Известия СПбГЭТУ «ЛЭТИ». 2023. Т. 16, № 1. С. 53–65
LETI Transactions on Electrical Engineering & Computer Science. 2023. Vol. 16, no. 1. P. 53–65

УДК 004.72 Научная статья

https://doi.org/10.32603/2071-8985-2023-16-1-53-65

Моделирование процесса передачи трафика реального времени с использованием планировщика и функцией контроля доставки в программно-конфигурируемых сетях

К. И. Никишин✉

Пензенский государственный университет, Пенза, Россия

✉ nkipnz@mail.ru

Аннотация. В статье исследована модель процесса передачи трафика реального времени с использованием планировщика и функцией контроля доставки в программно-конфигурируемых сетях (ПКС) на основе сетей Петри и пакета CPN Tools. Разработаны и описаны подсети Петри планировщика трафика, валидатора времени доставки; произведена модификация подсетей таблиц потоков, подсетей сравнения и поиска правила в таблице потоков, классификации трафика, очередей и диспетчера очередей. Проведено экспериментальное моделирование с различной загрузкой коммутатора, приведено сравнение результатов разработанного метода с классическим методом передачи разнородного трафика в ПКС. Согласно полученным данным метод процесса передачи трафика реального времени с использованием планировщика и функцией контроля доставки в ПКС доказал свою эффективность. Эффективность предложенного метода заключается в том, что загрузка коммутатора остается постоянной за счет более гибкой настройки расписания планировщика, а не жестких интервалов времени таймаутов, а также осуществляется возможность досрочной передачи эластичного трафика, пока не наступило время доставки трафика реального времени.

Ключевые слова: программно-конфигурируемые сети, OpenFlow, коммутатор, Ethernet, трафик реального времени, планировщик расписания, контроль доставки трафика, сети Петри, CPN Tools

Для цитирования: 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.

Original article

Modeling of a Process of Transmitting Real-Time Traffic Using a Scheduler and a Delivery Control Function in Software Defined Networks

K. I. Nikishin✉

Penza State University, Penza, Russia

✉ nkipnz@mail.ru

Abstract. The article is described a model of the real-time traffic transmission process using a scheduler and a delivery control function in software defined networks (SDN) based on Petri nets and the CPN Tools package. The Petri subnets of the traffic scheduler, the delivery time validator have been developed and described, the subnets of the flow tables have been modified, the subnets of the comparison and search rules in the flow table, traffic classification, queues and queue manager have been modified. Experimental modeling was carried out with different switch loads, and the results of the developed method were compared with the classical method of transmitting heterogeneous traffic to a SDN. According to the data obtained, the method of the process of transmitting real-time traffic using a scheduler and a delivery control function to a SDN has proven its effectiveness. The effectiveness of the proposed method lies in the fact that the load of the switch remains constant due to a more flexible configuration of the scheduler schedule, rather than rigid time intervals of timeouts, and the possibility of early transmission of elastic traffic is also realized until the time of delivery of real-time traffic has come.

Keywords: software defined networks, OpenFlow, switch, Ethernet, real-time traffic, scheduler, delivery control of traffic, Petri nets, CPN Tools

For citation: Nikishin K. I. Modeling of a Process of Transmitting Real-Time Traffic Using a Scheduler and a Delivery Control Function in Software Defined Networks // LETI Transactions on Electrical Engineering & Computer Science. 2023. Vol. 16, no. 1. P. 53–65. doi: 10.32603/2071-8985-2023-16-1-53-65.

Введение. Главной транспортной составляющей в работе любого программного продукта, автоматизированной системы, Интернета является компьютерная сеть [1]. Существует необходимость передавать по сети большие объемы информации. Самой распространенной сетевой технологией является Ethernet. Дальнейшим развитием Ethernet стало введение понятие качества обслуживания [2]. С помощью введения понятия качества обслуживания становится возможной передача разнородного трафика в зависимости от его приоритета, таким образом, в коммутаторе Ethernet происходит классификация трафика [3].

Дальнейшей парадигмой компьютерных сетей стало возникновение распределенных сетей – таких, как Time-Triggered Ethernet [4], программно-конфигурируемые сети (ПКС) [5]–[7], облачные вычисления. Причиной перехода к развитию распределенных сетей стало ограничение классической компьютерной сети Ethernet по дальнейшим разработкам и вариациям передачи трафика согласно стандарту IEEE 802.1. Тем самым накладываются новые ограничения по быстродействию трафика реального времени, разбросу средней задержки на выходе коммутатора (джиттер) и отказоустойчивости сети. Все эти причины привели к появлению ПКС [8]–[9].

Программно-конфигурируемые сети. В ПКС значительно упрощается администрирование сети, а также настраиваются сетевые параметры, метрики – задержка в сети, пропускная способность, управление очередями [10]–[12]. Управляются и администрируются ПКС на уровне приложений ПКС. Впервые идея ПКС была предложена в университетах Беркли и Стэнфорда в 2006 г.

Основные узлы в ПКС – это контроллер совместно с коммутатором, который работает по открытому протоколу OpenFlow. Протокол OpenFlow есть основная часть ПКС. Таким образом, открытый протокол OpenFlow успешно позволяет справляться с возросшими требованиями к пользовательскому трафику, приоритету [13].

Под разнородным пользовательским трафиком понимается сочетание различных видов трафика: трафика реального времени, для которого основное требование заключается в минимальной задержке кадра в сети и максимальное быстродействие, и стандартного (эластичного) трафика, к которому предъявляются гораздо меньшие требования.

К таким видам трафика можно отнести служебный трафик, который передает управляющие команды, команды синхронизации в сети, голос и видео, относящиеся к трафику реального времени, фоновый трафик, обычный трафик в LAN-сети и другие. Каждый вид трафика должен обладать приоритетом в формате кадра Ethernet согласно стандарту IEEE 801.2.

Методы процесса передачи разнородного трафика в ПКС. Однако передача трафика в коммутаторах по протоколу OpenFlow имеет ряд недостатков, основные из которых были рассмотрены и описаны в [14]. Рассмотрим некоторые из этих недостатков. В ходе передачи разнородного трафика коммутатор OpenFlow затрачивает значительное время на поиск правила для потока в таблицах потоков. Недостаток метода состоит в том, что данная ситуация приводит к информированию контроллера о необходимости удаления потока из ПКС на более позднем этапе.

Таким образом, необходимо выполнять поиск таймаутов для входящего потока в таблицах потоков, пока не будет найдено нужное правило или не будет удален поток из ПКС. В этом случае отсутствует быстрая передача разнородного трафика, поскольку тратятся значительные временные и аппаратные ресурсы для поиска таймаутов потока. Кроме этого, контроллер также может медленно реагировать на сигнал удаления потока из ПКС за счет различных архитектур и топологий ПКС, где возможна целая группа иерархий контроллеров.

В классическом методе процесса передачи трафика отсутствует контроль функций доставки трафика на входных портах коммутаторов OpenFlow. Недостаток заключается в том, что затруднительно определить доставку трафика реального времени получателю в конкретные моменты времени. Возникает необходимость получения трафика реального времени в строго отведенное время, поскольку для конечного пользователя оборудования критична поздняя доставка трафика реального времени или даже недоставка данного трафика.

Все рассмотренные недостатки передачи трафика в коммутаторах OpenFlow и ПКС приводят к возникновению новых методов процесса передачи трафика в ПКС. В [14] предложен новый метод процесса передачи трафика в ПКС – метод ранней диагностики потерь трафика реального времени с контролем таймаутов в ПКС.

На входящий порт коммутатора устанавливается аппаратный защитник по контролю таймаутов. Таким образом, для трафика реального времени непосредственно на входе выполняется контроль проверки таймаутов по протоколу OpenFlow. Выделяют следующие функции контроля: контроль при превышении таймаута (hard timeout) и принудительное удаление записи из таблицы и кадра в заданное время; контроль при превышении таймаута (idle timeout) и удаление записи из таблицы и кадра в случае недостижения передачи приемной стороны в зависимости от длины самого кадра.

