Ордена Трудового Красного Знамени
федеральное государственное бюджетное образовательное учреждение
высшего образования

«Московский технический университет связи и информатики»

(МТУСИ)

На правах рукописи

Мусатов Владислав Константинович

РАЗРАБОТКА МЕТОДА ОЦЕНКИ ПОКАЗАТЕЛЕЙ ПРОИЗВОДИТЕЛЬНОСТИ МЕЖСЕТЕВЫХ ЭКРАНОВ ПРИ ФУНКЦИОНИРОВАНИИ В УСЛОВИЯХ ПРИОРИТИЗАЦИИ ТРАФИКА

Специальность 05.12.13 - Системы, сети и устройства телекоммуникаций

Диссертация на соискание ученой степени

кандидата технических наук

Научный руководитель

к.т.н., профессор Пшеничников А.П.

Москва – 2018

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность темы исследования. Рост числа мобильных клиентских устройств, подключенных к сети связи, желание пользователей постоянно быть на связи, расширение номенклатуры онлайн-сервисов, распространение социальных сетей и видеоконференцсвязи способствуют распространению технологии широкополосного доступа и повышению требований к качеству передачи данных.

Согласно прогнозам ежегодного исследования Cisco Visual Networking Index, с 2016 г. по 2021 г. ожидается рост числа персональных сетевых устройств на человека с 2,3 до 3,5; доля людей, подключенных к глобальной сети, увеличится с 44% до 58% от общего населения планеты, а средняя скорость доступа мобильных устройств вырастет с 6,8 до 20 Мбит/с [1].

В ответ на подобные вызовы международное сообщество в лице *Международного союза электросвязи* (МСЭ) в 2015 г. сформировало оперативную исследовательскую группу IMT-2020 [2], которая способствует развитию технологии сетей 5G [3, 4] и концепции *Будущих сетей* (англ. Future networks) [5-8]. Эта группа в своём отчёте [3] и рекомендации ITU-R M.2083-0 [4] определила требования к разрабатываемой технологии сетей 5G, например, задержку передачи данных «из конца в конец» («end-to-end») в рамках одного сегмента сети необходимо снизить с 10-ти до 1 мс. Это, в свою очередь, требует повышения производительности сетевого оборудования.

Крайне актуальной задачей является обеспечение информационной безопасности. Один из ее подразделов – сетевая безопасность, является важной задачей, решаемой при создании и эксплуатации информационных систем и сетей связи. В концепции сетевой безопасности базовым и наиболее распространённым элементом является *межсетевой экран* (англ. Firewall).

Межсетевым экраном называется программное и программно-техническое средство, реализующее функции контроля и фильтрации информационных

потоков в соответствии с заданными правилами (не криптографическими методами) [9].

Если рассматривать межсетевой экран как узел сети передачи данных, то легко обнаружить, что в большинстве случаев он вызывает бóльшую задержку обслуживания пакетов, чем обычное сетевое оборудование (маршрутизаторы, коммутаторы). Это связано с наличием широкого спектра функций обеспечения безопасности, на выполнение которых межсетевому экрану требуется время. Подобные устройства могут стать «узкими местами» в Будущих сетях и сетях 5G.

Не весь сетевой трафик является критичным к задержкам и/или другим показателям *качества обслуживания* (англ. Quality of Service, QoS). Например, к трафику, критичному к отдельным показателям качества обслуживания, относят сигнальный трафик, трафик видеоконференцсвязи реального времени, медицинский трафик и др. В оборудовании современных сетей связи предусмотрены различные механизмы QoS. Один из механизмов предусматривает формирование приоритетов обслуживания одних классов трафика над другими и называется *механизмом управления перегрузками* (англ. congestion management) [10]. Для упрощения будем называть его *механизмом приоритизации обслуживания*.

В Будущих сетях требования к задержкам так высоки, что механизмы приоритизации обслуживания трафика начнут применяться не только внутри операторской сети связи, но и в *локальных вычислительных сетях* (ЛВС). Отдельные типы современных межсетевых экранов, используемых на границе ЛВС с операторской сетью, не поддерживает механизмы приоритизации обслуживания. Из-за особенностей обслуживания пакетов в межсетевых экранах применение приоритизации обслуживания в них актуальней во входной очереди, а не в выходной, что не распространено в современных межсетевых экранах корпоративного сегмента.

Всё вышесказанное обуславливает актуальность данного направления исследований, выражающегося в проведении оценки влияния механизмов

приоритизации на показатели качества обслуживания трафика в межсетевых экранах корпоративного сегмента (SOHO, Branch, Enterprise).

Степень разработанности темы. Вопросам истории развития, моделирования функционирования сетевого оборудования и фильтрации трафика в предметной области посвящены работы зарубежных и отечественных авторов: Acharya S. [11], Al-Shaer E. [12, 13], Gusev M. [14], Kaur H. [15], Liu A. X. [16-18], Salah K. [19-21], Барабанова В. Ф.[22], Богораза А. Г. [23], а в теоретической области работы: Гайдамаки Ю. В. [24], Коломойцева В. С. [25], Леваков А. К. [26, 27], Назарова А. Н. [28, 29], Самуйлова К. Е. [30, 31], Шабурова А. С. [32], Шелухина О. И. [33] и других.

Вопросы истории развития средств межсетевого экранирования рассмотрены в работах Ingham K. и Schimmel J. [34, 35].

Вопросам формирования концепции Будущих сетей и сетей 5G посвящены работы регуляторов в лице IMT-2020. Качественный обзор работ по концепции Будущих сетей выполнен Росляковым А. В. [5-8].

Объектом исследования является межсетевой экран, функционирующий в условиях приоритизации обслуживания трафика. Объект исследования приведён на рисунке 1.

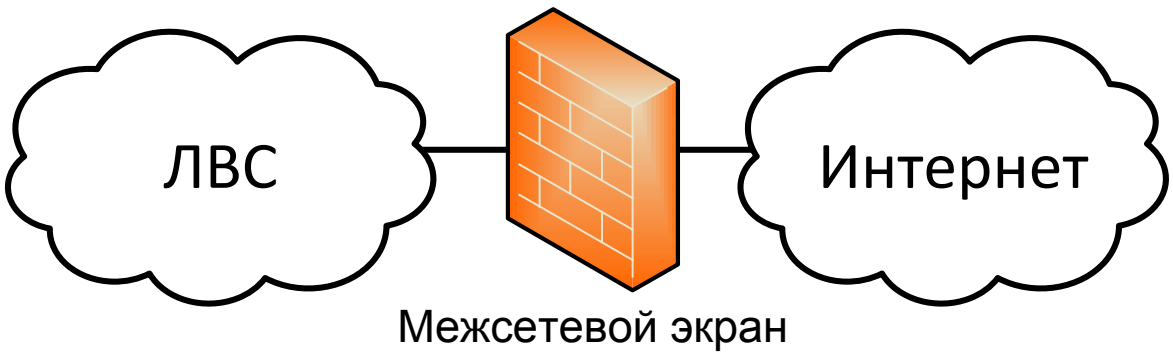


Рисунок 1 – Объект исследования

Предметом исследования являются показатели производительности межсетевого экрана, функционирующего в условиях приоритизации обслуживания трафика.

Цель и задачи исследования. Целью диссертационной работы является разработка метода оценки показателей производительности межсетевых экранов,

функционирующих в условиях приоритизации обслуживания трафика и исследование влияния приоритизации трафика на входном интерфейсе межсетевого экрана на его производительность.

Для достижения цели в диссертационной работе решены следующие задачи.

1. Анализ принципов реализации механизмов QoS в межсетевых экранах. Сформулирована постановка задачи для разработки модели функционирования межсетевого экрана в условиях приоритизации обслуживания трафика на его входе.

2. Разработка математической и имитационной моделей функционирования межсетевого экрана в условиях приоритизации обслуживания трафика на его входе.

3. Сравнительный анализ результатов математического и имитационного моделирования при снятии в имитационной модели ряда допущений, принятых в математической модели.

4. Исследование влияния числа проверок по базе правил фильтрации и объема пакетной очереди на показатели качества обслуживания пакетов межсетевым экраном. Сравнительный анализ показателей производительности межсетевого экрана, полученных с включенными и выключенными механизмами приоритизации обслуживания трафика на его входе.

В рамках работы исследованы следующие показатели производительности межсетевых экранов отдельно для приоритетного и неприоритетного потоков трафика:

1. Обслуженный поток пакетов (сквозная пропускная способность);

2. Средняя длина входной очереди межсетевого экрана;

3. Потеря пакетов;

4. Среднее время пребывание пакета в межсетевом экране (вносимая задержка);

5. Среднее время нахождение пакета в очереди;

6. Среднее время обслуживания пакета базой правил фильтрации;

7. Количество пришедших в систему пакетов и их приоритет;

8. Количество покинувших систему пакетов по тем или иным причинам и их приоритет.

Научная новизна результатов диссертационной работы.

1. Разработан метод оценки показателей производительности межсетевого экрана, включающий в себя математическую и имитационную модели, который в отличие от существующих методов позволяет учитывать обслуживание на входах межсетевых экранов двух классов трафика – приоритетного и неприоритетного.

2. Для решения системы уравнений равновесия трёхмерной марковской модели функционирования межсетевого экрана разработана процедура перевода трехмерной матрицы переходных вероятностей в двумерную матрицу, позволяющая применить эффективный вычислительный метод расчёта стационарных вероятностей, основанный на применении блочных треугольных разложений.

3. Выявлены зависимости показателей качества обслуживания трафика межсетевыми экранами, функционирующими в условиях приоритизации обслуживания критичного к задержкам трафика, от длины очереди, длительности обслуживания и правил фильтрации при функционировании в режиме без перегрузки и с перегрузкой. Учёт данных зависимостей при эксплуатации межсетевых экранов позволяет снизить задержки проходящего сквозь них трафика и избежать дополнительных перегрузок.

Теоретическую значимость работы. Разработан метод оценки показателей производительности межсетевых экранов при их функционировании в условиях приоритизации обслуживания чувствительного к задержкам трафика в их входных очередях. В основу метода положены математическая и имитационная модели. Разработанный метод позволяет рассчитать показатели

производительности межсетевого экрана в различных условиях его функционирования.

Практической ценностью работы. Разработанный метод оценки показателей производительности межсетевых экранов позволяет производителям оборудования оценить необходимость предусмотреть возможность использования механизмов приоритизации обслуживания в ряд существующих межсетевых экранов Enterprise-класса.

Выработанные рекомендации по управлению доступным объёмом входной очереди межсетевых экранов и механизмами приоритизации обслуживания трафика позволяют эксплуатационному персоналу улучшить показатели качества обслуживания проходящего сквозь межсетевой экран трафика. Рекомендации представлены в заключение к диссертации.

Имитационная модель с интуитивным и понятным графическим интерфейсом и инструкцией пользователя позволяет быстро получить искомые показатели производительности межсетевых экранов. Исходные коды имитационной модели на языке C# можно получить при обращении по email: vladmus89@gmail.com.

Результаты диссертации использованы в ЗАО «Научно-производственное предприятие «Безопасные информационные технологии», а также в учебном процессе на кафедре Сетей связи и систем коммутации МТУСИ, что подтверждено соответствующими актами (Приложение Ж).

Методология и методы исследования. Для решения поставленных в диссертационной работе задач использованы методы теории массового обслуживания, теории марковских случайных процессов, теории сетей связи, а также методы имитационного моделирования и программирования на высокоуровневом языке общего назначения C#.

Степень достоверности полученных результатов обеспечивается строгим математическим обоснованием утверждений и подкрепляется их согласованностью с данными имитационного моделирования.

Апробация работы. Основные результаты работы докладывались и обсуждались на следующих 9 конференциях:

- Международный форум информатизации (МФИ-2011). Москва, 2011;
- Международный форум информатизации (МФИ-2012). Москва, 2012;
- Международная научно-техническая конференция «Фундаментальные проблемы радиоэлектронного приборостроения» (INTERMATIC – 2012). Москва, 2012;
- 7-ая Отраслевая научная конференция «Технологии информационного общества». Москва, 2013;
- Международная конференция «Радиоэлектронные устройства и системы для инфокоммуникационных технологий» (RES-2013). Москва, 2013;
- Международный форум информатизации (МФИ-2013). Москва, 2013;
- 8-ая Отраслевая научная конференция «Технологии информационного общества». Москва, 2014;
- 9-ая Отраслевая научная конференция «Технологии информационного общества». Москва, 2015;
- 10-ая Отраслевая научная конференция «Технологии информационного общества». Москва, 2016.

Основные положения, выносимые на защиту.

1. В существующих межсетевых экранах корпоративного класса, как правило, не предусмотрена возможность использования механизмов приоритизации обслуживания трафика во входных очередях. При переполнении входных очередей межсетевых экранов для обеспечения заданного уровня обслуживания критичного к задержкам трафика необходимо применение механизмов приоритизации обслуживания во входных очередях.
2. Функционирование межсетевого экрана в условиях приоритизации трафика может быть представлено в виде однолинейной системы массового

обслуживания типа $\overrightarrow{M} / M / 1 / L / f_1^{22}$ с ограниченным накопителем пакетов, комбинированными потерями и смешанными приоритетами обслуживания.

3. При функционировании межсетевого экрана с приоритизацией обслуживания трафика в режиме перегрузки при времени обслуживания, распределённом по экспоненциальному закону, среднее время нахождения неприоритетных пакетов в очереди становится меньше, чем при постоянном времени обслуживания, что обуславливается повышенной вероятностью частичного освобождения очереди.

4. При функционировании межсетевого экрана с приоритизацией обслуживания трафика в режиме без перегрузок увеличение максимального объёма очереди приводит к сглаживанию флуктуаций входящего трафика, что снижает потери. Однако при этом в режиме продолжительной перегрузки увеличение максимального объёма очереди приводит к повышению времени нахождения пакетов в ней, но не обеспечивает снижение потерь пакетов.

5. Механизм приоритизации обслуживания пакетов во входной очереди межсетевого экрана позволяет снизить потери пакетов приоритетного потока и в несколько раз снизить их задержку в очереди. Так, в условиях 30% перегрузки при доле входящего потока приоритетных пакетов в 15% от общего потока, включение механизмов приоритизации обслуживания снижает:

- потери приоритетного потока с 23% до нулевого уровня при повышении потерь неприоритетного потока на 4 %;
- среднее время нахождение приоритетного пакета в очереди в 21 раз (с 105 до 5 мкс) при увеличении этой величины для неприоритетных пакетов на 19%.

Личный вклад. Результаты диссертационной работы получены автором самостоятельно, математические процедуры и программные средства разработаны при его непосредственном участии.

Публикации. По материалам диссертационной работы опубликовано 12 печатных работ, из них 4 – в рецензируемых журналах, рекомендованных ВАК Минобрнауки России.

Результаты исследования соответствуют следующим пунктам паспорта научной специальности 05.12.13 – «Системы, сети и устройства телекоммуникаций»:

– 3: разработка эффективных путей развития и совершенствования архитектуры сетей и систем телекоммуникаций и входящих в них устройств;

– 10: исследование и разработка новых методов защиты информации и обеспечения информационной безопасности в сетях, системах и устройствах телекоммуникаций;

– 14: разработка методов исследования, моделирования и проектирования сетей, систем и устройств телекоммуникаций.

Структура и объем работы. Диссертация состоит из введения, 5 разделов, заключения, списка сокращений и обозначений, списка литературы, списка иллюстративного материала и 7 приложений. Основные результаты изложены на 148 страницах, в том числе на 44 рисунках и в 11 таблицах. Дополнительные сведения изложены на 76 страницах. В библиографию включено 143 источников на русском и английском языках.

РАЗДЕЛ 1. АНАЛИЗ МЕТОДОВ ПОВЫШЕНИЯ ПРОИЗВОДИТЕЛЬНОСТИ МЕЖСЕТЕВЫХ ЭКРАНОВ

1.1. Стандартизация и регулирование применения межсетевых экранов

1.1.1. Краткая история развития межсетевых экранов

Определение. Существует множество организаций, регулирующих применение и распространение оборудования, выполняющего функции межсетевого экранирования. Продемонстрируем определения трёх организаций.

Федеральная служба по техническому и экспортному регулированию России (ФСТЭК России): межсетевым экраном считается программное и программно-техническое средство, реализующее функции контроля и фильтрации в соответствии с заданными правилами, проходящих через него информационных потоков, используемые в целях обеспечения защиты информации (не криптографическими методами) [9].

National Institute of Standards and Technology (NIST; США): межсетевой экран – это устройство или программа, которая контролирует потоки сетевого трафика между сетями или хостами, которые используют различные политики безопасности [36].

