, Systems, Networks in Economics, Technology, Nature and Society*, 2016, no. 1(17), pp. 299–307. (in Russ.)
19. Nikishin K.I. *Vestnik of Ryazan State Radioengineering University*, 2022, no. 80, pp. 67–74, DOI: 10.21667/1995-4565-2022-80-67-74. (in Russ.)
20. Nikishin K.I. *Vestnik of Ryazan State Radioengineering University*, 2022, no. 80, pp. 75–83, DOI: 10.21667/1995-4565-2022-80-75-83. (in Russ.)
21. Nikishin K.I. *Vestnik of Ryazan State Radioengineering University*, 2022, no. 81, pp. 32–41, DOI: 10.21667/1995-4565-2022-81-32-41. (in Russ.)
22. Nikishin K.I. *Vestnik of Ryazan State Radioengineering University*, 2022, no. 81, pp. 42–50, DOI: 10.21667/1995-4565-2022-81-42-50. (in Russ.)
23. Nikishin K.I. *Saint Petersburg Electrotechnical University Journal*, 2023, no. 1(16), pp. 53–65, DOI: 10.32603/2071-8985-2023-16-1-53-65. (in Russ.)
24. Nikishin K.I. *Vestnik of Volga State University of Technology. Ser.: Radio Engineering and Infocommunication Systems*, 2022, no. 2(54), pp. 47–60, DOI: 10.25686/2306-2819.2022.2.4. (in Russ.)

Data on author

Kirill I. Nikishin — PhD; Penza State University, Department of Computer Science; Associate professor; E-mail: nkipnz@mail.ru

Received 28.10.22; approved after reviewing 05.12.22; accepted for publication 28.02.23.