С помощью данного метода становится возможным определить более раннюю диагностику потерь трафика реального времени, чем классическим методом по таблице потоков, а также осуществляется раннее удаление кадров из ПКС и информирование контроллера ПКС о повторной передаче кадра.

В статье автор предлагает другой метод процесса передачи разнородного трафика на основе планировщика и функцией контроля доставки в ПКС. Данный метод позволяет не повышать загрузку коммутатора OpenFlow за счет введения более гибкого расписания планировщика, а не жестких интервалов времени таймаутов.

Становится возможной досрочная передача эластичного трафика, пока не наступило время доставки трафика реального времени. Также присутствует раннее удаление кадра из ПКС и раннее информирование контроллера ПКС о повторной передаче кадра.

Выбор математического аппарата для формализованного описания модели. Для того чтобы охарактеризовать и оценить эффективность предложенного метода, необходимо перейти к имитационному моделированию. Оно позволяет оценить модель метода и алгоритм в ходе различных экспериментов. Формализованное описание модели можно представить с помощью логики предикатов, детерминированных и недетерминированных цифровых автоматов, семантических сетей, сетей Петри, системы массового обслуживания [15].

Для исследования компьютерных сетей и телекоммуникаций часто используют и прикладные программные продукты, такие как Cisco Packet Tracer, NetSim, OMNET++ [16]. Первый недостаток таких программ – то, что они выпускаются только фирмой-производителем сетевого оборудования, и исследователи не имеют возможности унифицировать предложенные методы процесса передачи трафика. В результате возникает привязка к конкретному сетевому оборудованию производителя.

Следующий недостаток заключается в том, что сложно описывать, дорабатывать новые узлы и блоки ввиду отсутствия новых узлов для исследования предлагаемых методов в данных программах, часто необходимо изучать узкоспециализированные языки программирования и администрирование сетевого оборудования. Поэтому в статье не рассматриваются модели в специализированных прикладных программных продуктах.

Для исследования предложенного метода был выбран математический аппарат сетей Петри. Первое преимущество сетей Петри заключается в том, что появляется возможность динамического исследования модели (количество удаленных кадров из ПКС, задержка в ПКС, загрузку коммутаторов OpenFlow).

Следующее преимущество состоит в декомпозиции модели, для того чтобы модель не была слишком громоздкой и можно было управлять ею не только на уровне одного большого узла. Не происходит разрастания большими темпами дерева достижимости состояния, пространства состояний модели и синтеза самой модели, как с помощью аппарата цифровых автоматов. Поэтому, исходя из вышеописанного, был выбран аппарат сетей Петри.

Для этих целей была разработана и описана модель процесса передачи трафика реального времени с использованием планировщика и функцией контроля доставки в ПКС на основе сетей Петри и пакета CPN Tools [17]. CPN Tools обладает всеми преимуществами аппарата сетей Петри, позволяет строить различные виды сетей Петри, наилучшим образом подходит для исследования компьютерных сетей, их критериев, метрик и задержки в сети.

Пакет обладает максимальной производительностью по исследованию новых методов и алгоритмов в компьютерных сетях. Кроме этого он имеет встроенный язык программирования CPN ML, позволяющий строить сложные элементы, типы данных (структуры, списки, кортежи) и

функции на сетях Петри, позволяет верифицировать модель на тупиковые состояния, свойство живости и анализ пространства состояний.

Модель процесса передачи разнородного трафика с использованием планировщика и функцией контроля доставки. Модель читает сгенерированный разнородный трафик, который подразделяется на трафик реального времени и эластичный трафик. Формирование и работа генератора трафика Ethernet подробно описана в [18]. Вид трафика определяется на основе поля качества обслуживания (приоритета) в самом формате кадра согласно стандарту IEEE 802.1 или протоколу OpenFlow. Трафик передается в коммутатор OpenFlow согласно временным задержкам, реализованным на переходе подсети Петри чтения трафика – @+(delay-delta).

Кадр из потока в ПКС имеет цвет в виде следующего выражения: colset frm = product INT*InPort*MAC*MAC*VLID*QoS*IP*IP*INT*INTt*INTt declare input_ms. Данный цвет frm является кортежем и состоит из набора полей согласно протоколу OpenFlow. Можно выделить следующие поля: номер кадра в потоках, входной порт, MAC-адреса источника и приемника, VLID-индекс виртуальной связи, приоритет кадра, IP-адреса источника и приемника, размер кадра, входящая и выходящая задержка.

Для передачи кадра необходимо описать, свободен ли коммутатор или занят существующей в данный момент передачей. За такое действие в модели отвечает цвет frame и представляется в виде выражения: colset frame = union f:frm + avail timed. Параметр avail будет отвечать, что коммутатор в данный момент свободен. Исходя из вышеописанного, маркер, отвечающий за передачу кадра из потока и цвета frame, будет выглядеть как выражение: 1`f(number,inport,src_MAC,dst_MAC, vlid,qos,src_IP,dst_IP, szfrm, delay,delay2).

Основным узлом коммутатора OpenFlow является его буфер. Подсеть буфера коммутатора, представленная на рис. 1, состоит из следующих узлов: подсетей таблиц потоков согласно OpenFlow; подсетей очередей для трафика реального времени и эластичного трафика; диспетчеры очередей, входного и выходного портов коммутатора.

Входящий кадр из потока передается на входной порт из подсети чтения разнородного трафика, затем поступает в подсеть таблицы потоков Flow Table 1. Предложенный метод упрощает структуру и формат таблиц потоков согласно OpenFlow, поскольку удаляются поля, связанные с работой тайм-

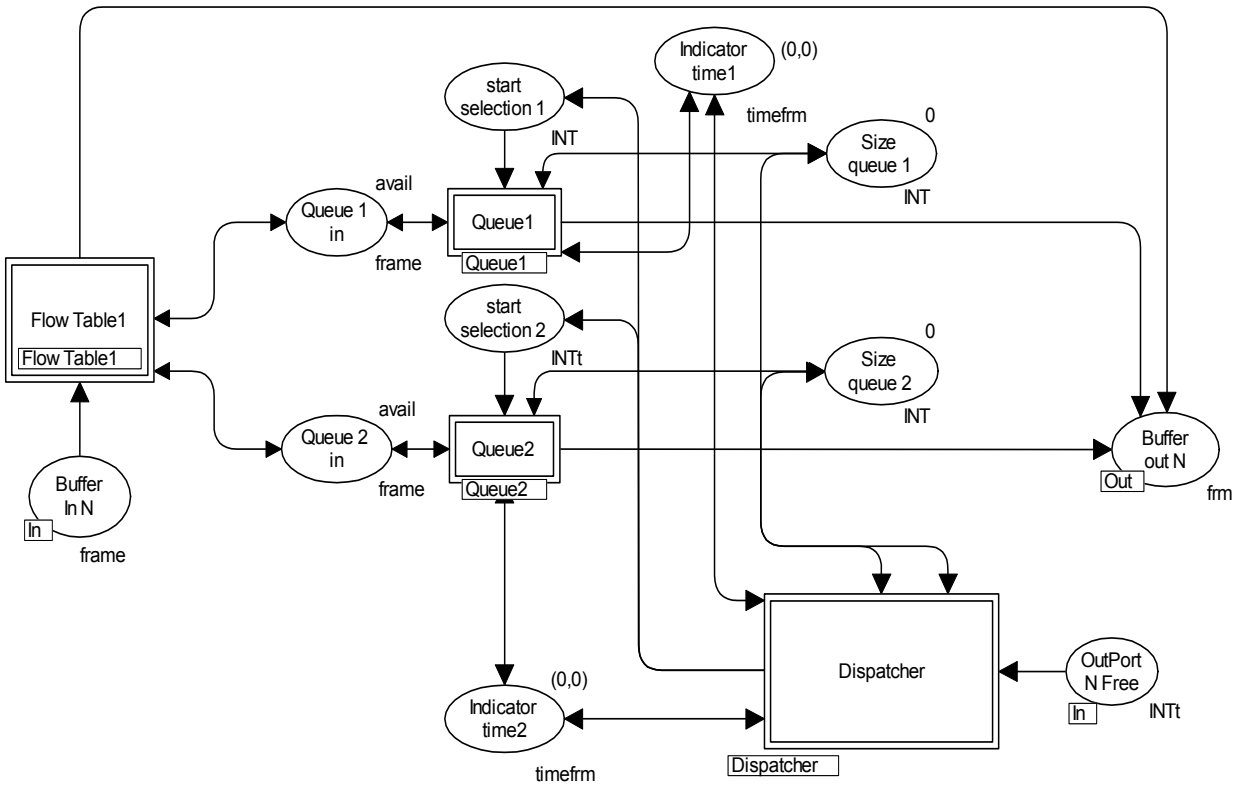

Рис. 1. Подсеть буфера коммутатора OpenFlow
Fig. 1. Subnet of buffer of OpenFlow switch

аутов. При передаче разнородного трафика контроль таймаутов не потребуется, потому что будет работать в коммутаторе планировщик трафика. Подсеть таблицы потоков представлена на рис. 2.