РОССТАНДАРТ: межсетевой экран – вид барьера безопасности, размещенного между различными сетевыми средами, состоящего из специализированного устройства или совокупности нескольких компонентов и технических приемов, через который должен проходить весь трафик из одной сетевой среды в другую и, наоборот, при этом пропускается только авторизованный трафик, соответствующий местной политике безопасности [37].

Разделим понятия межсетевого экрана и межсетевого экранирования и сформируем их обобщённые понятия.

Межсетевое экранирование – это функционал, предназначенный для разграничения взаимодействия между хостами, сегментами одной сети или различных сетей, на основе заранее сформированных правил (политик безопасности).

Межсетевой экран (МЭ; англ. firewall) – это специализированное программное или программно-техническое средство, основной функцией которого является межсетевое экранирование. Большая часть сетевого оборудования, такого как маршрутизаторы и коммутаторы, также обладает функцией межсетевого экранирования в упрощённом виде (по сравнению с МЭ). Современные межсетевые экраны выполняют более широкий спектр функций обеспечения безопасности помимо межсетевого экранирования.

Первые упоминания. Упоминание о межсетевом экранировании, как функции фильтрации трафика, встречается примерно с конца 1980 года. В это время МЭ как самостоятельный продукт не существовал на рынке. В те годы термин «межсетевое экранирование» применяли к фильтрации широковещательного трафика протоколов настройки сети [34, 35].

У истоков создания межсетевых экранов стояли такие специалисты, как J. Mogul, P. Vixie, B. Reid, F. Avolio и B. Chapman, R. Braden, D Clark, S. Crocker, C. Huitema. Их работы легли в описанные выше решения, а также их идеи широко использовались для разработки первых коммерческих МЭ.

Появление необходимости в межсетевом экранировании как функции обеспечения сетевой безопасности. Во многом современные МЭ начали разрабатываться в связи событием 1988 года, потрясшего американскую сеть ARPANET, которая являлась прототипом сети Internet. Этим событием являлся «Червь Морриса». Это был первый в мире зафиксированный сетевой червь. Он парализовал работу 6000 узлов сети ARPANET, а ущерб от червя Морриса был оценён примерно в 96,5 миллионов долларов. Анализ вируса был произведён широким числом специалистов и представлен в открытых отчётах [38, 39]. В результате этих событий мировое сообщество начало задумываться об обеспечении сетевой безопасности информационных систем.

Концепция современных межсетевых экранов. В 1992 году Steve Vellovin, работавший в американской компании AT&T, создал комплекс устройств безопасности, задачей которых являлось обнаружение действий злоумышленника и перевод его действий в область сети, названную им «тюрьмой». Данная область

эмулировала настоящую сеть, тем самым обманывая злоумышленников, которые считали, что проникновение удалось, и продолжали действовать.

Команда AT&T наблюдала за действиями различных злоумышленников, а через некоторое время опубликовала отчёт о мониторинге комплекса AT&T. На основании этого отчёта были сделаны выводы о множестве уязвимых мест в операционных системах того времени. Затем были разработаны общие принципы блокирования трафика, для ограничения его доступа к уязвимым функциям от недостоверных пользователей и приложений. Фактически это были требования к функциональности первых межсетевых экранов. В этом же 1992 году был выпущен первый коммерческий МЭ «SEAL» компанией DEC [34].

В 1994 году была выпущена рекомендация RFC1636 «Безопасность в архитектуре сети Интернет» [40]. В этом документе был прописан статус, функциональность и архитектура МЭ, а также архитектура безопасности сети при его применении. С этого момента можно отсчитывать современную историю МЭ.

1.1.2. Типы стандартов, рекомендаций и их применение

Регуляторы и организации, выполняющие работы по стандартизации, формируют два типа стандартов и рекомендаций. Первый – стандарты и рекомендации по конкретной технологии/протоколу/продукту. Второй – стандарты и рекомендации концептуального характера, где описываются общие подходы без конкретизации, а также нормы и технические требования по производительности сетей передачи данных.

Рекомендации из смежных областей содержат информацию, необходимую для эксплуатации МЭ. Например, рекомендации по протоколам аутентификации необходимы для выполнения настройки аутентификации пользователей на пограничном шлюзе, чью роль может выполнять МЭ.

По этой причине в данной работе могут упоминаться рекомендации как напрямую, так и косвенно связанные с МЭ.

Обзор организаций по стандартизации, действующих в области информационной безопасности, представлен автором в статье [41]. Краткий обзор

рисков информации, обрабатываемой в единой среде электросвязи России, дан автором в [42].

1.1.3. Организации по стандартизации и регулированию применения межсетевых экранов

Международный союз электросвязи – Сектор стандартизации электросвязи. *Международный союз электросвязи* (МСЭ, англ. International Telecommunication Union, ITU) был создан в 1865 году, как Международный телеграфный союз [43].

В рамках настоящей работы наибольший интерес представляют стандарты подразделения МСЭ называемого – *сектор стандартизации электросвязи МСЭ* (МСЭ-Т, англ. Telecommunication Standardization Sector, ITU-T) [44] и её 13 и 17 исследовательские группы [45].

Рекомендации МСЭ-Т предоставляют достаточно широкий спектр информации смежной по тематике с межсетевым экранированием. Также они позволяют лучше понять картину информационной безопасности сетей связи в целом. Рекомендации по обеспечению информационной безопасности в основном находятся в сериях: X.800, X.1000, Y.700, Y.2700. Серия E.800 описывает общую концепцию качества телекоммуникационных услуг. Серия Y.1500 описывает механизмы качества услуг и производительности сети. Также выделим серию Y.3000, затрагивающую аспекты Будущих сетей и глобальной информационной инфраструктуры.

Анализ тенденций разработки рекомендаций МСЭ-Т по информационной безопасности сетей связи проведён автором в [46].

ISO/IEC – Международная организация по стандартизации и международная электротехническая комиссия. *Международная организация по стандартизации* – ИСО (англ. International Organization for Standardization, ISO) занимается разработкой стандартов почти для всех сфер деятельности человека от здравоохранения и пищевой промышленности до сферы информационных технологий, исключая лишь сферы электротехники и

электроники, чьей стандартизаций занимается *Международная электротехническая комиссия* – МЭК (англ. International Electrotechnical Commission, IEC). Некоторые виды работ выполняются совместными усилиями ISO и IEC. Кроме стандартизации, ИСО занимается проблемами сертификации продукции [45]. ИСО вместе с МЭК ведёт разработку большого числа стандартов по комплексному подходу к обеспечению информационной безопасности. Информационной безопасности посвящены 18 и 27 серии рекомендаций (27 серия является приемником 18 серии).

IETF – Инженерный совет Интернета (англ. Internet Engineering Task Force, IETF) — открытое международное сообщество проектировщиков, учёных, сетевых операторов и провайдеров, созданное *Советом по архитектуре Интернета* (англ. Internet Architecture Board, англ. IAB) в 1986 году и занимающееся развитием протоколов и архитектуры Интернета [48].

Вся техническая работа осуществляется в рабочих группах, занимающихся конкретной тематикой (например, вопросами маршрутизации, передачи данных, безопасности и т. д.).

IEEE – Институт инженеров электротехников и электроники. *Институт инженеров и электротехников и электроники* (англ. Institute of Electrical and Electronics Engineers, IEEE) — международная некоммерческая ассоциация специалистов в области разработки стандартов по радиоэлектронике, электротехнике и аппаратному обеспечению вычислительных систем и сетей [49].

Под патронажем IEEE выпускаются стандарты и публикуются научные работы. Напрямую связанных с МЭ стандартов у IEEE нет, но есть косвенно связанные стандарты как по аппаратному, так и программному обеспечению. Также в рамках IEEE публикуется достаточное количество научных работ прямо или косвенно связанных с МЭ, например, работы [13, 16, 19, 20].

ФСТЭК России и ФСБ России. ФСТЭК России выполняет государственное регулирование и надзор в области технической защиты информации, а также в вопросах экспорта/импорта средств защиты информации [50].

С целью регулирования применения МЭ ФСТЭК России было выпущено несколько нормативно-распорядительных и руководящих документов.

Федеральная служба безопасности России (ФСБ России) выполняет государственное регулирование применения криптографических средств защиты информации и в целом защиты информации, содержащей сведения, составляющие государственную тайну. Внутренняя структура, выполняющая эти работы, именуется *Центром по лицензированию, сертификации и защите государственной тайны ФСБ России* (ЦЛСЗ ФСБ России). С полным перечнем исполняемых ЦЛСЗ ФСБ России функций можно ознакомиться в [51]. Требования центра недоступны для демонстрации в свободном доступе.

РОССТАНДАРТ – Федеральное агентство по техническому регулированию и метрологии. Федеральное агентство по техническому регулированию и метрологии входит в систему федеральных органов исполнительной власти Российской Федерации и находится в ведении Министерства промышленности и торговли Российской Федерации [52].

В настоящее время РОССТАНДАРТ в целях унификации требований в ИТ-сфере выполняет перевод и адаптацию стандартов ISO/IEC и принимает их в качестве ГОСТ Р. Маркировка документа такая же, что и у оригинала, то есть ГОСТ Р ИСО/МЭК серий 18 и 27.

Документы РОССТАНДАРТа отстают во времени от работ ISO/IEC приблизительно на одну ревизию документа вследствие необходимости перевода.

NIST – Национальный институт стандартов и технологий (англ. National Institute of Standards and Technology, NIST) является подразделением министерства торговли США [53].

NIST вместе с Американским национальным институтом стандартов (ANSI) участвует в разработке стандартов и спецификаций к программным решениям, используемым как в государственном секторе США, так и имеющим коммерческое применение.

Под патронажем NIST выпущено несколько документов, напрямую связанных с применением МЭ. Информационной безопасности посвящена 800 серия стандартов, а 1800 серия посвящена кибербезопасности.

Автором была рассмотрена деятельность организаций по стандартизации в работах [41, 46], риски информации, обрабатываемой в единой среде электросвязи России [42], исследованы вопросы информационной безопасности среды облачных вычислений в призме фильтрации трафика в работе [54], а также проанализирована актуальность разработки алгоритма автокоррекции списков правил фильтрации трафика в работах [55-57].

1.2. Анализ существующих методов повышения производительности межсетевых экранов и их адаптации к реалиям Будущих сетей

1.2.1. Анализ перечня возможных задач в предметной области

Вопросами повышения производительности и адаптации МЭ к условиям эксплуатации в рамках Будущих сетей занимаются как производители МЭ, так и отдельные исследовательские организации и свободные исследователи.

Условно можно разделить эту группу задач на три отдельных подгруппы:

– разработка новых и повышение производительности существующих решений, формирующих аппаратную платформу МЭ;

– разработка нового и повышение производительности существующих алгоритмов и *программного обеспечения* (ПО) МЭ;

– адаптация оборудования к эксплуатации в условиях изменяющейся архитектуры сетей связи и концепции Будущих сетей.

1.2.2. Задачи разработки новых и повышения производительности существующих аппаратных платформ межсетевых экранов

Аппаратная платформа МЭ является комплексом элементов, таких как материнская плата, процессоры (центральный, вспомогательные и т.п.), сетевые карты, оперативная память, системы ввода/вывода, блоки питания и охлаждения, системы пыле и влагозащиты и т.д. Напрямую влияет на производительность только часть этих элементов.

Далее кратко рассмотрен ряд наиболее ярких специализированных технических решений последнего десятилетия, которые используются в аппаратной платформе сетевого оборудования, в том числе МЭ.

Эволюция процессоров сетевого оборудования. До настоящего времени большая часть аппаратной обработки трафика на скоростях свыше 1 Гбит/с в сетевом оборудовании выполнялась на *интегральных схемах специального назначения* (англ. application-specific integrated circuit; ASIC). Обработчик типа ASIC исправно выполнял свою роль, но имел ряд недостатков, например, высокая стоимость разработки при малых партиях продукции и невозможность изменять функциональность ASIC без его непосредственной замены.

Ввиду быстрой эволюции сетей передачи данных и необходимости быстрого внедрения новых услуг, производители сетевого оборудования начали вести разработку более универсальных обработчиков, названных *«модулями сетевой обработки»* или *«сетевыми сопроцессорами»* (англ. network processor unit). Их разработка началась ещё в 2000 годах, но длилась достаточно долго, крупные производители представили первые коммерческие решения примерно к 2007-2010 годам, которые ознаменовались выходом маршрутизаторов серии Juniper MX на процессоре Juniper Trio [58] и линейки маршрутизаторов Cisco ASR на процессоре Cisco Flow Processor [59]. В настоящее время всё ещё идёт адаптация и развитие этих типов процессоров, а их применение ограничено.

Уточним, что программная обработка неспециализированными центральными процессорами до сих пор занимает всю нишу решений SOHO и большую часть Branch. В Enterprise-решениях широко применяются ASIC’и.

Память типов CAM – Content Addressable memory и TCAM – ternary CAM. В целях увеличения скорости проведения операций поиска по оперативной памяти *RAM* (англ. random access memory, рус. запоминающее устройство с произвольной выборкой) была создана её замена, технология *CAM* (англ. content addressable memory, рус. ассоциативное запоминающее устройство) [60].

Ключевым отличием CAM от RAM является то, что в RAM для возврата информации используется адрес ячейки памяти, в которой хранятся данные, а в

CAM поиск информации происходит по всему объёму памяти в соответствии с заданным ключевым словом, например, MAC-адресом.

Память CAM характеризуется высокой производительностью, позволяет производить распараллеливание поиска, но имеет высокую стоимость, большое потребление электроэнергии и ограничена по объёму. Ввиду этих ограничений чипы CAM используют в сетевом оборудовании в малых объёмах памяти, около 0,5-2 Мбайт.

Следующим витком эволюции стало появление TCAM [61]. Ограничением CAM был тот факт, что при поиске битам кодового слова можно задать префикс «правда» / «лож», т.е. должен совпасть бит или нет. В TCAM было добавлено третье состояние префикс «любое значение», т.е. при поиске не важно, какое состояние у этого бита. Это позволило эффективнее использовать TCAM для таблиц маршрутизации, *ACL* (англ. access list, рус. лист контроля доступа), таблиц *QoS* и другой информации свыше L3 по модели OSI/ISO. Описание и применение CAM и TCAM в оборудовании Cisco представлено в [62].

1.2.3. Задачи разработки нового и повышения производительности существующего программного обеспечения межсетевых экранов

В этой области «безграничное» количество возможных задач, однако, разработка какого-либо полезного ПО или алгоритмов для МЭ является достаточно сложной задачей для одного специалиста. Конкуренция с крупными разработчиками и производителями зачастую невозможна.

Рассмотрим ряд направлений исследований, по которым возможно выполнение работ.

Разработка операционных систем МЭ. Разработку операционных систем и драйверов для сетевого оборудования затрагивать не будем. Основную часть работ выполняют производители оборудования.

Разработка алгоритмов криптографической защиты информации. В области обеспечения безопасности передачи данных несколько особняком стоят вопросы криптографической защиты. Их касаться в работе не будем как по

причине сложности выполнения работ, так и закрытого характера подобных исследований.

Алгоритмы быстрой классификации пакетов. Как альтернатива разработке аппаратных решений, таких как CAM и TCAM, разрабатываются алгоритмы быстрой классификации пакетов, которые могли бы улучшить скорость классификации пакетов, оперируя обычной RAM. Качественный общий обзор такого типа алгоритмов выполнен в [63].

Наиболее известные из существующих алгоритмов такого типа являются *ABV* (англ. aggregated bit vector), *HiCuts* (Hierarchical Cutting), *RFC* (Recursive Flow Classification). В этой области исследований постоянно происходят анонсы новых или улучшенных существующих алгоритмов быстрой классификации пакетов.

Представленные алгоритмы относятся к алгоритмам древа принятия решений и декомпозиции. Оба типа алгоритмов дают в результате древовидную разветвлённую структуру, по которой происходит поиск и классификация пакета.

Основным минусом этих алгоритмов является геометрическое нарастание размера древа принятия решений при увеличении количества классификаторов. Увеличение древа принятия решений замедляет работу алгоритма и занимает память. Существуют экономящие память алгоритмы, но они работают медленнее. Классификаторами могут быть MAC-, IP-адреса, порты, служебные биты и т.п.

Работы такого типа ведутся продолжительное время, однако замены CAM и TCAM на подобные алгоритмы в промышленном масштабе не наблюдается.

С одной стороны, разработка подобных алгоритмов является перспективной областью исследований. Однако этим типом задач широко занимаются специалисты IEEE и конкурировать со сложившимися исследовательскими группами IEEE можно считать нерациональным.

Алгоритмы предотвращения перегрузок. Такой тип алгоритмов влияет на работу пакетных очередей. Больше информации об алгоритмах управления очередями представлено в пункте 2.1.3 и во 2 разделе диссертации.

В 2016 в университете ФГБОУ ВО ПГУТИ г. Самара, защищена диссертация по разработке алгоритма предотвращения перегрузок,

функционирующего на основе нечёткой логики [64]. О работе аналогичных алгоритмов представлена информация в пункте 2.1.4.

1.2.4. Анализ задач по адаптации существующего оборудования к эксплуатации в условиях Будущих сетей

Современные сети передачи данных – это быстро меняющая среда, в рамках которой функционируют различные типы оборудования, в том числе МЭ.

Скорость изменения этой среды высока, что обусловлено постоянным появлением новых услуг и технологий передачи данных. Решения задач по оценке необходимости адаптации оборудования к будущим условиям эксплуатации позволяют разработчикам подготовиться к изменению среды функционирования и заранее внести корректировки в своё оборудование. Исследования в этой области могут доказать или опровергнуть необходимость внесения изменений в существующее оборудование и его программное обеспечение.

Современные сети передачи данных строятся по концепции, именуемой «*Сети следующего поколения*» (англ. Next generation networks). Её эволюция привела к созданию концепции Будущих сетей. Концепция Будущих сетей предложена МСЭ и прорабатывается группой IMT-2020. МСЭ-Т полагает, что концепция будет стандартизирована до 2020 года. Описанию концепции Будущих сетей посвящена монография [7], а также работы [6, 8].

Группа IMT-2020, прорабатывающая технологию 5G, в своём отчёте [3] и рекомендации ITU-R M.2083-0 [4] определила, что при разработке технологической базы 5G задержку из конца в конец в рамках одного сегмента сети необходимо снизить с 10 до 1 мс. Чтобы удовлетворить подобные требования придётся бороться за производительность каждого узла сети.

Помимо технологических требований существуют и требования, диктуемые внедрением новых услуг. Примером таких услуг может быть широкий рост рынка носимой медицины. Аналитическая компания ABI research, Inc. представила 1 февраля 2017 г. свой прогноз о росте рынка носимой медицины с непрерывным подключением к сети связи с 2016 г. по 2021 г. на 35%, при этом такие устройства

составят до 60% от общего объёма изделий носимой медицины [65]. Стоит понимать, что качественный непрерывный доступ к сети таких устройств необходим для контроля жизненно важных параметров здоровья пациентов. Безусловно, не весь объём подобных устройств передаёт данные, требующие мгновенную реакцию, но с ростом этого рынка требования будут расти. При этом надо понимать, что в отличие от обеспечения качества обслуживания до конкретной стационарного узла сети потребуется обеспечить передачу данных с гарантированным качеством до узлов наиболее близких к клиенту, что в современных реалиях мира мобильных устройств зачастую не обеспечивается. Со временем будет появляться всё больше и больше услуг, которые будут требовательны к качеству передачи данных.

Для удовлетворения потребностей в качестве услуг потребуется распространить действие механизмов QoS ближе к клиентским устройствам. Стоит также отметить, что и общие требования к сети будут расти.

Значительная часть МЭ и другого оборудования сетей SOHO, Branch и Enterprise классов, которые могут быть установлены на последней миле у клиента, не поддерживают механизмы приоритизации обслуживания трафика. Следовательно и клиенты, находящиеся за подобным оборудованием, не получат определённый уровень качества услуги.

Работ, посвящённых моделированию функционирования МЭ в среде сетей с гарантированным качеством услуг, не было найдено в свободном доступе. Это связанно с тем, что математические модели с приоритетами считаются одними из наиболее сложных классов моделей в теории массового обслуживания [66].

Отличия МЭ от маршрутизаторов и коммутаторов характеризуют бóльшей удельной задержкой обслуживания пакетов, порождаемой выполнением функций обеспечения безопасности.

Как правило, МЭ исполняет роль пограничного оборудования ЛВС клиента. Наиболее актуально применение приоритизации обслуживания трафика в МЭ крупных офисных сетей или сетей торговых центров, вследствие того, что в них содержатся большое число разнородных клиентов, которым необходимы

различные сервисы. Трафик пользователей будет проходить сквозь МЭ торгового центра или офиса, если посетители будут пользоваться Wi-Fi-сетью или сотами мобильной связи, подключенными к ЛВС. В таком случае их трафик будет передаваться поверх интернет-соединения этого торгового центра или бизнес-центра.

1.3. Анализ функционирования механизмов качества обслуживания в современных сетях

1.3.1. Применения механизмов качества обслуживания в современных сетях передачи данных

Необходимость применения механизмов качества обслуживания. Стандарт МСЭ-Т Y.1541 [67] является базовым стандартом, описывающим общую архитектуру функционирования механизмов QoS в рамках сети передачи данных. Стандарт даёт следующее обоснование необходимости применения механизмов качества обслуживания: потребители нуждаются в таких уровнях сетевых показателей качества, которые в сочетании с их хостами, оконечным оборудованием и другими устройствами обеспечивают удовлетворительную поддержку их приложений.

Из этого определения становится ясно, что потребителю требуется определённый уровень обслуживания, а оператор связи, в свою очередь, обладает техническими возможностями по предоставлению своему пользователю запрашиваемого уровня качества. Оператор связи рассматривает обеспечение гарантированного качества обслуживания как дополнительную услугу и имеет с неё дополнительный доход. Вследствие чего оператор заинтересован, чтобы эта услуга была востребована, а пользователи были довольны качеством её предоставления.

Применение механизмов качества обслуживания в современных сетях связи. Существуют наборы стандартизованных классов QoS, определяющих рабочие характеристики сети, которые необходимо обеспечить. Эти классы QoS описаны в рекомендации Y.1541. Представленные в рекомендации классы QoS

являются базовыми и расширяются операторами связи самостоятельно для предоставления более дифференцированных услуг.

На рисунке 1.1 представлена схема канала передачи данных *UNI-UNI* (user network interface, сетевой интерфейс пользователя), приведённая в Y.1541.

То, что обозначено на рисунке как оконечное оборудование, может быть представлено как оконечным устройством, так и пограничным устройством. За пограничным устройством может находиться некая ЛВС клиента. Как упоминалось в предыдущем пункте, пограничное оборудование ЛВС обычно представлено МЭ или оборудованием, выполняющим функции межсетевого экранирования. Устройство, выполняющее функции пограничного маршрутизатора провайдера, может быть представлено обычным коммутатором, маршрутизатором или специализированным МЭ, но, как правило, на границе транспортной сети и ЛВС клиентов или транспортных сетей двух провайдеров используют маршрутизаторы.

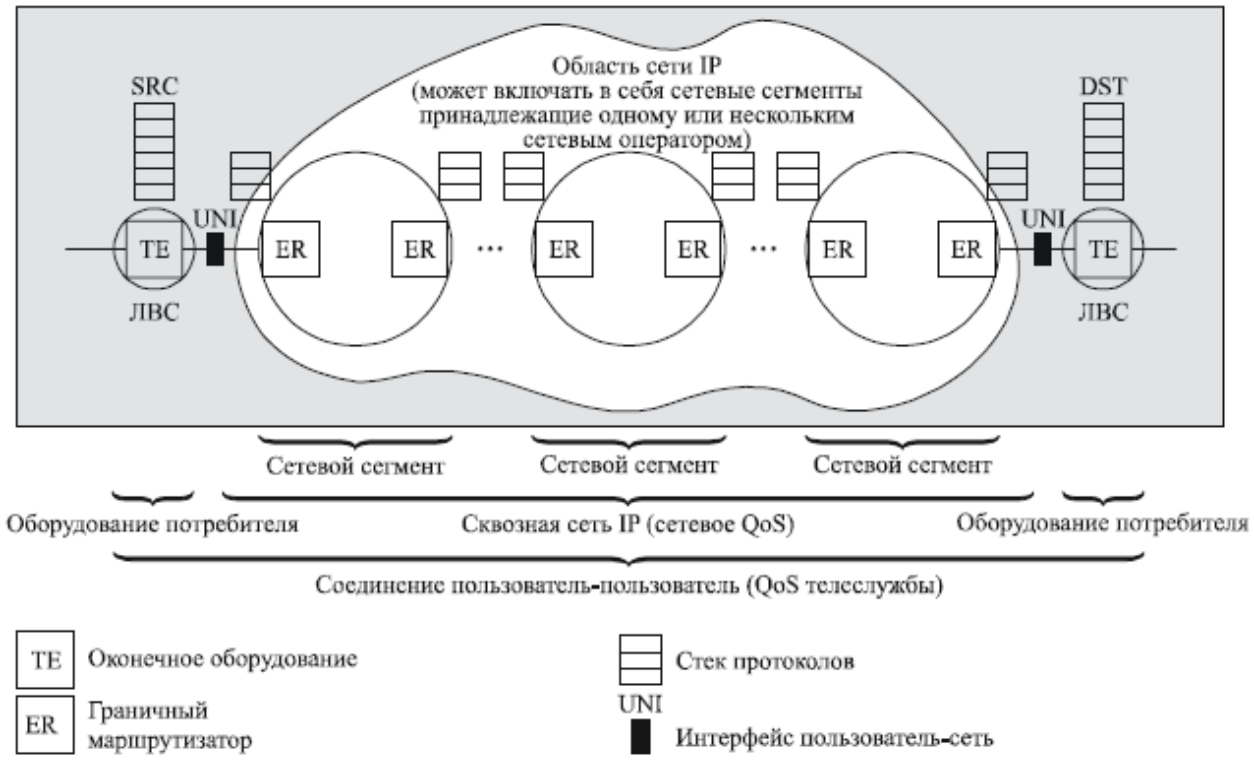


Рисунок 1.1 – Канал передачи данных UNI-UNI

Под доменом качества обслуживания будем понимать некое пространство сети, в рамках которого действуют механизмы QoS, определены классы

обслуживания и правила обработки трафика различных классов. Существует два основных способа построения домена QoS. Первый способ – UNI-UNI, т.е. домен QoS действует внутри сети провайдера между *пограничными маршрутизаторами оператора связи* (ER), на которых происходит классификация и маркировка трафика. Второй способ – между клиентским оборудованием, то есть end-to-end, который почти не рассматривается в Y.1541. Во втором способе пользователь и оператор заранее договариваются о передаваемых классах трафика и метках. Второй способ, в свою очередь, может распространять домен QoS внутрь ЛВС клиента, но такое в настоящее время практикуется редко.

В рамках диссертационной работы рассматривается именно второй способ построения домена QoS потому, что МЭ должен находиться внутри домена QoS.

То есть рассматриваемый МЭ классифицирует трафик от ЛВС или доверяет меткам обслуживания клиентов, а также доверяет меткам обслуживания пограничного маршрутизатора оператора. В свою очередь, пограничный маршрутизатор оператора доверяет меткам трафика, полученным от МЭ.

1.3.2. Межсетевые экраны и поддержка механизмов приоритизации обслуживания трафика

Если рассматривается МЭ, находящийся внутри домена QoS, то он должен быть способен поддерживать необходимую функциональность, иначе он будет обслуживать трафик без приоритизации. Не все МЭ поддерживают механизмы приоритизации обслуживания. При этом их поддержка распределена между оборудованием разных классов и производителей неравномерно.

Если рассматривать такие классы оборудования, как SOHO, Branch, Enterprise, Data center, Service provided, то большая часть МЭ классов SOHO и Branch не обладает поддержкой механизмов приоритизации.

В линейке Enterprise решений, подходящих для ЛВС крупных организаций, поддержка приоритизации обслуживания не слишком распространена. При этом она применяет в выходных очередях, а не входных, что должно быть менее эффективным в случае перегрузок на межсетевых экранах. Несколько продуктов

линейки Enterprise различных производителей рассмотрены во втором разделе диссертации. В пунктах 2.1.6-2.1.7 и в приложении А рассмотрены принципы функционирования нескольких МЭ Enterprise-класса.

В оборудования классов Datacenter и Service provider поддержка приоритизации обслуживания распространена почти во всех решениях.

Кажется неоправданным малое распространение поддержки механизмов приоритизации обслуживания в МЭ классов Branch и Enterprise, так как на долю такого типа оборудования приходится значительная часть рынка оборудования, выполняющего роль пограничного оборудования ЛВС.

Вышесказанное обосновывает необходимость проверки влияния применения механизмов приоритизации во входных очередях МЭ на его показатели производительности.

Выводы раздела

1. Результаты анализа документов профильных организаций по стандартизации Будущих сетей и сетей 5G демонстрируют увеличение требований к показателям производительности сетей связи и качеству обслуживания в них. Так, задержки передачи трафика между оконечными устройствами в рамках сегмента сети (за исключением задержки распространения сигнала) планируется снизить до уровня 1 мс.

2. Повышение требований к качеству передачи данных в Будущих сетях и сетях 5G обуславливает распространение домена QoS в сторону оконечного оборудования (клиентских устройств). Это требует широкой поддержки механизмов QoS и, в частности, механизмов приоритизации обслуживания у оборудования корпоративных классов: SOHO, Branch и Enterprise.

3. Анализ публикаций, выполненных в области эксплуатации и разработки МЭ, выявил, что вопросам функционирования МЭ в условиях приоритизации обслуживания трафика посвящено малое число исследовательских работ. Математических моделей поведения МЭ в условиях приоритизации обслуживания трафика в их входных очередях не обнаружено.

РАЗДЕЛ 2. АНАЛИЗ ПРИНЦИПОВ ОБРАБОТКИ ПАКЕТОВ В МЕЖСЕТЕВЫХ ЭКРАНАХ

2.1. Анализ различных подходов к организации процессов обслуживания пакетов в межсетевых экранах

2.1.1. Определение процесса обслуживания пакетов и причины наличия различных подходов к организации этого процесса

Под подходом к организации процесса обслуживания пакета понимается набор операций (их очерёдность), который выполняется над пакетом при его обслуживании рассматриваемым устройством. В зарубежной литературе часто используются следующие термины «packet flow throw device», «packet processing order», «order of operation», «packet transfer process».

Наличие различных подходов к организации процессов обслуживания пакетов как в МЭ, так и другом сетевом оборудовании объясняется несколькими факторами. Каждый производитель оборудования в процессе разработки своих продуктов принимал различные концептуальные решения, которые в свою очередь развивались или заменялись, вследствие наличия или отсутствия положительного опыта использования.

Каждый производитель имел изначальную базу клиентов, которые имели свои требования к продуктам и передавали его разработчикам в качестве отзывов. В дополнении к этому стоит считать значительным фактором целевое место применения продуктов. Например, серверный межсетевой экран может отличаться от пограничного по необходимому набору функций и архитектуре вследствие того, что в различных местах применения могут быть более эффективны те или иные подходы к обслуживанию пакетов. Современное многообразие подходов к обслуживанию пакетов в МЭ объясняется их продолжительным процессом «эволюции».

В рамках работы не будут рассматриваться операции обслуживания с применением технологии *виртуальных частных сетей* (VPN).

Проведём анализ ряда концептуальных решений прямо или косвенно касающихся работы механизмов качества обслуживания и особенностей обслуживания пакетов МЭ, а именно:

– очереди и работа с памятью;

– архитектура применения механизмов качества обслуживания;

– *фильтрация пакетов с учётом и без учёта состояния соединения* (stateless, stateful filtering).

2.1.2. Рассматриваемый класс оборудования

Анализа различных архитектур обработки пакетов МЭ производится по документации производителей. Анализ производится среди специализированных межсетевых экранов Enterprise-класса. Ограничение вводится по нескольким причинам: высокопроизводительные решения классов Service provider и Data center заметно сложнее «младших» классов оборудования, в них большую роль играет решения аппаратного уровня: манипуляция с памятью, узкая специализация процессоров и т.п., а также по ним заметно меньше открытой документации технического характера.

2.1.3. Очереди и работа с памятью

Очереди – неотъемлемый элемент буферизации пакетов и механизмов приоритизации их обслуживания и отправки. Принципы их работы важны при составлении будущей модели. Основная часть информации этого пункта заимствована из источников [68-71].

Разделим понятие очереди и буфера памяти. Очередью будем считать логический объект, в котором хранятся пакеты и за счёт которого реализуются различные дисциплины обслуживания пакетов, находящихся в ней. Буфером памяти будем считать комплекс программно-аппаратных мощностей оборудования, выделяемых для работы очередей и других элементов МЭ, то есть хранения данных (пакетов, алгоритмы функционирования МЭ и т.п.). Иначе говоря, очередь – логический элемент, хранящийся в буфере памяти.

Терминология, может несколько отличатся от производителя к производителю, но в целом остаётся верной и логически понятной.

Разделим понятия *аппаратной очереди* (англ. hardware queue) и *программной очереди* (software queue). Аппаратной очередью будем считать такую очередь, под которую буфер памяти выделен аппаратно, например, в предустановленном выделенном чипе памяти. Такой буфер памяти не меняет свой объём. Программной очередью будем считать такую очередь, под которую операционная система МЭ будет выделять необходимый динамический объём памяти из общего физически ограниченного объёма.