Подсеть таблицы потоков состоит из подсети сравнения и поиска правила в таблице потоков, входящего кадра, выходных позиций – в очередь (Queue 1-2 in) или в выходной порт коммутатора (Buffer out N). Цвет таблицы потоков flowt представляется в измененном виде: colset flowt= product INT*InPort*MAC*MAC*VLID*QoS*IP* IP*INT declare input_ms.

В данном цвете flowt отсутствуют таймауты и набор инструкций для дешифрирования коммутатором. За эти цели будет отвечать планировщик трафика. Переход Flow Table 1 отвечает за фор-

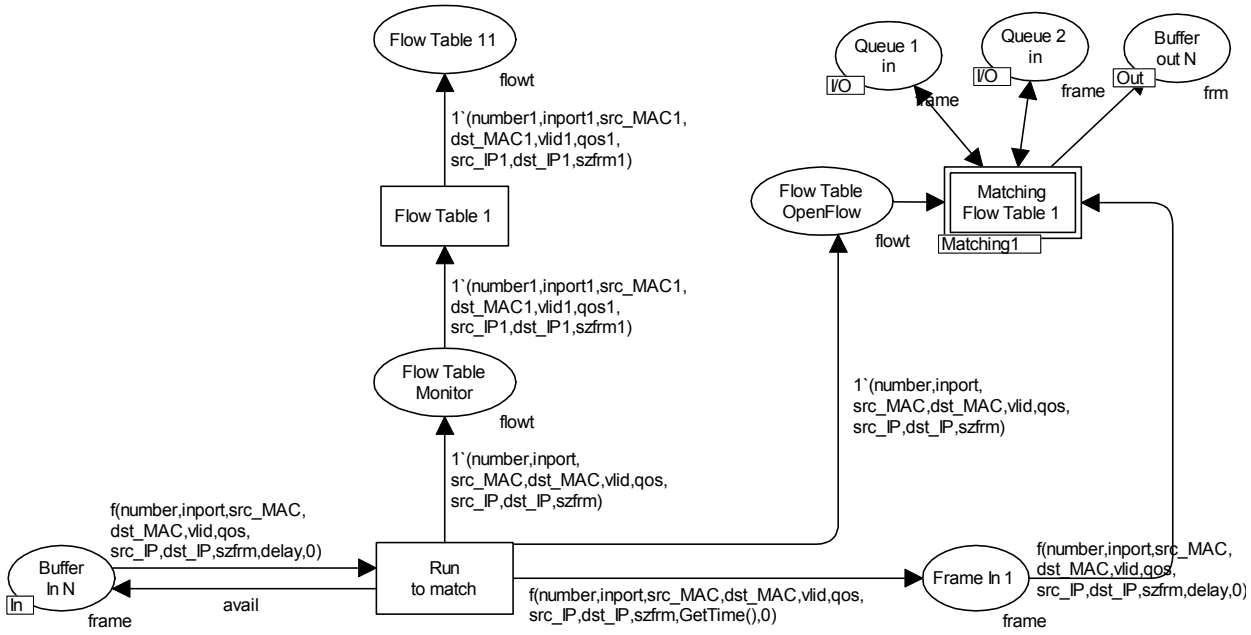

Рис. 2. Подсеть таблицы потоков 1
Fig. 2. Subnet of flow table 1

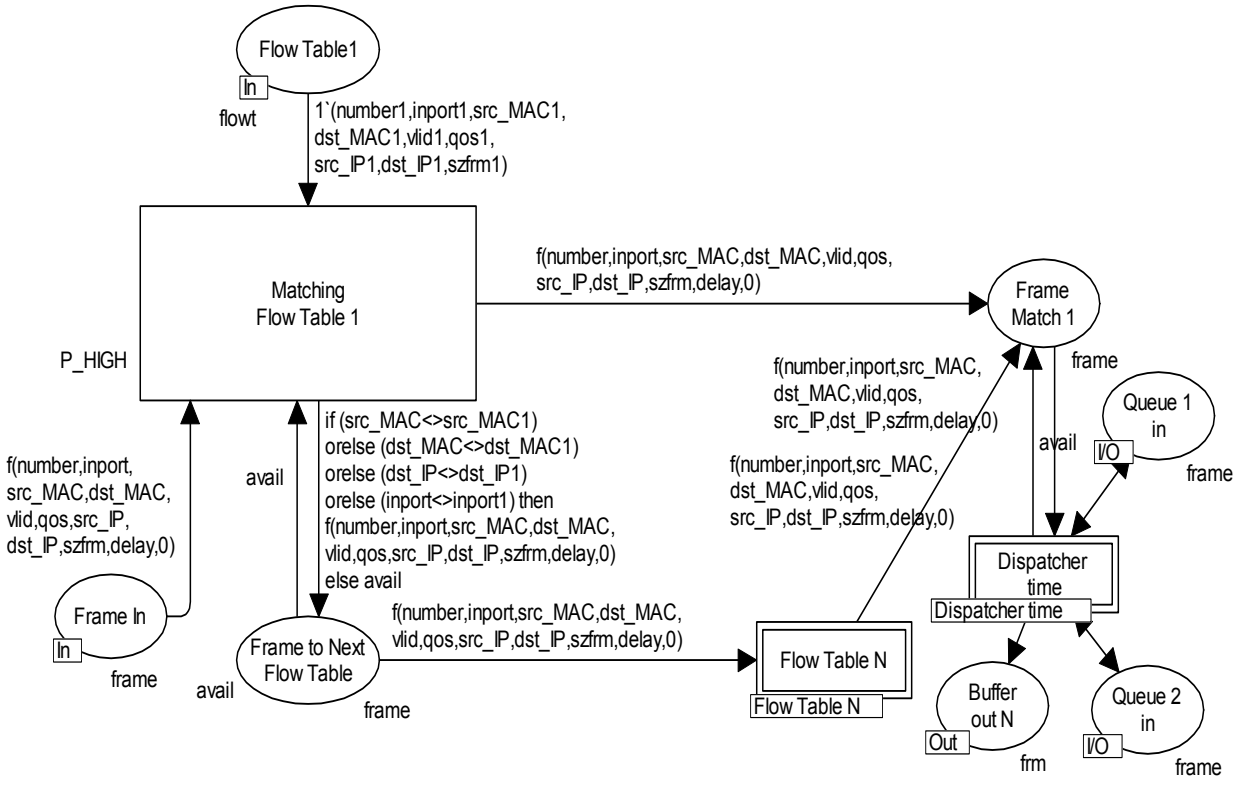

Рис. 3. Подсеть сравнения и поиска правила в таблице потоков 1
Fig. 3. Subnet of matching flow rule in flow table 1

мирование и запись в файл таблицы потоков. Этот файл может быть использован для дальнейшего анализа и исследования сгенерированных таблиц потоков коммутатора OpenFlow.