Аппаратная очередь реализуется за счёт буфера памяти сетевой карты и функционирует на основе не настраиваемых алгоритмов. Буферы памяти сетевой карты также называют *интерфейсными буферами* (англ. interface buffer), а аппаратные очереди – *интерфейсными очередями* (англ. interface queues). Аппаратные очереди поддерживают дисциплину обслуживания FIFO, зачастую в них применяются простые алгоритмы управления очередями, такие как tail drop. Аппаратные очереди представлены двумя типами очередей: *очередями приёма* (англ. receive queues) и очередями отправки (англ. transmit queues) пакетов.

Аппаратные очереди могут храниться в двух структурах памяти: *буфере сетевой карты* (англ. interface FIFO) и буфере Rx/TxRings. Очереди Rx/TxRing именуются на основе слов Rx – *receive* (получение), Tx – *transmit* (отправка). Слово *ring* (кольцо) обозначает способ организации памяти в буфере, именуемый *кольцевым или циклическим буфером* (англ. ring-buffer).

Каждому интерфейсу выделяется один буфер сетевой карты, один RxRing, один TxRing. В некоторых случаях на интерфейсной плате устанавливается общий буфер памяти для группы интерфейсов, который делится между ними. Очередь сетевой карты работает напрямую с пакетом, который прибывает или отправляется устройством в данный момент. В RxRing пакет попадает после получения пакета от буфера сетевой карты на время до переноса пакета в *общий системный буфер* (англ. shared/main memory system buffer). В TxRing пакеты попадает из общего системного буфера на время занятия буфера сетевой карты.

Буфер памяти Rx/TxRing реализуется в общей памяти устройства программно или аппаратно на сетевой карте (интерфейсной плате).

Программные очереди реализуются за счёт общей системной памяти и настраиваемых алгоритмов, например, сложных алгоритмов управления и предотвращения перегрузок и управления ресурсами. Программные очереди также называют *очередями системного интерфейса* (англ. system interface queues), которые, в свою очередь, делятся на *входные очереди* (англ. input hold queues) и *выходные очереди* (англ. output hold queues). Входная очередь хранит пакеты после их получения до их обработки устройством. Выходная очередь хранит пакеты после обработки и до начала процесса их отправки.

Программные очереди хранятся в структурах данных, называемых *внутренними буферами* (англ. internal buffer). К внутренним буферам относятся *заголовок буферов* (англ. buffers header), общие системные буферы, *общий интерфейсный буфер* (англ. shared interface buffer). Заголовок буферов – структура, хранящая информацию об остальных буферах, распределении объектов в памяти, размерами, занятостью и т.п. Общий системный буфер используется для хранения пакетов. Память выделяется системой по необходимости и, как правило, представляют собой ячейки типовых размеров, таких, чтобы пакеты различных размеров гарантированно поместились в ячейки (с избытком). Такой подход неэкономно использует память, но имеет высокую производительность. Общий интерфейсный буфер используется для хранения (обмена) пакетов между драйвером сетевой карты и *трактом коммутации* (англ. switching path), в отличие от других программных буферов не настраивается вручную, а его объём зависит от параметра MTU. Общий интерфейсный буфер используется в процессе *быстрой коммутации* (англ. fast switching), далее он не рассматривается.

Общий путь прохождения пакета по очередям и буферам памяти представлен на рисунке 2.1. Движение пакета по очередям интуитивно понятно, но с хранением пакета в буферах памяти есть нюансы. После попадания пакета в RxRing пакет забирается из него драйвером устройства, производится проверка

фрейма (L2) на соответствие контрольной суммы. Если всё в порядке, то производится обращение к заголовку буферов для проверки наличия свободного пространства для сохранения пакета. Если свободная ячейка есть, то пакет помещается свободную ячейку входной очереди, если нет, то пакет сбрасывается.

Обычно на этом этапе нет манипуляций с памятью, в очередь помещается служебная информация, сам пакет не перезаписывается. Затем пакет попадает на обслуживание и после него переносится в выходную очередь, а из неё – в TxRing. На этапах до обслуживания при переполнении пакет может быть сброшен, также пакет может быть сброшен на этапе обслуживания.

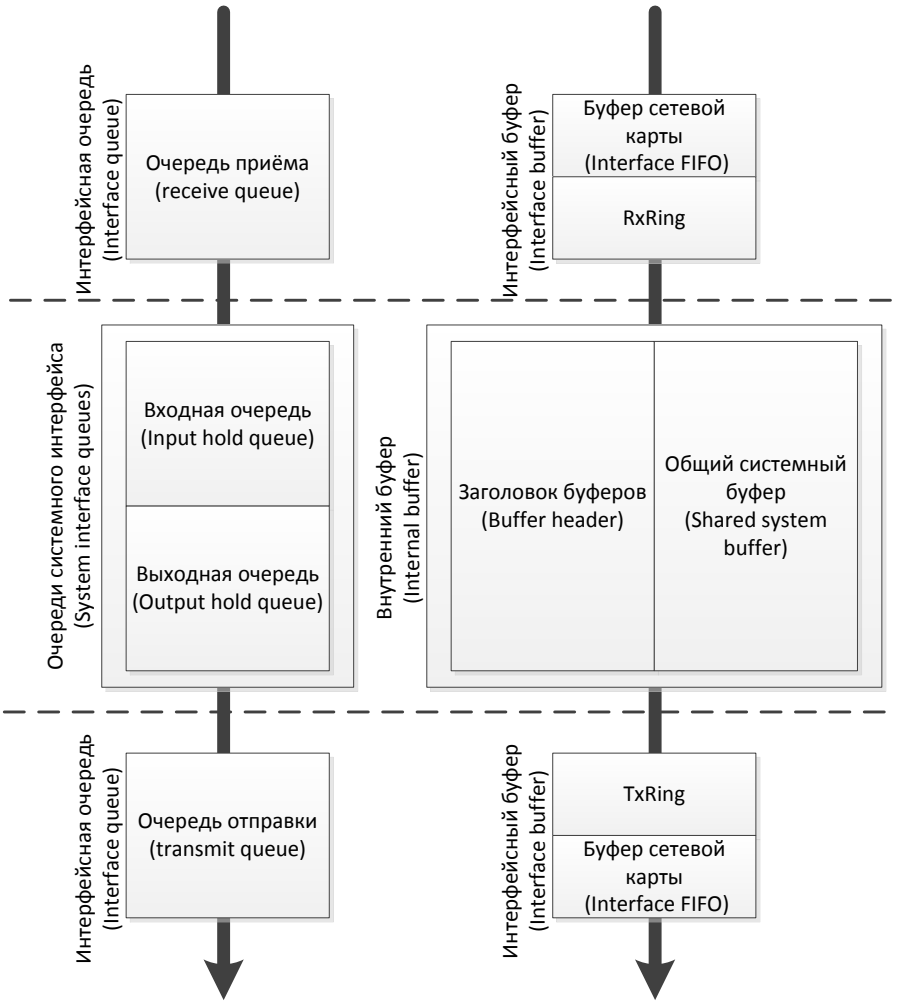


Рисунок 2.1 – Путь прохождения пакетов по очередям и буферам памяти

Конкретные решения по организации памяти для реализации очередей в рамках данной работы не рассматриваются по причине того, что большая часть

документации производителей закрыта. Вопрос организации достаточно важен с точки зрения производительности, однако, это не влияет на очерёдность выполнения проверок при обработке пакетов в МЭ.

2.1.4. Механизмы качества обслуживания пакетов и архитектура их применения в МЭ

Основная часть информации этого пункта взята из источников [72, 73].

Классификация и маркировка пакетов. Классификация пакетов используется для распознавания пакетов, принадлежащих разным классам трафика и их последующей маркировки. Предварительная классификация с присвоением метки является времязатратной операцией, поэтому её стараются производить на пограничном оборудовании сети один раз. Учитывая, что в сетях разных операторов и их клиентов метки могут не совпадать между собой и предоставляемыми классами обслуживания, на границе сети необходимо проводить повторную классификацию, либо договариваться с «соседом» о соответствии определённых меток и классов обслуживания.

Распознанные пакеты маркируются (помечаются). Оборудование внутри сети различает пакеты по установленным ранее меткам и соотносится с определённым классом обслуживания.

Классификация на 2-4 уровнях модели OSI (L2-L4) может производиться на основании битов полей priority (802.1p) [74] Ethernet-кадра, поля *differentiated services code point* (DSCP) [75, 76], IP-адресов и MAC-адресов источника и получателя, портов протоколов транспортного уровня. В ранней спецификации протокола IP поле DSCP было полем *type of service* (ToS).

Описанные в дальнейшем механизмы по характеру своей работы могут быть *классовыми* (classful) или *бесклассовыми* (classless). Классовый механизм (алгоритм) учитывает класс обслуживания (метку) трафика при обработке пакетов, а бесклассовый не учитывает.

Управление интенсивностью трафика. Управление интенсивностью трафика используется в целях выделения определённых полос пропускания для

определённых классов трафика (клиентов, служб и т.п.), а также для того, чтобы выравнивать броски трафика и не допускать потери от превышения допустимых значений пропускной способности на последующих узлах сети.

Для выполнения этого предусмотрено две функции. Первая – *«жёсткое» ограничение полосы пропускания* (англ. policing, далее – ограничение полосы пропускания). Вторая – *«мягкое» ограничение полосы пропускания* (англ. shaping, далее – выравнивание полосы пропускания), также его называют сглаживанием полосы пропускания

Механизм ограничения полосы пропускания выполняет сброс всех пакетов, которые превысили отведённую полосу пропускания. Механизм выравнивания полосы пропускания выполняет буферизацию пакетов, вышедших за границы полосы пропускания, и начинает сбрасывать пакеты, только если дополнительные квоты передачи трафика израсходованы. Оба механизма чаще всего реализуются на основе алгоритма называемого *корзина маркеров* (англ. token bucket).

Распределение ресурсов оборудования. Под распределением ресурсов понимается процесс выделения некоторого объёма ресурсов устройства определённым классам трафика. Под ресурсами понимается доступная полоса пропускания, а также комплекс вычислительных ресурсов и алгоритмов, выполняющих обслуживание пакета внутри устройства. В некоторых источниках механизм распределения ресурсов также называют *механизмом управления перегрузками* (англ. congestion management). Напомним, что в рамках работы он называется механизмом приоритизации обслуживания (отправки) пакетов.

Основой механизмов приоритизации обслуживания является *планировщик* (англ. scheduler) очередей. Планировщик по определённому алгоритму вынимает пакеты из очередей и отправляет на обслуживание или на выходную сетевую карту. Планировщик поддерживает различные дисциплины обслуживания. Эти дисциплины обслуживания могут учитывать или не учитывать классы пакетов, их размер, количество пакетов одного класса, пропущенного за единицу времени и другие характеристики.

Наиболее простой и распространённой дисциплиной обслуживания является «*первый пришёл, первым обслужен*» (англ. first-in, first-out – FIFO). Существуют также множество других дисциплин обслуживания и построенных на их основе механизмов обслуживания очередей [66, 72, 73, 77].

Предотвращение перегрузки и политика отбрасывания пакетов. Под *предотвращением перегрузок* (англ. congestion avoidance) понимается процесс сброса пакетов из очереди для понижения нагрузки на устройство. Бездумный сброс пакетов может приводить к дополнительным перегрузкам, которые появляются в результате лавинообразного эффекта *глобальной синхронизации* в TCP [78], что в большей степени характерно при использовании пассивных политик *отбрасывания хвоста* (англ. tail drop). Также подобный эффект может проявляться при использовании других протоколов с гарантированной доставкой и адаптацией полосы пропускания помимо TCP.

Предотвращение перегрузок выполняется путём сброса пакетов на основе правил *политики отбрасывания пакетов* (англ. packet discard policy). За сброс отвечают *механизмы управления очередями* (англ. buffer/queue management). Эти механизмы делятся на пассивные и активные. К пассивным типам относится уже упомянутый механизм отбрасывания «хвоста». Активных механизмов гораздо больше, например, *random early detection* (RED) и другие [79, 80].

Активные механизмы характеризуются тем, что начинают сбрасывать пакеты не в момент полного переполнения, а постепенно при преодолении граничных значений. Например, может отбрасываться каждый 10 пакет, при заполнении очереди свыше 70%, а свыше 80 % каждый 7 и т. д.

Архитектура применения механизмов качества обслуживания в оборудовании сетей связи. Механизмы качества обслуживания, описанные ранее, могут использоваться в комплексе или по отдельности. При этом стоит различать применение механизмов *QoS на входе устройства* (англ. ingress QoS) и на *выходе устройства* (англ. egress QoS). Сами по себе механизмы QoS, применяемые на входе и выходе устройства, никак не различаются функционированию, однако, цель и перечень применяемых механизмов различается. Как и с программными

очередями, ingress QoS применяется до основной части процесса обслуживания (коммутации, маршрутизации, фильтрации и т.п.), а механизмы egress QoS – после процесса обслуживания и до отправки.

Перечень механизмов QoS, применяемых на входе и выходе устройства, зачастую зависит от: типа устройства (коммутатор, маршрутизатор, межсетевой экран и т.п.), класса устройства и места применения (доступ, агрегация, ядро сети и т.п.). Также у различных производителей свой взгляд на необходимый и достаточный набор механизмов QoS.

В аппаратных очередях (сетевых картах) механизмы качества обслуживания не реализуются. Это обусловлено тем, что главная цель аппаратной очереди состоит в том, чтобы быстро и безошибочно записать пакеты, приходящие на интерфейс. Целью программной очереди является относительно долгое хранение пакетов до и после их обработки (маршрутизации, коммутации, фильтрации).

Обработка пакетов с приоритетом обслуживания требует буферизации пакетов, что автоматически связывает механизмы ingress QoS с входной очередью, а egress QoS – с выходной очередью, относящимся к очередям системного интерфейса. Очереди, использующиеся для реализации механизмов QoS, будем называть QoS-очередями. В некоторых реализациях оборудования QoS-очередь может быть отделена от очереди системного интерфейса в отдельную структуру данных. Как правило, в таких случаях между этими структурами данных передаётся только ссылка на пакет, а сам пакет находится недвижимо внутри памяти устройства.

Классификация и пометка пакетов применяется на ступени ingress QoS до помещения пакетов в QoS-очередь в том случае, если трафик не помечен или оператор устройства «не доверяет» полученным меткам QoS и хочет проверить/заменить их. Также вторичная классификация может понадобиться, если к одному и тому же классу обслуживания в разных сетях соответствуют разные метки, то есть их надо заменить. Многие МЭ, которые обладают возможностью анализа полезной нагрузки пакетов, способны выполнять более точную классификацию пакетов на основе этой информации. Подобный

функционал применяется на ступени обслуживания пакетов между ingress и egress QoS (рисунок 2.2).

Задержку, вносимую процессом классификации, можно считать постоянной и зависящей от того, информация какого уровня используется для классификации (влияет фактор распаковки и время на анализ информации).

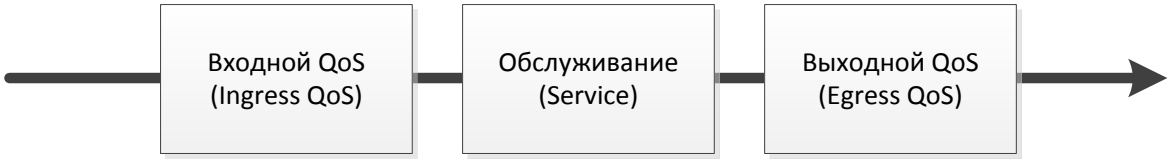


Рисунок 2.2 – Расположение групп механизмов QoS

Оба механизма управления интенсивностью трафика: ограничение и выравнивание полосы пропускания могут применяться как на ступени ingress QoS, так и egress QoS. Более стандартным решением является использование ограничения полосы пропускания на этапе ingress QoS и выравнивание полосы пропускания на этапе egress QoS. В этом случае на входе МЭ трафик обрезается до оговоренных значений и не перегружает устройство, а на выходе уже обслуженный трафик выравнивается для создания равномерного потока (при высокой загрузке). Использование выравнивания полосы пропускания на этапе ingress QoS даёт положительный результат при обслуживании большого числа «долго живущих» соединений с разными классами обслуживания, также это позволяет терять меньше пакетов при случайных бросках трафика. Однако механизмы выравнивания полосы пропускания имеют недостатки.

Основным минусом выравнивания полосы пропускания является его ресурсозатратность. Работа механизма требует буферизации пакетов (памяти). К тому же обязательным условием будет активация планировщика, который будет решать, в какой последовательности обрабатывать задержанные и текущие пакеты. В противовес этому ограничение полосы пропускания является более простым механизмом, не требующим лишней буферизации и хорошо реализуемым аппаратно, что обуславливает его применение.

Распределение ресурсов (управление перегрузками), а также предотвращение перегрузок (отбрасывание пакетов) – механизмы неразрывно

связанные с очередями. Если нет очереди, то невозможно обеспечить приоритизацию обслуживания или алгоритмы сброса пакетов. Условно можно считать, что механизм управления перегрузками представлен дисциплиной обслуживания пакетов, реализованной в планировщике, а механизм предотвращения перегрузок представлен алгоритмом сброса пакетов. В этом случае оба механизма являются дополнительными элементами очереди.

При необходимости введения приоритета обслуживания или сложных алгоритмов сброса пакетов обычно новых структур типа очередь не создают, а усложняют уже функционирующие. Дополнительная перезапись пакетов в новые структуры типа очередь является затратной по времени и отрицательно сказывается на скорости обслуживания, поэтому такой подход редко применяется. Для реализации описанных механизмов создаются дополнительные логические структуры, которые хранят информацию только об усложнённой структуре очереди. Такие структуры обычно хранят только ссылки на пакеты, а сами пакеты остаются недвижимы в рамках памяти (если не требуется передача с одной интерфейсной карты на другую). Зачастую в таких логических структурах переносят заголовки пакетов, а полезная нагрузка храниться отдельно.

Возможность приоритизации обслуживания во входных очередях обычно предусмотрена в коммутаторах, маршрутизаторах и МЭ классов service provider и data center.

Реализация описанных в настоящем пункте комплекса механизмов качества обслуживания достаточно разнообразна. Стоит уточнить, что производители предусматривают возможность использования различных комбинаций механизмов QoS для различных типов устройств их классов и мест применения. Наиболее часто встречающийся набор функций качества обслуживания для пограничного оборудования L3+ представлен на рисунке 2.3.

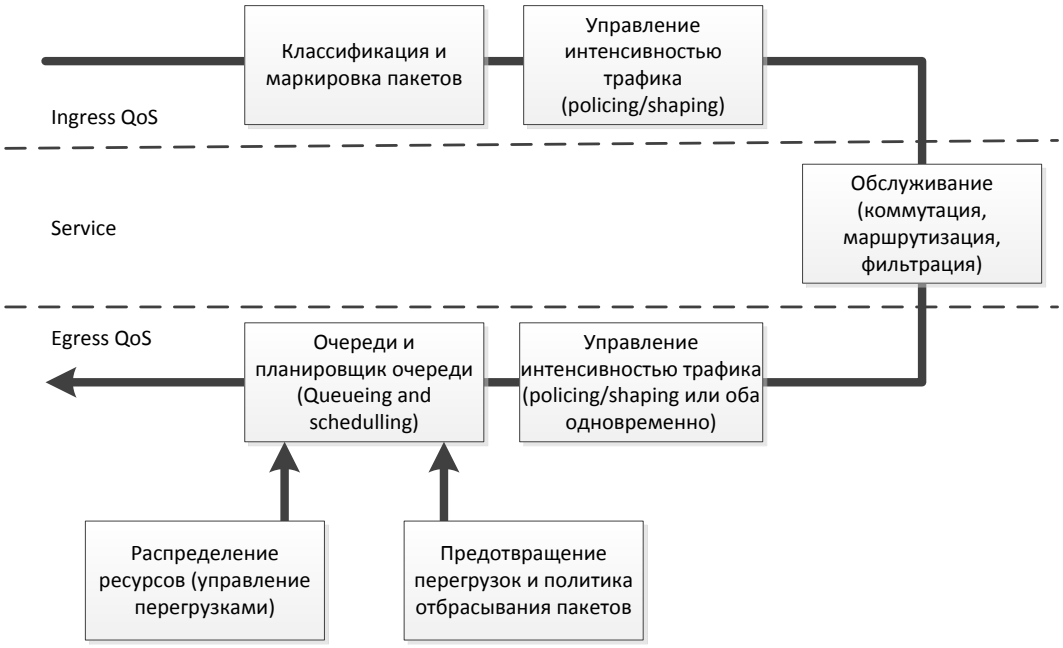


Рисунок 2.3 – Типовой предусмотренный набор механизмов QoS для пограничного оборудования сетей передачи данных

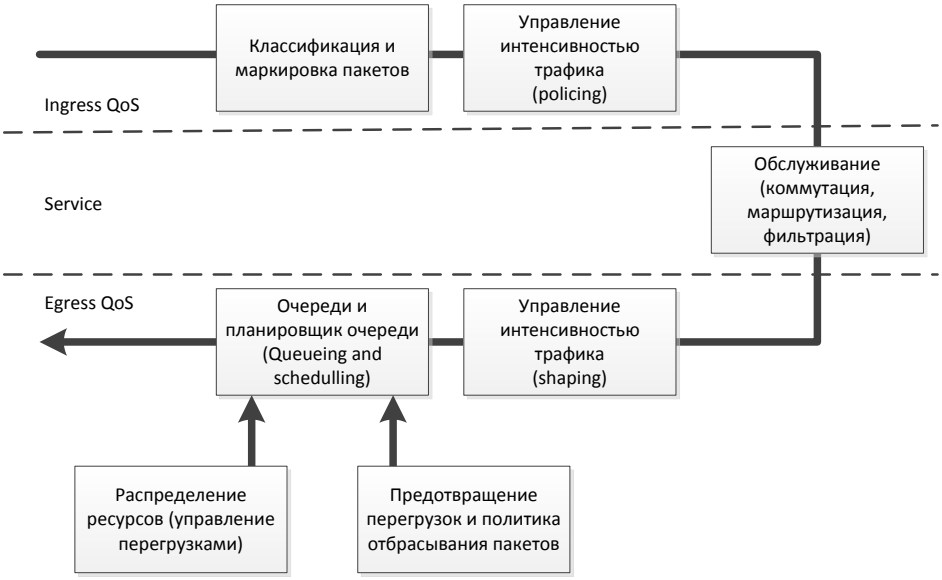


Рисунок 2.4 – Типовой использующийся набор механизмов QoS для пограничного оборудования сети провайдера услуг связи

Из предусмотренного набора механизмов обычно используется ограниченный набор, зависящий от места установки оборудования. Например, оборудование, установленное на *границе провайдера* (англ. providers edge), обычно должно поддерживать цепочку механизмов качества обслуживания, представленную на рисунке 2.4 [73].

В ядре сети провайдера механизмы управления интенсивностью трафика типа policer, shaper, как правило, не сбрасывают пакеты, а выполняют их перемаркировку в классы с меньшим приоритетом обслуживания. Механизмы распределения ресурсов и предотвращения перегрузок используются также как и на границе сети.

Из рисунков 2.3, 2.4 видно, что из предусмотренного набора используется только определённый перечень механизмов. Набор используемых функций на стороне клиента зависит от его потребностей и может, как отличаться от рассмотренного ранее для провайдеров услуг доступа, так и полностью совпадать с ним. На рис 2.5 представлен типовой набор механизмов QoS для оборудования, поддерживающего функционал приоритизации обслуживания в ingress QoS.

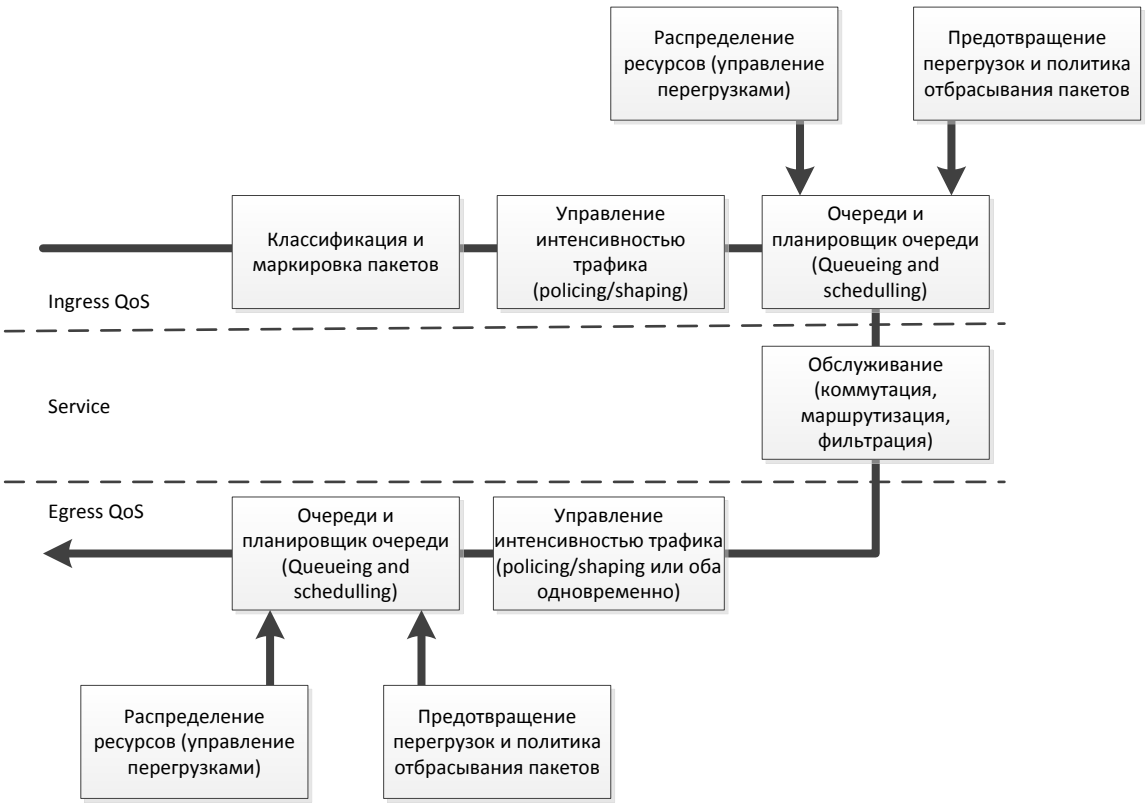


Рисунок 2.5 – Типовой набор механизмов QoS для оборудования, поддерживающего функционал приоритизации обслуживания ingress QoS

2.1.5. Фильтрация пакетов с учётом и без учёта состояния соединения, а также технология анализа информации прикладного уровня

Основная часть информации, содержащаяся в этом пункте, взята из [81-83].

Фильтрация трафика является основной функцией МЭ. Существует две основных технологии фильтрации трафика на L2-L4: с учётом состояния и без учёта состояния соединения. Отдельно стоит технология анализа информации протоколов прикладного уровня и полезной нагрузки пакетов.

Исторически первой была фильтрация без учёта состояния соединения, ещё её называют попакетной фильтрацией. При такой фильтрации проверяется каждый пакет, но МЭ не обладает возможностью ассоциировать пакеты с какими-либо соединениями, не различает пакеты, открывающие соединения от пакетов с полезной нагрузкой и т.п.

При фильтрации с учётом состояния соединения МЭ способен различать отдельные сеансы связи транспортного уровня и открытые в рамках них соединения, статус пакетов в рамках этих соединений. Соединения различаются друг от друга по адресам сетевого уровня, портам транспортных протоколов и статусу соединения. Данная технология особенно эффективна при работе с протоколами, ориентированными на установление соединения.

Для ускорения работы оборудования используется концепция создания двух путей обслуживания, называемых: *первичный путь* (first path) и *быстрый путь* (fast path). Идея данной концепции в том, что пакет, открывающий соединение, проходит через первичный путь и подвергается полному объёму проверок безопасности (L2-L4), а также маршрутизации и трансляции сетевых адресов. Все пакеты, отнесённые к уже открытым (разрешённым) соединениям, будут направляться по быстрому пути, где они подвергаются минимальному набору проверок из первичного пути, а также дополнительным проверкам, которые в быстром пути не проводятся.

Сравнение технологий фильтрации с учётом состояния и без учёта состояния соединения приводит к следующим выводам. Технология фильтрации без учёта состояния соединения функционирует быстрее, потенциально проще реализуема аппаратно, но не предоставляет достаточного уровня безопасности. Технология фильтрации с учётом состояния соединения функционирует несколько медленнее, её функции сложнее реализовать аппаратно, но она

предоставляет более высокий уровень безопасности. Это привело к тому, что во многих современных МЭ применяется комбинация данных решений. На входе МЭ первоначально можно обеспечить «грубую» фильтрацию трафика без учёта состояния, что позволит отсеять часть пакетов, тем самым снизив нагрузку на более затратную по времени функцию, учитывающую состояние соединения.

Технология анализа информации протоколов прикладного уровня и полезной нагрузки появилась сравнительно недавно. В свою очередь её обычно разделяют на фильтрацию протоколов и сервисов прикладного уровня на основе их служебной информации и на фильтрацию полезной нагрузки. Большая часть современных МЭ поддерживают данный комплекс технологий в большей или меньшей степени. Процесс анализа информации прикладного уровня ресурсозатратен для устройства, требует значительного времени и приводит к большим задержкам при анализе пакетов. Подобные операции обычно выделены в отдельную, независимую ступень обслуживания. В рамках настоящей работы не рассматриваются технологии анализа информации прикладного уровня.

2.1.6. Частные случаи схем обработки пакетов в МЭ

В диссертации проанализировано несколько частных реализаций МЭ на предмет проверки процесса обслуживания пакетов в них. Выбраны распространённые решения в сегментах Branch, Enterprise.

Уточним, что впоследствии для реализации модели был выбран МЭ NetFilter/IPTables. Для упрощения восприятия материала и вследствие ограничения объёма работы подробное описание процессов обработки пакетов для выбранных далее решений перенесено в приложение А.

Cisco systems. К рассмотрению приняты линейки оборудования Cisco ASA 5500. Более продвинутые межсетевые экраны линеек Cisco Firepower 2100, 4100, 9000 выходят за границу Enterprise-решений, также по их функционированию не удалось найти достаточно информации, ввиду их новизны.

Анализ проводился по источникам [81, 85-91]. Условно разделим линейку Cisco ASA 5500 на два поколения: обычные Cisco ASA – предыдущее поколение и

Cisco ASA with FirePOWER – новое поколение. Функционирование Cisco ASA предыдущего поколения описано в книге [81], а также информация с официального сайта производителя для версии ПО 8.2, актуальной на 2011 г. [85]. Дополнительная информация получена из неофициальных источников для версии ПО 8.1, актуальной на 2008 г. [86, 87]. По информации источников [81, 85] становится ясно, что значительное изменение процесса обслуживания между версиями 8.1, 8.2 не наблюдается. Версии ПО для Cisco ASA описаны в [88]. Критических изменений архитектуры с версии Cisco ASA 8.2 до 9.2, актуальной на 2017 год, не зафиксировано. Функционирование Cisco ASA with Firepower описано в [89, 90]. Принципы работы механизмов QoS в Cisco ASA версий ПО 9.2 и более ранних описано в статье [91]

Схемы, представленные в источниках [80, 83], сильно упрощены, информации, представленной в них, недостаточно. На основе схем, представленных в [79, 83], а также дополнительной информации из [85-91] сформирована схема процесса обслуживания пакетов для Cisco ASA (рисунок А.1), представленная в приложении А.

Juniper networks. К рассмотрению приняты межсетевые экраны серии Juniper SRX. Модельный ряд продукта находится в сегментах Branch, Enterprise, Datacenter, Service provider, однако, принципы обслуживания пакетов одинаковые в оборудовании для разных сегментов рынка.

Описание Juniper SRX и принципов её функционирования сделано по [82, 92-94]. Схема обработки пакетов версии ПО 12.1R1.9, актуальная на 2016 г.

Схемы обработки пакетов, представленные в [80, 90], являются специализированными для объяснения функционирования устройства в различных ситуациях и при обработке пакета различными операциями. Учитывая вышесказанное, была сформирована обобщённая схема обработки пакета (рисунок А.2), представленная в приложение А.2.

Huawei Technologies. К рассмотрению приняты межсетевые экраны серий Huawei USG 6000, находящиеся как в сегменте Branch, так и Enterprise.

Описание линейки Huawei USG6000 и принципов её функционирование сделано по [95, 96]. Схемы обработки пакетов, представленные в [95], специализированы под определённые сценарии обработки пакетов. Сформирована обобщённая схема на рисунке А.3 в приложение А.

IPTables/NetFilter. Решения на основе свободно распространяемого ПО достаточно популярны вследствие того, что необходимо заплатить только за аппаратную платформу МЭ и в некоторых случаях за дистрибутив ОС, если он платный. МЭ с открытым исходным кодом довольно много, например, NetFilter/IPTables, PF, IPFILTER, IPFW и т.д. Эти продукты предназначены для использования на UNIX-подобных ОС, таких как Linux, OpenBSD, FreeBSD, SunOS и т.д. Процесс обслуживания пакетов в этих МЭ в достаточной степени различается. К рассмотрению примем только NetFilter/IPTables вследствие того, что имеется большее число работ по анализу функционирования этого МЭ, что снижает вероятность совершения ошибочных выводов при анализе.