Подсеть сравнения и поиска правила в таблице потоков представлена на рис. 3. Данная подсеть состоит из подсетей других таблиц потоков, количество которых в коммутаторе может быть N, подсети планировщика трафика, входящих позиций – таблицы расписания и входящего кадра из потока, выходные позиции такие же, как на рис. 2 в подсети таблицы потоков 1.

Поиск правила осуществляется по немаскируемым полям в структуре протокола OpenFlow. К таким полям относятся входной порт, MAC-адреса источника и приемника, IP-адреса приемника. Если не найдено правило для потока в заданной таблице, то он переходит в следующую подсеть таблицы потоков N, где происходит дальнейший поиск и сравнение с правилами в следующих таблицах. Такая проверка выполняется на переходе Matching Flow Table 1, выражение имеет следующий вид: if (src_MAC<>src_MAC1) orelse (dst_MAC<>dst_MAC1) orelse (dst_IP<> dst_IP1) orelse (inport<>inport1) then f(number, inport, src_MAC,dst_MAC,vlid,qos,src_IP,dst_IP, szfrm, delay, 0) else avail. При этом если в последней таблице потоков N коммутатора не будет найдено правило, такой кадр из потока удаляется из ПКС, информируется контроллер о повторной передаче. Удаленные кадры записываются в позицию Drop Frames N.

В случае если правило найдено для кадра, такой кадр направляется в следующую подсеть планировщика трафика Dispatcher time, представленную на рис. 4. Подсеть планировщика трафика состоит из подсети классификации трафика, входного порта, позиций очередей и выходного порта коммутатора. При прибытии кадра из потока фиксируется время прибытия кадра в планировщике расписания через функцию GetTime().

Также генерируется для каждого входящего кадра режим работы планировщика трафика. Режим устанавливается через позицию Mode dispatcher, выражение имеет следующий вид: if Range.ran() <= percent then 1`1 else 1`2. Режим планировщика позволяет имитировать реальную работу коммутатора, когда кадр прибыл в строго отведенное время доставки (режим 1) или раньше (режим 2).

Сгенерированная таблица расписания планировщика хранится в позиции Table of dispatcher. Цвет у этой позиции выглядит следующим образом: colset Disp_time_lst = INTt. После этого подготовленные данные (входящий кадр и таблица расписания) направляются в подсеть классификации трафика.

В модели модифицируется подсеть классификации трафика (рис. 5) в отличие от классической модели. Подсеть классификации трафика состоит из подсети валидатора времени доставки, входящего кадра, таблицы расписания, позиций, отве-

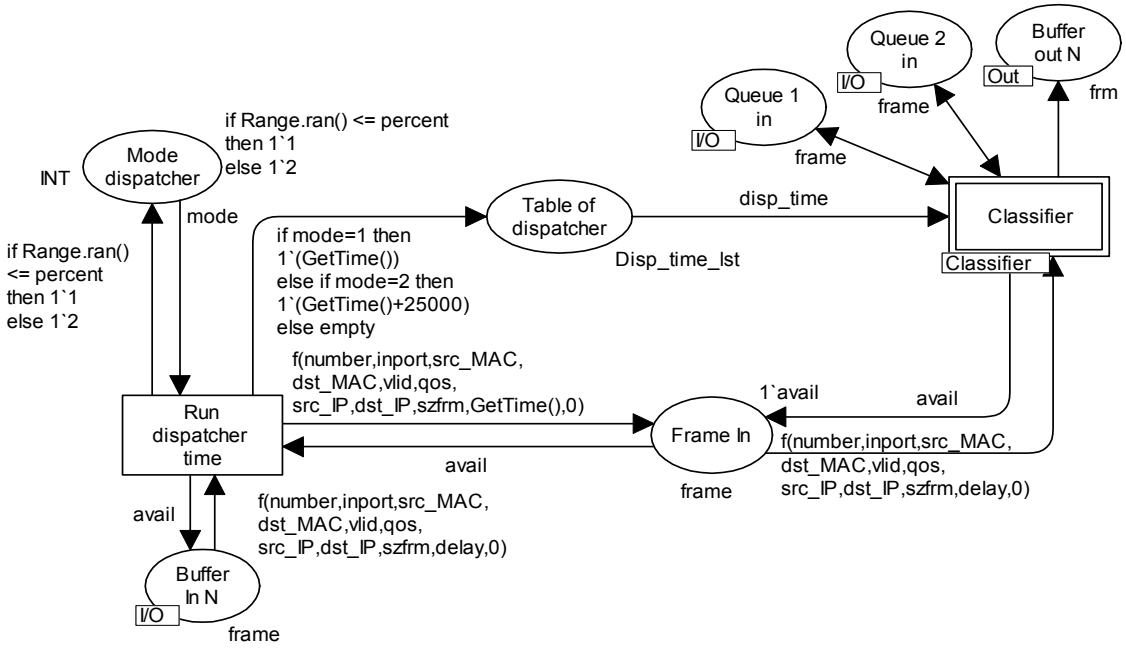

Рис. 4. Подсеть планировщика трафика
Fig. 4. Subnet of traffic scheduler

чающих за статистику передачи трафика (Count Queue, Count Port), позиций очередей и выходного порта коммутатора.

Позиция Count Port считает количество сразу же направленных кадров в выходной порт коммутатора, выражение выглядит следующим образом: if (qos = 7) andalso delay = disp_time then count1+1 else count1. Позиция Count Queue считает количество переданных кадров в очереди коммутатора, выражение выглядит следующим образом: if (qos = 7 andalso delay < disp_time) orelse qos<>7 then count2+1 else count2.

Подсеть классификации трафика совместно с подсетью валидатора времени доставки обрабатывают следующие ситуации:

– если время прибытия кадра реального времени равно времени доставки из таблицы расписания, то такой кадр передается сразу же в выходной порт коммутатора;

– если время прибытия кадра реального времени меньше времени доставки из таблицы расписания, такой кадр помещается в очередь 2 для трафика реального времени;

– если прибыл эластичный трафик, такой кадр помещается в низкоприоритетную очередь 1 коммутатора.

Подсеть валидатора времени доставки представлена на рис. 6, данная подсеть обрабатывает рассмотренные ситуации по управлению трафиком реального времени. Подсеть состоит из входящего кадра, таблицы расписания, позиции очереди 2 для трафика реального времени и выходного порта коммутатора, позиции удаленных кадров из ПКС. Основным элементом управления подсети является переход Validation time.

В позицию Queue 2 in направляется кадр реального времени, если входящая задержка из кортежа кадра будет меньше времени доставки из таблицы расписания планировщика. Выражение выглядит следующим образом: if qos = 7 andalso delay < disp_time then 1`f(number,inport,src_MAC, dst_MAC, vlid, qos,src_IP,dst_IP,szfrm,disp_time,0) else 1`avail.

В позицию Buffer out N направляется кадр реального времени, если входящая задержка из кортежа кадра будет равна времени доставки из таблицы расписания планировщика. Выражение выглядит следующим образом: if qos = 7 andalso delay = disp_time then 1`(number,inport,src_MAC, dst_MAC, vlid,qos, src_IP, dst_IP,szfrm,delay, GetTime()) else empty.