Говоря о МЭ IPTables/NetFilter, надо понимать, что основной задачей этого программного пакета является фильтрация трафика, хотя он сам по себе поддерживает ряд механизмов QoS в упрощённой форме. Основные механизмы QoS встроены в большинство современных версий ядра Linux, но не используются без дополнительных программных пакетов, таких как iproute2 и др.

Каждый раз, говоря о МЭ IPTables/Netfilter, будем подразумевать программные пакеты IPTables/NetFilter, iproute2 и некую ОС семейства Linux, в среде которой они функционируют.

Описание МЭ IPTables/NetFilter и его принципов функционирования сделано по [84, 97-106].

Схемы обработки пакетов представлены в источниках [83, 105]. Схемы несколько различаются, использовались более новые. При составлении адаптированной под нужды работы схемы обработки пакетов (рисунок 2.6) использовалась информация из источников [83, 105]. Подробное описание схемы обработки пакетов представлено в приложении А.

Общая концепция функционирования механизмов качества обслуживания совпадает с той, которая рассматривалась в пункте 2.1.4, однако, существуют отличия по другим аспектам, рассмотренные далее.

Операции 1 и 26 выполняются на сетевых картах (сетевых контроллерах). Операции 2-25 выполняются ресурсами МЭ.

В IPTables/NetFilter нет ярко выраженной концепции первого и быстрого путей. Существует возможность создавать правила на основе состояния соединения, что, в свою очередь, позволяет пропускать пакеты, которые передаются в рамках уже разрешённого сеанса связи в обход ряда проверок. Допускается создание более сложных сочетания правил на основе этого.

Очерёдность обслуживания пакета с точки зрения движения пакета между аппаратными и программными очередями может различаться от реализации конкретной версии и комплектации ядра Linux. Также могут различаться принципы и объёмы выделения памяти, переноса пакетов из одних участков памяти в другие и т.п. Работа физических очередей сетевых карт зависит от их модели и будет привязана к конкретной аппаратной реализации.

Приоритет обслуживания трафика может быть учтён в группах операций 3 и 25, а также при принятии решений о маршрутизации и фильтрации при использовании внутренней маркировки пакетов IPTables/NetFilter. В группах операций 3 и 25 перечень применяемых механизмов качества обслуживания является одинаковым, то есть можно производить ограничение полосы пропускания, сглаживание трафика, приоритизацию постановки на обслуживания/отправки в сеть, а также активное управление очередью. На практике используются только необходимые механизмы качества обслуживания.

Напрямую механизмы приоритизации постановки на обслуживание/отправку в сеть и выравнивания полосы не доступны на входе IPTables/NetFilter, но многие системные инженеры используют возможность создания виртуального выходного интерфейса с этими функциями и выполняют переадресацию трафика с входа на этот интерфейс и обратно. Тем самым на входе можно выполнить выравнивание полосы пропускания и приоритизацию

обслуживания трафика. Эту возможность стоит считать предусмотренным комплексом механизмов ingress QoS.

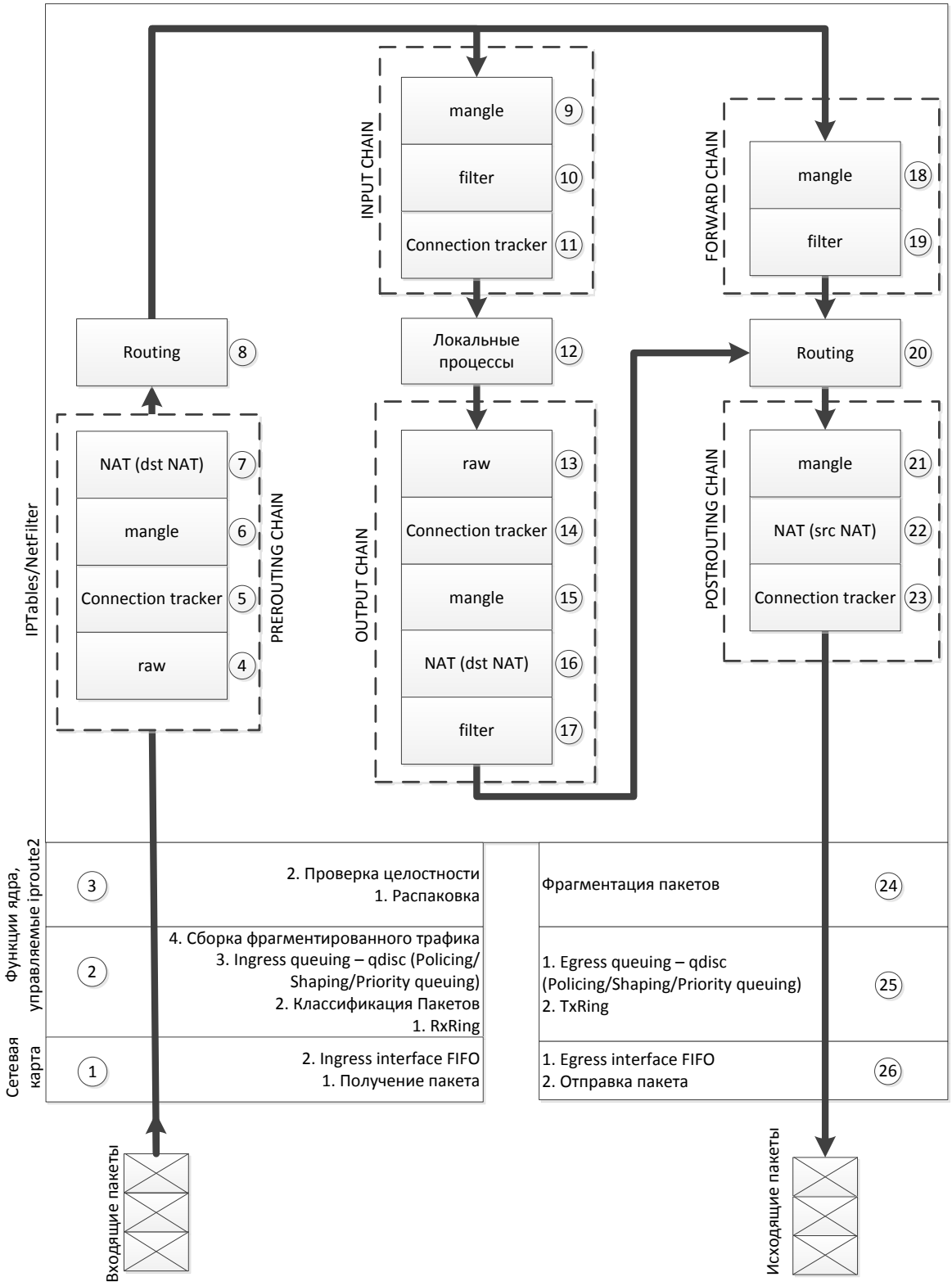


Рисунок 2.6 – Схема операций обработки пакетов в IPTables/NetFilter

Перечень возможных дисциплин обслуживания пакетов QoS-очереди несколько отличаются от рассмотренных ранее МЭ. Доступные алгоритмы

активного управления очередью: tail drop, RED. Для сглаживания трафика доступны алгоритмы: *token bucket filter* (TBF), *hierarchy token bucket* (HTB). Доступные алгоритмы управления порядком обслуживания пакетов: FIFO, PFIFO_FAST, *class based queuing* (CBQ), HTB, *PRIOR* (priority queuing).

Вносящими динамические задержки стоит считать операции 2, 3, 9, 10, 12, 17, 18, 19, 25. Можно допустить, что остальные операции вносят постоянную задержку, а также что при малом количестве правил в операциях с динамической задержкой можно считать вносимую задержку постоянной величиной. Для NetFilter/IPTables определение операций, вносящих постоянные и динамические задержки, достаточно сложно. Это обусловлено тем, что распределение правил по таблицам может иметь заметно более хаотичный характер, нежели чем у МЭ рассмотренных ранее, к тому же, не стоит забывать о возможности создавать вложенные таблицы.

2.1.7. Результаты рассмотрения архитектуры обработки пакетов в рассматриваемых межсетевых экранах

Схемы организации обработки пакетов. Концепция первого и быстрого путей присутствует в явном виде во всех рассмотренных МЭ, кроме МЭ NetFilter/IPTables, в котором присутствует возможность воспроизвести механизм функционирования данной концепции.

В общих чертах архитектура обработки пакетов у всех устройств схожа, однако, присутствует большое число различий в наличии/отсутствии определённых операций обработки, их расположении и очерёдности в рамках общей схемы обработки пакетов. Это объясняется всё тем же фактором «эволюции» этих устройств. Сходство объясняется однотипностью назначения устройств и стандартизацией протоколов, способов функционирования оборудования и их аппаратного обеспечения.

Функционирование аппаратных и программных очередей. Информации, раскрывающей особенности функционирования очередей в рассматриваемых устройствах, для сравнения недостаточно. Вероятно, различий в общей

концепции работы очередей нет, так как все рассматриваемые МЭ (Cisco, Juniper, Huawei, IPTables/NetFilter) работают на основе UNIX-подобных ОС, а способов быстрого захвата и сохранения пакетов не так много. Уточним, что предыдущее утверждение верно только для оборудования в рамках конкретного класса устройств (Branch, Enterprise). Между классами могут существовать значительные различия вследствие упрощения или усложнения принципов функционирования (удешевления производства, понижение/повышения производительности).

Механизмы качества обслуживания. Рассмотренное оборудование Juniper SRX и Cisco ASA имеет предусмотренный набор механизмов качества обслуживания с приоритизацией обслуживания в egress QoS представленный ранее на рисунке 2.3. Оборудование Huawei USG 6000 не имеет предусмотренных механизмов качества обслуживания за исключением ограничения полосы пропускания, однако, старшая модель продуктовой линейки USG 9500 имеет предусмотренный набор механизмов качества обслуживания, функционирующие как во входной, так и выходной очередях МЭ (рисунок 2.5). Также предусмотренный набор механизмов качества обслуживания трафика во входных и выходных очередях МЭ доступен в IPTables/NetFilter (рисунок 2.5).

В оборудовании Huawei USG 6000 несколько усложнены механизмы управления полосой пропускания по сравнению с другими продуктами.

Дисциплины обслуживания. Разработчики всех МЭ используют либо «удачные» общедоступные алгоритмы, реализующие определённые дисциплины обслуживания, или разрабатывают собственные (проприетарные) алгоритмы для уже известных дисциплин обслуживания. Во всех рассмотренных МЭ реализована дисциплина обслуживания FIFO, то есть дисциплина без приоритета. Предусмотренные дисциплины обслуживания с приоритетами реализованы достаточно разнообразным комплексом алгоритмов. Схожая дисциплина обслуживания реализована в алгоритмах: PRIOR (IPTables/NetFilter), LLQ (Cisco ASA), stick-priority queue (Juniper SRX). Ряд рассмотренных устройств поддерживает и другие алгоритмы обслуживания пакетов с приоритетами, но эти алгоритмы имеют различия.

Все рассмотренные МЭ поддерживают дисциплину сброса пакетов tail drop. Второй по распространённости алгоритм RED и его модификация WRED, который используется только в оборудовании Cisco и является проприетарным.

Краткие выводы. Архитектура обработки пакетов у всех производителей схожа, но имеется ряд значительных различий в составе и количестве операций и проверок, которые выполняются в рамках обслуживания пакета.

В части рассмотренных схем не предусмотрена возможность приоритизации обслуживания пакетов во входных очередях. Из рассмотренных МЭ только IPTables/NetFilter обладает механизмами приоритизации обслуживания во входной очереди. Возвращаясь к приоритизации обслуживания трафика во входной и выходной очередях устройства следует заметить, что если скорость отправки пакетов выше, чем скорость их обслуживания, то выходная очередь не заполняется и, как следствие, положительный эффект на показатели качества обслуживания от приоритизации обслуживания трафика в ней невелик.

Выходная очередь может заполняться, если её выходной интерфейс в значительной степени загружен, например, принятием пакетов, а также при условии концентрации трафика с нескольких интерфейсов и скорости обслуживания выше скорости интерфейсов. Напомним, что такой класс устройств как МЭ значительно дольше обрабатывает пакеты, нежели маршрутизаторы или коммутаторы. Стоит считать, что приоритизация обслуживания во входной очереди является достаточно эффективным средством ускорения обслуживания критических к задержкам пакетов в МЭ при условии загрузки входной очереди.

В ряде случаев дисциплины обслуживания с приоритетами и дисциплины сброса пакетов совпадают полностью или близки по функционированию.

Описанные выводы показывают необходимость исследования влияния приоритизации обслуживания трафика во входных очередях на производительность МЭ, как оборудования фильтрации трафика.

2.2. Способы оценки производительности межсетевых экранов

2.2.1. Характеристики качества обслуживания трафика межсетевыми экранами

Основная часть информации, представленная в этом пункте, взята из источников [72, 73].

Полоса пропускания. Под термином *полоса пропускания* (англ. bandwidth) понимается номинальная пропускная способность среды передачи информации, оборудования, протокола, соединения или комплекса этих и других объектов.

Задержка при передаче пакета. Под термином *задержка при передаче пакета* (англ. packet delay, latency) понимается сумма временных задержек: задержки сериализации, задержки распространения, задержки обслуживания.

При измерении задержек сети обычно оперируют следующими величинами: *односторонней задержкой пакетов* (англ. one-way delay metric) [107] и *временем оборота пакетов* (англ. round trip time) [108].

Задержка сериализации **(serialization delay)** – время, которое требуется оборудованию сети для передачи пакета при заданной полосе пропускания. Задержка сериализации зависит от величины полосы пропускания и размера передаваемого пакета. В некоторых источниках задержку сериализации также называют задержкой передачи (англ. transmission delay). Задержка сериализации заметна на «медленных» каналах связи и почти незначительна при «быстрых».

Задержка распространения **(англ. propagation delay)** – время, которое требуется пакету для прохождения между передающим и принимающим оборудованием по какой-либо среде передачи. Скорость движения пакета в разнородных средах передачи, используемых в современных сетях связи, соизмерима со скоростью света (несколько различается в зависимости от среды передачи). При передаче пакета на значительные расстояния задержка распространения становится достаточна заметна.

Задержка обслуживания **(англ. service delay)** – время, которое требуется оборудованию, получившему пакет, для его передачи следующему узлу сети. Чем

больше операций оборудование выполняет над пакетом: коммутация, маршрутизации, фильтрация и т.п., тем значительней задержка. Меньше всего задержки обслуживания вносят функции коммутации, затем маршрутизации трафика, следом идут функции обеспечения безопасности, в то числе фильтрация, шифрование трафика и др. Также к задержкам обслуживания относится время задержки пакета в очередях.

Вариация задержки **(англ. jitter)** – величина отклонения задержки каждого отдельного пакета от средней величины задержки [109].

Потери пакетов. Под термином потеря пакетов (англ. packet loss) понимается количество пакетов, отброшенных сетью связи (комплексом оборудования) во время их передачи. Основными причинами потери пакетов являются перегрузки сети и повреждения пакетов во время передачи.

Рассматриваемые характеристики. В рамках данной работы основной интерес представляют полоса пропускания МЭ, задержка обслуживания, потеря пакетов. Хотя задержка сериализации напрямую относится к оборудованию, она незначительна на высокосортных каналах и ей можно пренебречь.

2.2.2. Математическая модель функционирования межсетевого экрана без приоритизации обслуживания трафика

Наибольшее число различных математических моделей МЭ формируется для IPTables/NetFilter вследствие того, что его документация полностью открыта.

Одна из таких моделей представлена в работе [20] за авторством специалистов IEEE Khaled Salah, Khalid Elbadawi, Raouf Boutaba. Работа посвящена обоснованию уязвимости существующих механизмов списков фильтрации трафика, применяемых в оборудовании, выполняющем межсетевое экранирование. Обоснование было выполнено с использованием разработанного авторами [20] математического аппарата, представляющего практический интерес. Рассмотрим математическую модель специалистов IEEE.

Математическая модель реализует процесс обслуживания пакетов, принятый в МЭ IPTables/NetFilter, применяемый в ОС Linux. Если сравнивать

принятый там процесс обслуживания пакетов с МЭ других производителей, то можно обнаружить значительные отличия. Описание различных концепций обслуживания пакетов МЭ представлено в подразделе 2.1.

Основываясь на технической документации [94] специалисты IEEE сформировали аналитическую модель (рисунок 2.7).

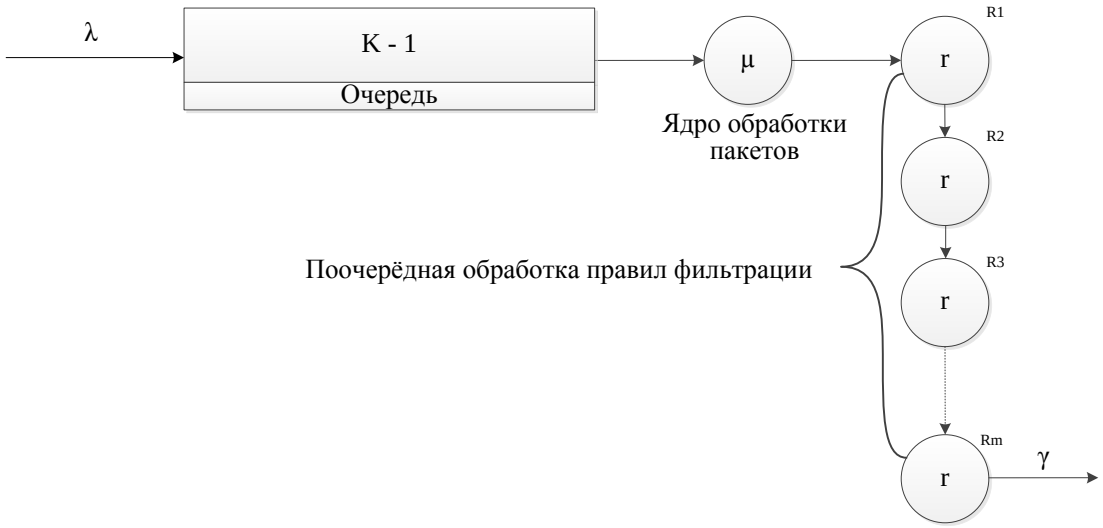


Рисунок 2.7 – Аналитическая модель функционирования МЭ из работы [20]

Функционирование МЭ представлено в виде модели массового обслуживания и проанализировано специалистами IEEE с использованием математического аппарата Марковских процессов.

Авторы [20] получили значения интенсивности обслуженного потока пакетов, вероятности отказа (потери пакетов), среднего число пакетов в системе, среднего времени нахождения пакета в очереди, среднего времени пребывания пакета в системе.

2.2.3. Возможности использования существующей модели

Текущая модель не позволяет провести исследования производительности МЭ, функционирующих в условиях приоритизации обслуживания трафика. Наработки, полученные специалистами IEEE, возможно адаптировать и использовать при разработке новой математической модели. Первым шагом к созданию новой математической модели является формирование содержательной постановки задачи на разработку математической модели.

Для формирования содержательной постановки задачи проанализированы:

– работа механизмов качества обслуживания (приоритизации обслуживания трафика);

– различные подходы к организации процессов обслуживания пакетов МЭ;

– принципы работы очередей и распределения памяти в МЭ.

Необходимо выбрать определённый МЭ и его процесс обслуживания пакетов с учётом работы механизмов качества обслуживания, а также сформулировать принятые допущения и ограничения по использованию модели.

2.3. Постановка задачи на разработку математической модели

2.3.1. Выбор модели процесса обслуживания

Рассмотренные ранее схемы организации обработки пакетов различаются в достаточной мере, чтобы говорить о невозможности разработки универсальной математической модели для оценки параметров функционирования МЭ. Каждая из рассмотренных схем обработки пакетов достаточно сложна для того, чтобы реализовывать её полностью, как единую математическую модель, использующую в основе математический аппарат теории вероятности. Вследствие этого необходимо сформировать либо ряд моделей, передающих результаты своей работы друг в друга, либо сформировать упрощённую математическую модель, содержащую ряд допущений.

Учитывая тот факт, что уже имеется в значительной степени проработанная модель, представленная в [20], имеет смысл за основу модели взять МЭ IPTables/NetFilter. Также МЭ с открытым исходным кодом обладают значительно большей документальной базой.

Будем считать, что МЭ исполняет роль пограничного устройства между двумя сетями, функционирующими на основе технологии Ethernet на канальном уровне и IP – на сетевом уровне.

Для упрощения модели рассматривается не полный цикл приёма-передачи пакета, а комплекс операций от получения пакета до фильтрации включительно. Будем считать, что МЭ анализирует проходящий

сквозь неё трафик, а трафика, приходящего на сам МЭ нет. Для упрощения рассматривается фильтрация без учёта состояния соединения.

Комплекс рассматриваемых функций качества обслуживания. Классификация и маркировка пакетов на входе МЭ выполняется по одним и тем же признакам во всём рассмотренном оборудовании. Условно считаем, что МЭ выполняет классификацию трафика, полученного от ЛВС, а меткам, полученным от пограничного оборудования оператора, доверяет. Сам процесс классификации представляет собой операцию с малой задержкой, которую можно принять за постоянную величину или не учитывать. Для прощения не будем её учитывать.

Управление интенсивностью трафика на входе МЭ не рассматривается по двум причинам. Механизмы ограничения полосы пропускания не будут рассматриваться так, как в рамках модели всегда точно известно, какова интенсивность входящего потока пакетов. Эмулировать работу ограничителя можно управляя самой интенсивностью входящих потоков пакетов. Механизмы вырывания полосы пропускания в принципе редко используются на входе устройства и обычно предназначены для вырывания исходящего потока, чтобы тот не был обрезан входными ограничителями полосы пропускания последующего устройства.

В рамках исследования наибольший интерес представляют механизмы приоритизации обслуживания пакетов. Как упоминалось ранее, обслуживание пакетов входными программными очередями по умолчанию подчиняется дисциплине FIFO. Эта дисциплина без приоритета обслуживания. Теория массового обслуживания предоставляет широкий ряд различный дисциплин обслуживания с приоритетами [66, 72, 73, 77]. Эти дисциплины обслуживания реализованы во многих алгоритмах и методах управления очередями. При составлении модели стоит использовать наиболее «опасный» метод управления очередями, который легко может привести к «оккупации» канала. Это позволит получить граничные значения производительности, на которые может рассчитывать неприоритетный трафик в различных ситуациях. Удачным вариантом являются методы управления очередью: strict-priority queuing (Juniper

SRX), LLQ (Cisco), PRIOR (NetFilter/IPTables). Фактически они предоставляют одинаковый функционал.

Классов трафика в сети, как правило, много, однако, придётся совершить упрощение и представить, что имеется всего два класса трафика приоритетный и неприоритетный. Приоритетный будет иметь абсолютный приоритет при постановке в очередь и относительный приоритет при постановке на обслуживание. Эти дисциплины реализуют выбранные ранее алгоритмы.

К качестве метода предотвращения перегрузок выбран tail drop. Другие методы, такие как RED, WRED и другие [79, 80], достаточно сложны для моделирования, а их реализация математическими методами зачастую невозможна. Укажем заранее, что перепосылка пакетов не рассматривается, можно считать, что перепосылки либо нет, либо пакеты, появившиеся в результате перепосылки, считаются частью общего потока и не различаются от породивших их пакетов ничем.

2.3.2. Содержательная постановка задачи

При составлении содержательной постановки задачи за основу принят МЭ типа NetFilter/IPtables на основе ОС Linux.

С точки зрения математики пакеты являются заявками на обслуживание.

В модели рассматривается два типа приоритетов. Первый – абсолютный приоритет постановки пакетов в очередь. Если очередь заполнена полностью, а на интерфейс пришёл пакет с более высоким приоритетом, чем находящийся в очереди, то он его вытеснит и займёт место в очереди в соответствии со своим приоритетом [72, 110]. Второй – относительный приоритет постановки на обслуживание. Это обуславливается тем, что обслуживание пакета не может быть остановлено или отменено с приходом более приоритетного пакета. В отличие от телефонных сетей связи, в которых результатом обслуживания заявки является продолжительный сеанс связи (разговор абонентов), в пакетных сетях является обслуженный пакет, время обслуживания которого настолько мало, что нет смысла прерывать его обслуживание.

На вход модели поступает два потока пакетов с интенсивностями поступления λ_1 для приоритетного потока и λ_2 для неприоритетного потока.

Полный объём модели равен $L+1$ пакетов. Один пакет находится на обслуживании, остальные пакеты находятся в очереди, вмещающей L пакетов.

Ядро обработки пакетов имеет среднее время обслуживания $1/\mu^*$. Оно включает в себя функции 3-8, 18, описанные в пункте «2.3.3 IPTables/NetFilter». Операции, выполняемые в «ядре обработки пакетов», применяются ко всем пакетам без исключения и в полном объёме. Если пакет был признан повреждённым при проверке контрольных сумм, то время его нахождения в системе будет меньше, но в модели будет пренебрегаться подобными случаями. Для упрощения задержки, вносимые операциями 3-8, 18, будут считаться одной комплексной задержкой.

Следующий этап – это проверка пакета по базе правил, а точнее таблицам filter. Так как в диссертации рассматривается проходящий сквозь МЭ трафик, то он будет проходить через таблицу 19 в цепочке FORWARD.

В результате имеем *K* этапов обслуживания. Каждое правило фильтрации представлено этапом обслуживания со временем сопоставления $1/\mu$. Правила фильтрации занимают от 2 до *K* этапа. В рамках одного цикла расчёта последним правилом будет являться правило на позиции *K*, то есть то, на котором произошло совпадение по критерию проверки пакета. Пакет, достигнувший правила *K*, покидает систему обслуживания. Не важно, сколько правил будет после правила *K* так как они не участвуют в обработке пакетов и не вносят задержки. Необходимо помнить, что *K* не максимальный объём базы правил, а математическое ожидание номера правила, на котором происходит совпадение. Модель не рассматривает задержки, вносимые после этапа фильтрации.

На практике $\mu << \mu^*$, так как время работы с заголовками транспортного, сетевого и канального уровней значительно больше, чем сличение полученных служебных параметров пакета с каждым отдельным правилом. Однако если значительно увеличить количество правил фильтрации, то время, затрачиваемое

на процесс фильтрации, становиться сравнимо со временем работы «ядра обработки пакетов» $\mu \cdot (K-1) \approx \mu^*$.

На рисунке 2.8 приведена модель функционирования МЭ в условиях приоритизации обслуживания трафика.

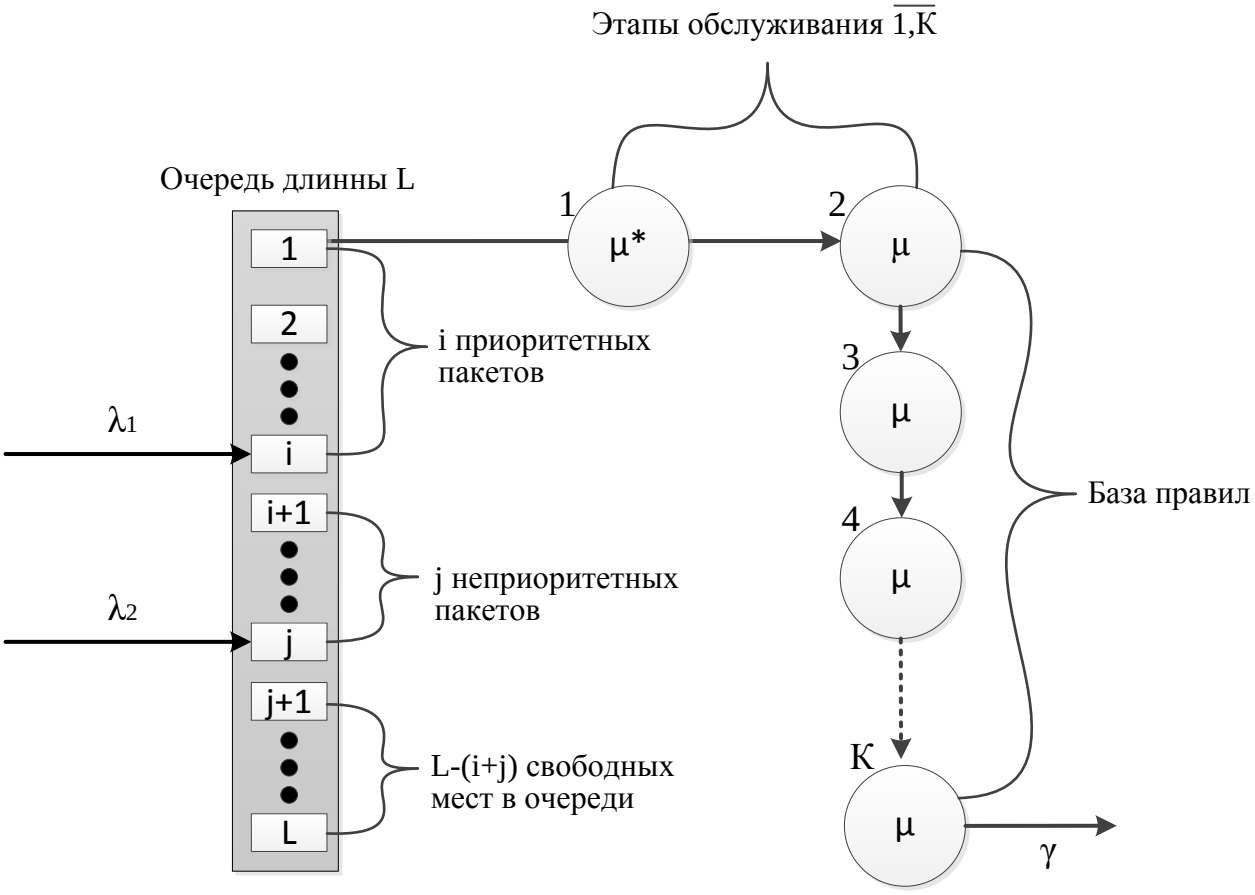


Рисунок 2.8 – Модель функционирования МЭ в условиях приоритизации обслуживания трафика

Допущение 1. МЭ подключен к сети связи, функционирующей на основе технологии Ethernet на канальном уровне и технологии IP на сетевом уровне.

Допущение 2. МЭ расположен на границе между сетями клиента и оператора доступа или на границе между сетями двух операторов доступа.

Допущение 3. Процесс функционирования сетевой карты, процесс переноса пакета из памяти сетевой карты в память МЭ и процесс переноса пакета между программами очередями МЭ не рассматривается, задержка этого процесса не учитывается. Считать, что сетевая карта имеет пропускную способность, заметно превышающую возможности МЭ.

Допущение 4. Рассматривается не полный цикл приёма-передачи пакета, а комплекс операций от получения пакета до фильтрации включительно. Рассматривается проходящий сквозь МЭ трафик, ассоциирующийся с таблицей Filtering в цепочке Forward.

Допущение 5. В модели пренебрегается задержкой, порождаемой при классификации пакетов. Это допущение влияет на наличие задержки перед постановкой пакетов в очередь. Условимся, что пакеты из внутренней сети классифицируются на МЭ, а выстроенная цепочка доверия (договоров о качестве услуг) позволяет оператору доверять установленным на МЭ меткам качества обслуживания, а МЭ доверять меткам, приходящим от операторской сети.

Допущение 6. В модели не рассматриваются механизмы управления полосой пропускания: ограничение и выравнивание полосы пропускания. Условно можно считать, что входящий поток и есть результат ограничения полосы пропускания.

Допущение 7. В модели рассматривается дисциплина обслуживания с приоритетом, реализуемая алгоритмами PRIOR (IPTables/NetFilter). Аналогичные алгоритмы у других рассмотренных разработчиков МЭ именуются LLQ (Cisco), stick-priority queuing (Juniper).

Допущение 8. В качестве метода предотвращения перегрузок рассматривается tail drop, вследствие сложности реализации других алгоритмов с использованием математических моделей.

Допущение 9. Рассматривается фильтрация без учёта состояния соединения.

Допущение 10. Входящих потоков всего два: с приоритетом обслуживания и без приоритета. Допущение было взято для уменьшения количества состояний системы и, как следствие, упрощения машинных расчётов.

Допущение 11. Оба входящих потока активируют одно и то же правило на позиции «К». Допущение использовано в целях упрощения системы.

Допущение 12. В модели не рассматривается возможность того, что пакет был повреждён при передаче или для него не был найден маршрут, а также не рассматривается перепосылка пакетов.

Выводы раздела

1. В результате анализа подходов к организации процесса обслуживания пакетов в межсетевых экранах выявлена концептуальная схожесть организации работы очередей и работы с памятью, поддержки «первого» и «быстрого» путей обслуживания пакетов, поддержки технологий фильтрации трафика на различных уровнях модели OSI, а также принципов функционирования механизмов QoS у межсетевых экранов различных производителей. Выявленные различия заключаются в наличие/отсутствие определённых операций/проверок, их расположение в процессе обслуживания и очерёдности выполнения, а также в различии предусмотренных наборов механизмов QoS. Выявленные различия затрудняют формирование универсальной математической модели.

2. Сочетание особенности обработки пакетов межсетевыми экранами и значительной задержки обслуживания, вносимой ими, может быть скомпенсировано функционированием механизмов приоритизации трафика в их входных очередях, которые позволят обеспечить определённым классам трафика необходимый уровень качества обслуживания.