В позицию удаленных кадров Garbage collector classifier направляется кадр реального времени, если входящая задержка из кортежа кадра будет больше времени доставки из таблицы расписания планировщика. Выражение выглядит сле-

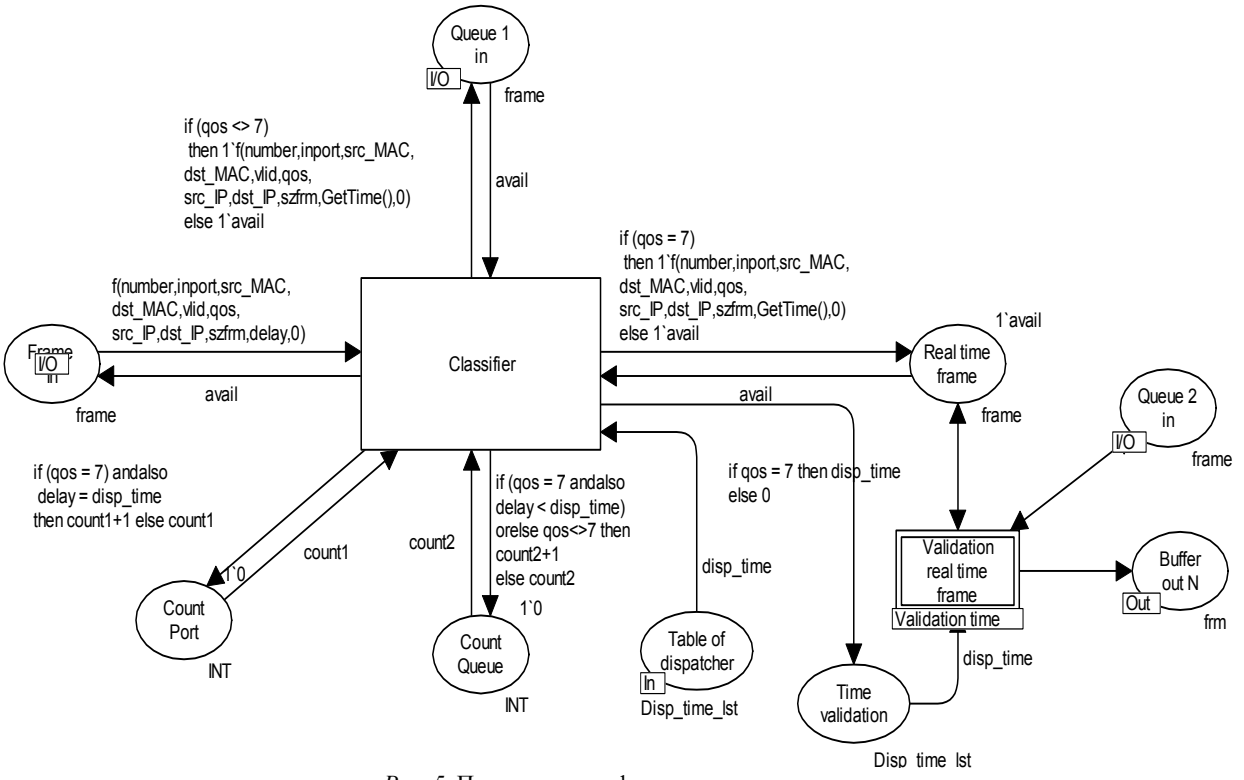

Рис. 5. Подсеть классификатора коммутатора
Fig. 5. Subnet of classifier of switch

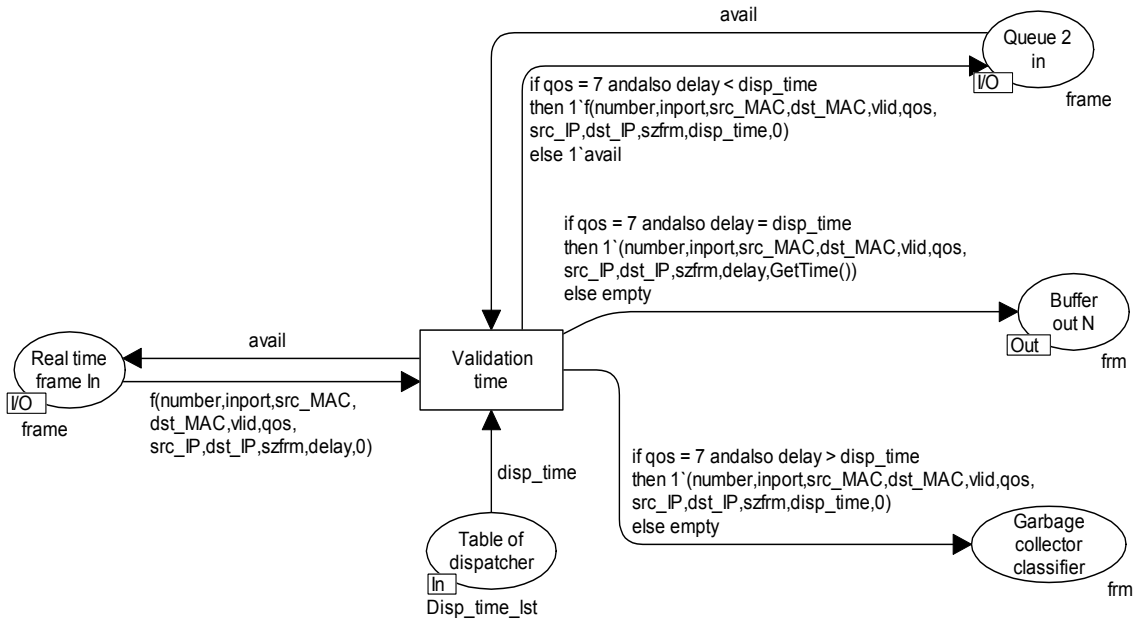

Рис. 6. Подсеть валидатора времени доставки
Fig. 6. Subnet of validation of delivery time

дующим образом: if qos = 7 andalso delay > disp_time then 1`(number,inport,src_MAC, dst_MAC, vlid, qos,src_IP,dst_IP,szfrm,disp_time,0) else empty.

Подсети очередей 1 и 2 одинаковы по своей структуре, поэтому рассмотрим пример функционирования на примере очереди 1 (рис. 7). Основным цветом в очереди является LBuffer – список, куда добавляются кадры из сети.

Отличительной особенностью очередей в данной модели является введение дополнительного буфера Buffer delay1. Данный буфер используется в диспетчере очередей коммутатора. Цвет данного буфера – это кортеж, и он состоит из следующих полей: время поступления кадра в очередь и размер входящего кадра. Через позицию Indicator1, в которую записываются данные поля, управляется диспетчер очередей.

Подсеть очереди состоит из входящего кадра в очередь, текущего размера очереди, сигнала для выборки кадра из очереди, выходящего кадра из очереди, списка дополнительных параметров входящего кадра, удаленных кадров из очереди.

Подсеть диспетчера очередей также отличается от других диспетчеров коммутатора, таких как FIFO и циклических алгоритмов [19]. Диспетчер очередей учитывает время поступления кадра и его размер. Для этого через позиции Indicator передаются эти параметры для трафика реального времени и эластичного трафика. Подсеть диспетчера очередей представлена на рис. 8.

Диспетчер очередей запускается при наличии хотя бы одной непустой очереди и сигнала свободного выходного порта. Сигнал для передачи кадра из очереди эластичного трафика вырабатывается, если очередь эластичного трафика не пуста и очередь трафика реального времени пуста, или не пусты обе очереди и время поступления кадра эластичного трафика и его размер позволяют его передать до наступления времени доставки кадра реального времени. Диспетчер запускает очередь 1 по следующему выражению: if (sz1>0 andalso sz2=0) orelse (sz2>0 andalso sz1>0 andalso (ind1+szf1<ind2)) then 1`1 else empty.

Сигнал для передачи кадра из очереди трафика реального времени вырабатывается в случае: если не пуста эта очередь и пуста очередь эластичного трафика, или не пусты обе очереди, но кадр эластичного трафика не может передаться, так как его время доставки больше времени доставки кадра реального времени, поэтому будет передаваться кадр реального времени.

Сигнал вырабатывается четко в строго отведенное время доставки кадра, что достигается временной задержкой ind2-GetTime(). Сигнал для разрешения из очереди трафика реального времени представляется в виде следующего выражения: (if (sz2>0 andalso sz1=0) orelse (sz2>0 andalso sz1>0 andalso (ind1+szf1>ind2)) then 1`1 else empty)@+ (ind2-GetTime()). Подсеть выходного порта аналогична подсети в модели классического метода.

Результаты моделирования. Было проведено экспериментальное исследование моделей классического и предложенного методов процесса передачи трафика в коммутаторе OpenFlow с разной загрузкой трафика: 3436, 5300 и 7500 кадров. При

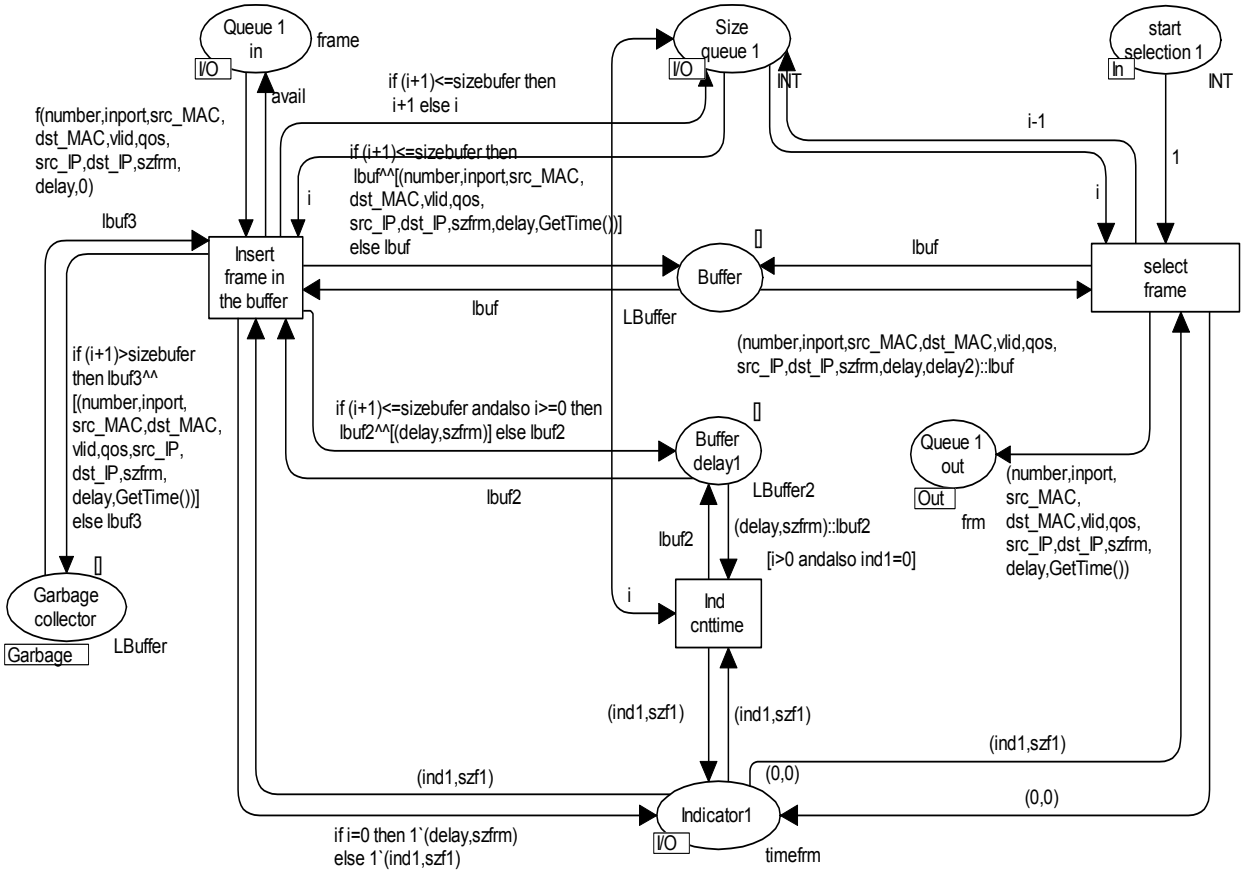

Рис. 7. Подсеть очереди
Fig. 7. Subnet of queue

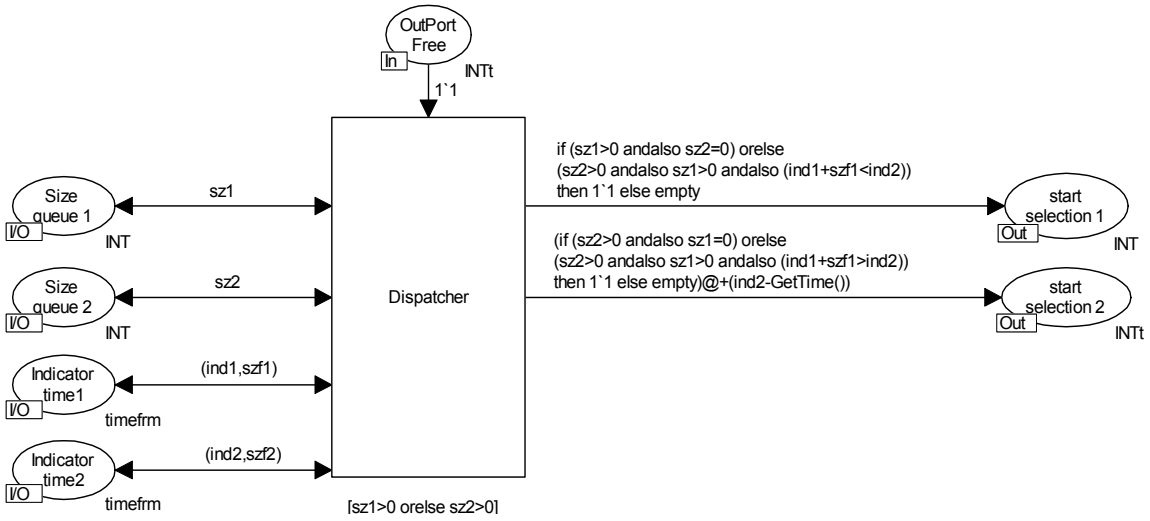

Рис. 8. Подсеть диспетчера очередей
Fig. 8. Subnet of queue dispatcher

анализе результатов экспериментов было выявлено увеличение загрузки коммутатора из-за удаления кадров в модели классическим методом [20].

Потери возникают со стороны жестких временных рамок контроля таймаутов по протоколу OpenFlow в случае, когда время прибытия потоков больше отведенного времени в блоках контроля таймаутов. Загрузка коммутатора определяется отношением общей суммы всех переданных бит в коммутаторе к итоговому времени передачи всех кадров, поэтому загрузка отображается в процентах (рабочая загрузка 80 %) или числовым коэффициентом (рабочая загрузка 0.8). При загрузке коммутатора, равной 0.8, появляется критическая доля удаленных кадров, тем самым загрузка увеличивается до 0.88 и приводит к неустойчивости работы коммутатора.

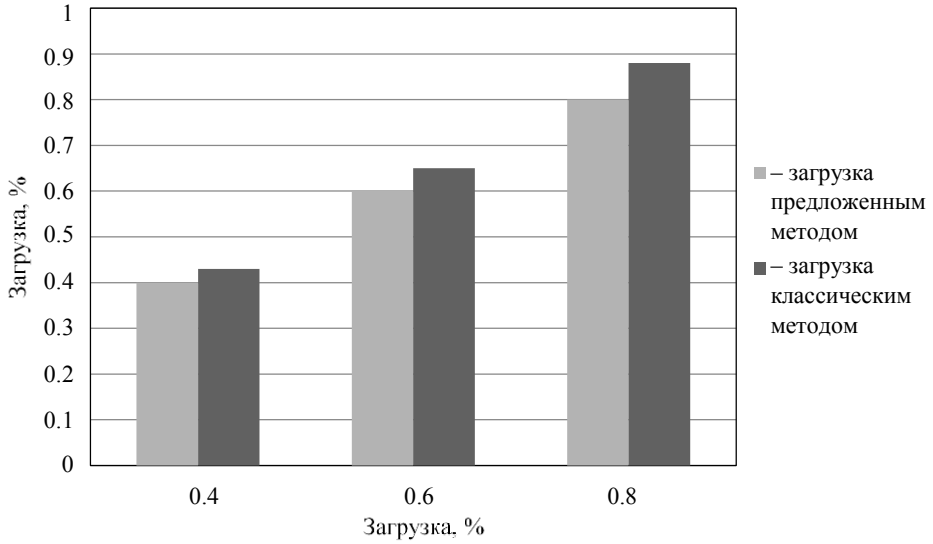

Рис. 9. Загрузка коммутатора
Fig. 9. Load of switch

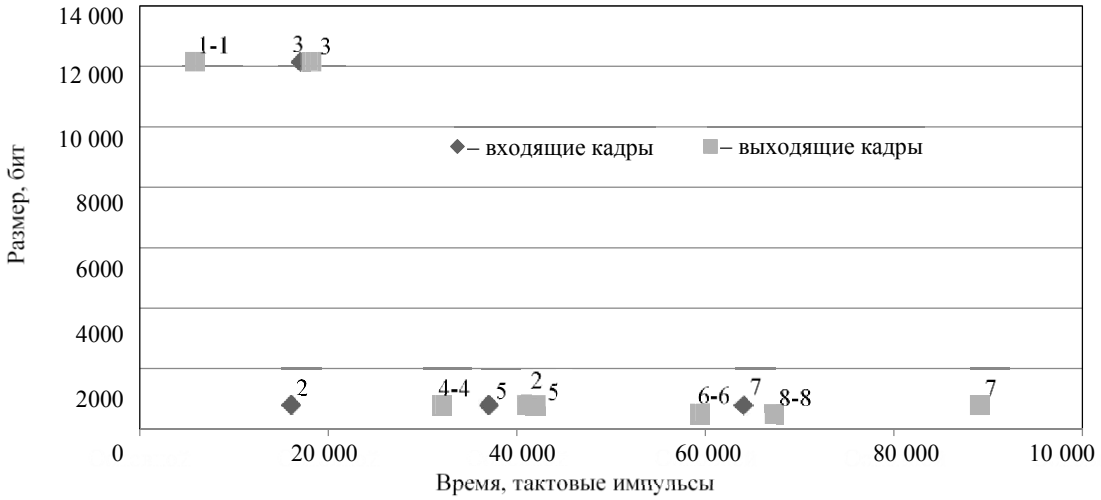

Рис. 10. Передача кадров в ПКС
Fig. 10. Transmission frames in SDN

В классическом методе загрузка коммутатора увеличивается на 11 %, что хуже загрузки предложенным методом. При увеличении еще большего количества кадров (потоков) в ПКС могут возникнуть перебои в работе коммутатора и достичь пиковой отметки загрузки коммутатора на уровне 0.9–0.95. На рис. 9 представлена гистограмма сравнения загрузки коммутатора OpenFlow двумя методами процесса передачи разнородного трафика.

Даже при передаче небольшого количества кадров модель предложенного метода подтвердила свою эффективность еще и за счет возможности досрочной передачи эластичного трафика, пока не наступило время доставки трафика реального времени. На рис. 10 показано поступление кадров в ПКС, их моменты поступления на входной порт и ухода с выходного порта. Если момент прибытия кадра равен моменту убытия, такой кадр помечается через дефис. Время на рисунке указывается в тактовых импульсах среды CPN Tools.

Особый интерес представляет передача кадра реального времени 2. Данный кадр поступил в ПКС раньше обычного кадра 3. В свою очередь запустился планировщик трафика вместе с валидатором времени доставки, выявив, что длина кадра 3 позволяет ему пройти до момента доставки кадра 2, что и произошло с кадром 3 на рис. 10. Такая же ситуация возникает с кадрами 7–8.

На рис. 11 показано количество ранней передачи кадров стандартного трафика, наибольшая эффективность предложенного метода достигается при рабочей загрузке коммутатора, равной 0.8. Таким образом, на 17 % увеличивается возмож-

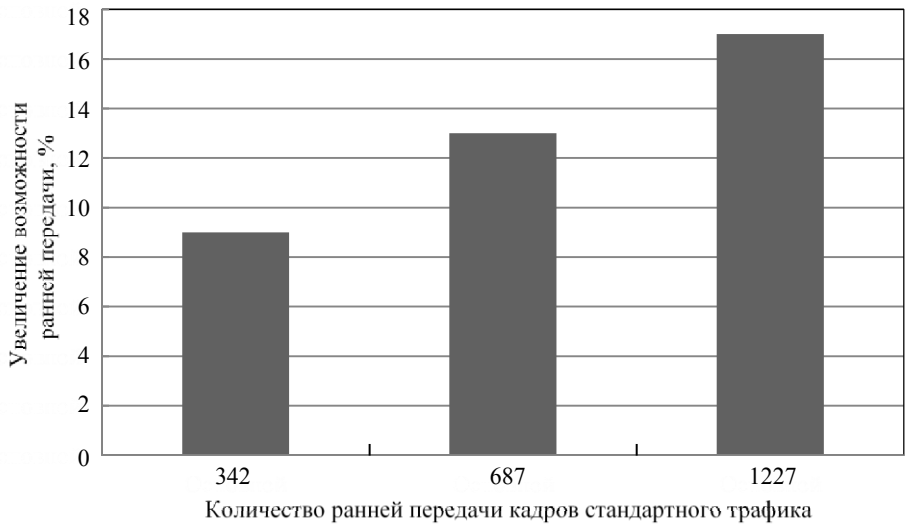

Рис. 11. Досрочная передача кадров стандартного трафика в ПКС
Fig. 11. Early transmission frames of standard traffic in SDN

ность досрочной передачи эластичного трафика, чем классическим методом. Удаленных кадров в ПКС также не выявлено, что сказывается на конечной загрузке коммутатора (рис. 9).

Заключение. Была разработана и исследована модель процесса передачи трафика реального времени с использованием планировщика и функцией контроля доставки в ПКС на сетях Петри и пакета CPN Tools.

Были разработаны и описаны подсети Петри планировщика трафика, валидатора времени доставки, произведена модификация подсетей таблиц потоков, подсетей сравнения и поиска правила в таблице потоков, классификации трафика, очередей и диспетчера очередей.

Было проведено экспериментальное моделирование с различной загрузкой коммутатора, приведено сравнение результатов разработанного метода с классическим методом передачи разнородного трафика в ПКС.

Согласно полученным данным метод процесса передачи трафика реального времени с использованием планировщика и функцией контроля доставки в ПКС доказал свою эффективность. Эффективность предложенного метода заключается в том, что остается постоянной загрузка коммутатора за счет более гибкой настройки расписания планировщика, а не жесткими интервалами времени таймаутов, а также осуществляется возможность досрочной передачи эластичного трафика, пока не наступило время доставки трафика 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. Кучерявый Е. А. Управление трафиком и качество обслуживания в сети Интернет. СПб.: Наука и Техника, 2004. 336 с.

3. Scheduling queues in the Ethernet switch, considering the waiting time of frames / E. Kizilov, N. Konnov, K. Nikishin, D. Pashchenko, D. Trokoz // MATEC Web of Conf. 2016. Vol. 44. P. 1–5.

4. Nikishin K., Konnov N. Schedule time-triggered Ethernet // Intern. Conf. on Engin. Management of Communication and Technol., EMCTECH 2020. Vienna: IEEE, 2021. P. 6–15. doi: 10.1109/EMCTECH49634.2020.9261540.

5. Openflow: enabling innovation in campus networks / N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar // ACM SIGCOMM Comp. Communication Rev. 2008. Vol. 38, no. 2. P. 69–74.

6. Advanced study of SDN/OpenFlow controllers / A. Shalimov, D. Zuikov, D. Zimarina, V. Pashkov, R. Smeliansky // Proc. of the 9th Central & Eastern European Software Engin. Conf. in Russia. ACM. M. 2013. P. 1–6. doi: 10.1145/2556610.2556621.

7. Maturing of OpenFlow and Software-Defined networking through deployments / M. Kobayashi, S. Seetharaman, G. Parulkar, G. Appenzeller, J. Little, J. Van Reijendam, N. McKeown // Computer Networks. 2014. Vol. 61. P. 151–175.

8. Karakus M., Durresi A. Quality of service (QoS) in software defined networking (SDN): A survey // J. of Network and Computer Appl. 2017. Vol. 80. P. 200–218.

9. 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.

10. Maniu R., Dumitru L. A. Exploring the possibilities of a self-regulating SDN controller // Scientific Bull. «Mircea cel Batran» Naval Academy. 2015. Vol. 18, no. 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. P. 514–518. doi: 10.1109/CyberC.2016.103.

13. Formal modeling and Verification of SDN-OpenFlow / M. Kang, E.-Y. Kang, D. Hwang, B. Kim, K.-H. Nam, M.-K. Shin, J.-Y. Choi // IEEE Sixth Intern. Conf. Software Testing, Verification and Validation (ICST). Luxembourg, 2013. P. 481–482. doi: 10.1109/ICST.2013.69.

14. Никишин К. И. Моделирование метода ранней диагностики потерь трафика реального времени в программно-конфигурируемых сетях на основе аппарата сетей Петри // Вестн. Поволжского гос. техн. ун-та. Сер.: Радиотехнические и инфокоммуникационные системы. 2022. № 2 (54). С. 47–60. doi: 10.25686/2306-2819.2022.2.4.

15. Алиев Т. И. Погрешности моделирования высоконагруженных систем в GPSS World // Науч.-техн. вестн. информационных технологий, механики и оптики. 2013, № 1 (83). С. 70–75.

16. An extension of the OMNeT++ INET framework for simulating real-time Ethernet with high accuracy / T. Steinbach, H. Dieumo Kenfack, F. Korf, T. Schmidt // Proc. of the 4th Intern. ICST Conf. on Simulation Tools and Techniques. Barcelona, Spain: 2011. P. 375–382.

17. Jensen K., Kristensen L. M. Coloured Petri nets. Modellin 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. 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.

20. Никишин К. И. Моделирование процесса передачи трафика в программно-конфигурируемых сетях // Вестн. Рязанского гос. радиотехнического ун-та. 2022. № 81. С. 32–41. doi: 10.21667/1995-4565-2022-81-32-41.

Информация об авторе

Никишин Кирилл Игоревич – канд. техн. наук, доцент. Пензенский государственный университет, ул. Красная, д. 40, Пенза, 440026, Россия.
E-mail: nkipnz@mail.ru
http://orcid.org/0000-0001-7966-7833

References

1. Olifer V. G., Olifer N. A. Komp'juternye seti. Principy, tehnologii, protokoly, 4-e izd. SPb.: Piter, 2010. 943 s. (In Russ.).

2. Kucherjavyj E. A. Upravlenie trafikom i kachestvo obsluzhivanija v seti Internet. SPb.: Nauka i Tehnika, 2004. 336 s. (In Russ.).

3. Scheduling queues in the Ethernet switch, considering the waiting time of frames / E. Kizilov, N. Konnov, K. Nikishin, D. Pashchenko, D. Trokoz // MATEC Web of Conf. 2016. Vol. 44. P. 1–5.

4. Nikishin K., Konnov N. Schedule time-triggered Ethernet // Intern. Conf. on Engin. Management of Communication and Technol., EMCTECH 2020. Vienna: IEEE, 2021. P. 6–15. doi: 10.1109/EMCTECH49634.2020.9261540.

5. Openflow: enabling innovation in campus networks / N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar // ACM SIGCOMM Comp. Communication Rev. 2008. Vol. 38, no. 2. P. 69–74.

6. Advanced study of SDN/OpenFlow controllers / A. Shalimov, D. Zuikov, D. Zimarina, V. Pashkov, R. Smeliansky // Proc. of the 9th Central & Eastern European Software Engin. Conf. in Russia. ACM. M. 2013. P. 1–6. doi: 10.1145/2556610.2556621.

7. Maturing of OpenFlow and Software-Defined networking through deployments / M. Kobayashi, S. Seetharaman, G. Parulkar, G. Appenzeller, J. Little, J. Van Reijendam, N. McKeown // Computer Networks. 2014. Vol. 61. P. 151–175.

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

9. Perepelkin D. A., Byshov V. S. Balansirovka potokov dannyh v programmno-konfiguriruemyh setjah s obespecheniem kachestva obsluzhivanija setevyh servisov // Radiotehnika. 2016, № 11. S. 111–119. (In Russ.).

10. Maniu R. Dumitru L. A. Exploring the possibilities of a self-regulating SDN controller // Scientific Bull. «Mircea cel Batran» Naval Academy. 2015. Vol. 18, no. 1. P. 58–61.

11. Perepelkin D. A. Konceptual'nyj podhod dinamicheskogo formirovanija trafika programmno-konfiguriruemyh telekommunikacionnyh setej s balansirovkoj nagruzki // Informacionnye tehnologii. 2015. T. 21, № 8. S. 602–610. (In Russ.).

12. Ren H., Li X., Geng J. A SDN-based dynamic traffic scheduling algorithm // IEEE Intern. Conf. on Cyber-Enabled Distributed Computing and Knowledge Discov-

ery (CyberC). Chengdu. China. 2016. P. 514–518. doi: 10.1109/CyberC.2016.103.

13. Formal modeling and Verification of SDN-OpenFlow / M. Kang, E.-Y. Kang, D. Hwang, B. Kim, K.-H. Nam, M.-K. Shin, J.-Y. Choi // IEEE Sixth Intern. Conf. Software Testing, Verification and Validation (ICST). Luxembourg, 2013. P. 481–482. doi: 10.1109/ICST.2013.69.

14. Nikishin K. I. Modelirovanie metoda rannej diagnostiki poter' trafika real'nogo vremeni v programmno-konfiguriruemyh setjah na osnove apparata setej Petri // Vestn. Povolzhskogo gos. tehn. un-ta. Ser.: Radiotehnicheskie i infokommunikacionnye sistemy. 2022. № 2 (54). S. 47–60. doi: 10.25686/2306-2819.2022.2.4. (In Russ.).

15. Aliev T. I. Pogreshnosti modelirovanija vysokonagruzhennyh sistem v GPSS World // Nauch.-tehn. vestn. informacionnyh tehnologij, mehaniki i optiki. 2013, № 1 (83). S. 70–75. (In Russ.).

16. An extension of the OMNeT++ INET framework for simulating real-time Ethernet with high accuracy / T. Steinbach, H. Dieumo Kenfack, F. Korf, T. Schmidt // Proc. of the 4th Intern. ICST Conf. on Simulation Tools and Techniques. Barcelona, Spain: 2011. P. 375–382.

17. Jensen K., Kristensen L. M. Coloured Petri nets. Modellin and validation of concurrent systems. Berlin: Springer, 2009. 384 p.

18. Nikishin K. I, Konnov N. N. Generator trafika Ethernet na osnove cvetnyh setej Petri // Modeli, sistemy, seti v jekonomike, tehnike, prirode i obshhestve. 2016. № 1 (17). S. 299–307. (In Russ.).

19. Mehanov V. B., Kizilov, E. A. Modelirovanie cvetnymi setjami Petri obsluzhivanija ocheredej algoritmom WRR // Tr. IX Mezhdunar. nauch.-tehn. konf. «Novye informacionnye tehnologii i sistemy», Ch. 1. Penza: Izd-vo PGU, 2010. S. 67–73. (In Russ.).

20. Nikishin K. I. Modelirovanie processa peredachi trafika v programmno-konfiguriruemyh setjah // Vestn. Rjazanskogo gos. radiotehnicheskogo un-ta. 2022. № 81. S. 32–41. doi: 10.21667/1995-4565-2022-81-32-41. (In Russ.).

Information about the author

Kirill I. Nikishin – Cand. Sci. (Eng.), Associate professor of Penza State University, Krasnaya str., 40, Penza, 440026, Russia.
E-mail: nkipnz@mail.ru
http://orcid.org/0000-0001-7966-7833

Статья поступила в редакцию 18.10.2022; принята к публикации после рецензирования 24.11.2022; опубликована онлайн 30.01.2023.

Submitted 18.10.2022; accepted 24.11.2022; published online 30.01.2023.