3. В качестве основы модели межсетевого экрана выбран процесс обработки пакетов, принятый в межсетевом экране типа IPTables/NetFilter. Это сделано по причине того, что существует ряд проработанных математических моделей функционирования этого типа межсетевых экранов со схожими целями моделирования, но без учёта приоритизации обслуживания трафика. Межсетевые экраны типа IPTables/NetFilter имеет широкий спектр общедоступной документации и сопутствующих программных пакетов.

4. Сформулирована содержательная постановка задачи на разработку моделей функционирования межсетевого экрана в условиях приоритизации обслуживания трафика в их входных очередях. К содержательной постановке задачи сформулированы технические допущения 1-9 и логические допущения 10-12 для упрощений модели.

РАЗДЕЛ 3. РАЗРАБОТКА МАТЕМАТИЧЕСКОЙ МОДЕЛИ ФУНКЦИОНИРОВАНИЯ МЕЖСЕТЕВОГО ЭКРАНА В УСЛОВИЯХ ПРИОРИТИЗАЦИИ ОБСЛУЖИВАНИЯ ТРАФИКА

3.1. Постановка задачи на разработку математической модели

3.1.1. Адаптация содержательной постановки задачи

Адаптация содержательной постановки задачи из п. 2.3.2.

На вход модели поступают пакеты (заявки), представленные сетевыми пакетами. Пакеты могут поступать в систему от различных независимых друг от друга источников (конечных пользователей). Это не противоречит концепции приоритизации обслуживания трафика, так как каждому источнику, группе источников может быть присвоен приоритет обслуживания тем или иным техническим способом [110, 111]. Пакеты от большого числа источников после агрегации становятся потоками приоритетных и неприоритетных заявок, поступающими на вход модели. Взаимное наложение большого числа малых независимых ординарных потоков с различным последействием (теорема Хинчина), а также преобразование потока в сети в пределе даёт поток близкий к простейшему [110]. Настоящий факт позволяет считать входящие потоки пакетов простейшими. Хотя время между пакетами (заявками) в современных пакетных сетях подчиняется несколько более сложным законам распределения, отличным от экспоненциального, допущение обеспечит необходимый уровень приближения результатов.

Допущение 13. На вход математической модели поступают простейшие потоки вызовов, время между приходом пакетов является случайной величиной распределённой по экспоненциальному закону.

Обслуживание пакетов в специализированном сетевом оборудовании с высокой степенью аппаратной обработки является постоянной величиной для каждой отдельной операции обслуживания. Для устройств с низкой степенью аппаратной обработки зависимость времени обработки от загрузки устройства не столь очевидна, ввиду чего корректней рассматривать её в качестве случайной

величины, распределённой по экспоненциальному закону. Забегая вперёд отметим, что в рамках имитационного моделирования (раздел 4) проведена работа по моделированию функционирования МЭ при постоянной длительности обслуживания.

Допущение 14. Время обслуживания заявок в математической модели является случайной величиной распределённой по экспоненциальному закону.

В результате анализа и адаптации содержательной постановки задачи, а также принятых допущений использован аппарат марковских процессов с непрерывным временем для реализации математической модели.

Представим МЭ в виде *системы массового обслуживания* (СМО). Математическая модель представляет собой однолинейную СМО с ограниченным накопителем пакетов (заявок), комбинированными потерями и смешанными приоритетами обслуживания (абсолютным приоритетом постановки в очередь и относительным приоритетом постановки на обслуживание). На вход модели поступает простейший поток пакетов (вызовов). В соответствии с модифицированной классификацией Кенделла СМО будет обозначаться следующим образом $\overrightarrow{M} / M / 1 / L / f_1^{22}$ [113-114].

При разработке математической постановки задачи, *системы уравнений равновесия* (СУР), диаграммы состояний и переходов, а также математических методов и процедур использовались источники по теории сетей массового обслуживания и способам расчёта показателей производительности структурно-сетевого оборудования сетей связи [110, 112-116].

3.1.2. Стадии разработки математической модели

Выделим следующие стадии разработки математической модели:

- постановка задачи, поиск способа реализации математической модели:
  - разработка содержательной и математической постановок задачи: аспирант Мусатов В.К., аспирант Щербанская А.А.;
  - консультации: проф. Пшеничников А.П., проф. Самуйлов К.Е.;

– разработка математической модели:
  - разработка СУР: аспирант Щербанская А.А., аспирант Мусатов В.К.;
  - разработка математических процедур: аспирант Щербанская А.А.;

– воплощение математической модели в среде математического моделирования и компьютерной алгебры: аспирант Мусатов В.К.;

– поиск ошибок, отладка алгоритма, корректировка математических процедур:
  - разработка тестов, поиск ошибок, откладка, корректировка математической модели: аспирант Мусатов В.К.;
  - разработка тестов, корректировка математических процедур: аспирант Щербанская А.А., аспирант Мусатов В.К.;

– проведение математического моделирования: аспирант Мусатов В.К.

– агрегация результатов моделирования:
  - агрегация результатов: аспирант Мусатов В.К., аспирант Щербанская А.А.;
  - консультации: проф. Пшеничинов А.П., проф. Самуйлов К.Е.;

– выводы по результатам математического моделирования:
  - разработка выводов: аспирант Мусатов В.К.;
  - консультации: проф. Пшеничинов А.П., проф. Самуйлов К.Е., аспирант Щербанская А.А.

3.2. Разработка математической модели

3.2.1. Математическая постановка задачи

Рассматривается однолинейная СМО с ограниченной очередью длины L. В систему поступают два независимых пуассоновских потока заявок с интенсивностями λ_1 и λ_2. Обслуживание на приборе состоит из K этапов, время обслуживания заявок на которых имеет экспоненциальное распределение с параметрами $\mu*$ – для первого этапа и μ – для остальных $(K-1)$ этапов. Приоритетные пакеты имеют относительный приоритет в обслуживании и

абсолютный приоритет в постановке в очередь, по сравнению с неприоритетными пакетами. При наличии в очереди пакетов обоих приоритетов, на обслуживание поступает приоритетный пакет, а неприоритетный пакет может поступить на обслуживание только тогда, когда в очереди нет приоритетных пакетов. Если накопитель полностью заполнен и поступает приоритетный пакет, то он сбрасывает из накопителя последний поступивший неприоритетный пакет (при его наличии) и занимает место в накопителе. Пакеты одного приоритета обслуживаются в порядке поступления (дисциплина FIFO).

3.2.2. Система уравнений равновесия

Для описания статической системы, на которую подаются различные по величине потоки заявок, необходимо сформировать систему уравнений равновесия (далее – СУР).

Пусть $\eta(t)$ – описывает номер этапа обслуживания пакета на приборе и состояние «СМО пуста» в момент времени t, $0 \le \eta(t) \le K$. При $\eta(t) = 0$ в системе нет пакетов, если $\eta(t) = K$ то пакет находиться на K-ом этапе обслуживания. Обозначим $\nu_i(t),\ i = 1,2$ – число приоритетных $(i = 1)$ и неприоритетных $(i = 2)$ пакетов в очереди в момент времени t, $0 \le \nu_i(t) \le L,\ \nu_1(t) + \nu_2(t) \le L,$ при $\eta(\mathrm{t}) = 0,\ \nu_i(\mathrm{t}) = 0,\ i = 1,2$. Далее $\nu_1(t) = i,\ \nu_2(t) = j$. Определив основные характеристики (состояния) исследуемой СМО, обозначим процесс поведения исследуемой СМО через $X(t) = (\nu_1(t), \nu_2(t), \eta(t))$.

Процесс $\{X(t), t \ge 0\}$, описывающий поведение СМО, является марковским случайным процессом с непрерывным временем и дискретным множеством состояний: $S = \left\{(0,0,0);\ (i,j,k),\ 0 \le i \le L,\ 0 \le j \le L,\ i + j \le L,\ k = \overline{1,\mathrm{K}}\right\}$.

$K = 0$ только при состоянии $(0,0,0)$, то есть состоянии, при котором межсетевой экран пуст и не обслуживает пакетов ($K \ne 0$, при $i > 0,\ j > 0$**).**

Обозначим через $p_{i.j.k}$ – стационарную вероятность того, что система находится в состоянии (i, j, k), т.е. в накопителе находится i приоритетных

пакетов, j неприоритетных пакетов, и на приборе обслуживается пакет на этапе под номером k. Стационарная вероятность $p_{0.0.0}$ означает, что в системе нет пакетов. Учитывая вышеизложенное, представим диаграмму состояний и переходов на рисунке Б.1 в приложении Б, а формирование СУР в приложении В.

Очевидно, что все состояния марковского процесса $\{X(t),\ t \geq 0\}$ сообщаются, что наглядно отображено на рисунке Б.1. Поэтому этот процесс является эргодическим и существуют строго положительные предельные вероятности $p_{i.j.k},\ 0 \leq i \leq L,\ 0 \leq j \leq L,\ 0 \leq i + j \leq L,\ k = \overline{1,\mathrm{K}}$, независящие от начального состояния процесса, являющиеся стационарными вероятностями, что подтверждается в [116]. Таким образом, стационарное распределение существует и удовлетворяет следующей СУР:

$$(\lambda_1 + \lambda_2) p_{0.0.0} = \mu p_{0.0.\mathrm{K}}, \tag{1}$$

$$(\lambda_1 + \lambda_2 + \mu^*) p_{0.0.1} = (\lambda_1 + \lambda_2) p_{0.0.0} + \mu p_{1.0.\mathrm{K}} + \mu p_{0.1.\mathrm{K}}, \tag{2}$$

$$(\lambda_1 + \lambda_2 + \mu_1) p_{i.0.1} = \lambda_1 p_{i-1.0.1} + \mu p_{i+1.0.\mathrm{K}},\ 0 < i < L, \tag{3}$$

$$(\lambda_1 + \lambda_2 + \mu^*) p_{0.j.1} = \lambda_2 p_{0.j-1.1} + \mu p_{0.j+1.\mathrm{K}} + \mu p_{1.j.\mathrm{K}}\ , 0 < j < L, \tag{4}$$

$$\mu^* p_{L.0.1} = \lambda_1 p_{L-1.0.1} + \lambda_1 p_{L-1.1.1}, \tag{5}$$

$$(\lambda_1 + \mu^*) p_{0.L.1} = \lambda_2 p_{0.L-1.1}, \tag{6}$$

$$(\lambda_1 + \lambda_2 + \mu^*) p_{i.j.1} = \lambda_1 p_{i-1.j.1} + \lambda_2 p_{i.j-1.1} + \mu p_{i+1.j.K},\ i + j < L,\ i > 0,\ j > 0, \tag{7}$$

$$(\lambda_1 + \mu^*) p_{i.j.1} = \lambda_1 p_{i-1.j.1} + \lambda_2 p_{i.j-1.1} + \lambda_1 p_{i-1.j+1.1},\ i + j = L,\ i > 0,\ j > 0, \tag{8}$$

$$(\lambda_1 + \lambda_2 + \mu) p_{0.0.2} = \mu^* p_{0.0.1}, \tag{9}$$

$$(\lambda_1 + \lambda_2 + \mu) p_{i.0.2} = \lambda_1 p_{i-1.0.2} + \mu^* p_{i.0.1},\ 0 < i < L, \tag{10}$$

$$(\lambda_1 + \lambda_2 + \mu) p_{0.j.2} = \lambda_2 p_{0.j-1.2} + \mu^* p_{0.j.1},\ 0 < j < L, \tag{11}$$

$$\mu p_{L.0.2} = \lambda_1 p_{L-1.0.2} + \lambda_1 p_{L-1.1.2} + \mu^* p_{L.0.1}, \tag{12}$$

$$(\lambda_1 + \mu) p_{0.L.2} = \lambda_2 p_{0.L-1.2} + \mu^* p_{0.L.1}, \tag{13}$$

$$(\lambda_1 + \lambda_2 + \mu) p_{i.j.2} = \lambda_1 p_{i-1.j.2} + \lambda_2 p_{i.j-1.2} + \mu^* p_{i.j.1}, i + j < L,\ i > 0,\ j > 0, \tag{14}$$

$$(\lambda_1 + \mu) p_{i.j.2} = \lambda_1 p_{i-1.j.2} + \lambda_2 p_{i.j-1.2} + \lambda_1 p_{i-1.j+1.2} + \mu^* p_{i.j.1}, i + j = L,\ i > 0,\ j > 0, \tag{15}$$

$$(\lambda_1+\lambda_2+\mu)p_{0.0.k}=\mu p_{0.0.k-1},\ 2<k\le K, \tag{16}$$

$$(\lambda_1+\lambda_2+\mu)p_{i.0.k}=\lambda_1 p_{i-1.0.k}+\mu p_{i.0.k-1},\ 0<i<L,\ 2<k\le K, \tag{17}$$

$$(\lambda_1+\lambda_2+\mu)p_{0.j.k}=\lambda_2 p_{0.j-1.k}+\mu p_{0.j.k-1},\ 0<j<L,\ 2<k\le K, \tag{18}$$

$$\mu p_{L.0.k}=\lambda_1 p_{L-1.0.k}+\lambda_1 p_{L-1.1.k}+\mu p_{L.0.k-1},\ 2<k\le K, \tag{19}$$

$$(\lambda_1+\mu)p_{0.L.k}=\lambda_2 p_{0.L-1.k}+\mu p_{0.L.k-1},\ 2<k\le K, \tag{20}$$

$$(\lambda_1+\lambda_2+\mu)p_{i.j.k}=\lambda_1 p_{i-1.j.k}+\lambda_2 p_{i.j-1.k}+\mu p_{i.j.k-1},\quad i+j<L,\ i>0,\ j>0, 2<k\le K, \tag{21}$$

$$(\lambda_1+\mu)p_{i.j.k}=\lambda_1 p_{i-1.j.k}+\lambda_2 p_{i.j-1.k}+\lambda_1 p_{i-1.j+1.k}+\mu p_{i.j.k-1},\quad i+j=L,\ i>0,\ j>0,\ 2<k\le K. \tag{22}$$

Также стационарные вероятности удовлетворяют условию нормировки:

$$p_{0.0.0}+\sum_{i=0}^{L}\sum_{j=0}^{L-i}\sum_{k=1}^{K}p_{i.j.k}=1. \tag{23}$$

Диаграмма состояний и переходов, продемонстрированная на рисунке Б.1, представляет собой трёхмерную структуру. По диаграмме состояний и переходов сформирована СУР, состоящая из 22 групп уравнений (1-22) и уравнения нормировки (23).

Пусть p_n – стационарная вероятность того, что в системе находится n пакетов обоих приоритетов, тогда

$$p_n=\sum_{l=0}^{n-1}\sum_{k=1}^{K}p_{l.n-l-1.k},\ i>0;\ p_0=p_{0.0.0}, 1\le \mathrm{n}\le L+1.$$

Вероятности потери заявок. Обозначим через p_1^{loss} – вероятность сброса из заполненного накопителя неприоритетный пакет в случае, если в систему поступает приоритетный пакет, а через p_2^{loss} – вероятность потери неприоритетного пакета при поступлении в систему с полностью заполненной очередью.

$$p_1^{loss}=\frac{\lambda_1}{\lambda_1+\lambda_2}\sum_{i=0}^{L-1}\sum_{k=1}^{K}p_{i.L-i.k},$$

$$p_2^{loss}=\frac{\lambda_2}{\lambda_1+\lambda_2}\sum_{i=0}^{L}\sum_{k=1}^{K}p_{i.L-i.k}.$$

Таким образом, полная вероятность потери неприоритетного пакета равна:

$$p_{unprior}^{loss} = p_1^{loss} + p_2^{loss} = \frac{\lambda_1}{\lambda_1 + \lambda_2} \sum_{i=0}^{L-1} \sum_{k=1}^{K} p_{\mathrm{i.L}-i.k} + \frac{\lambda_2}{\lambda_1 + \lambda_2} \sum_{i=0}^{L} \sum_{k=1}^{K} p_{\mathrm{i.L}-i.k}.$$

Вероятность потери приоритетного пакета имеет вид:

$$p_{prior}^{loss} = \frac{\lambda_1}{\lambda_1 + \lambda_2} \sum_{k=1}^{K} p_{L.0.k}.$$

Показатели заполнения очереди. Стационарное распределение позволяет получить средние показатели заполнения СМО пакетами. Пусть N – среднее число пакетов в СМО, тогда

$$N = \sum_{i=0}^{L} \sum_{j=0}^{L-i} \sum_{k=1}^{K} (i + j + 1) p_{i.j.k}.$$

Обозначим через Q_{sum} – среднее число пакетов в очереди, через Q_{prior} и $Q_{unprior}$ – среднее число приоритетных и неприоритетных пакетов в очереди.

$$Q_{sum} = \sum_{i=0}^{L} \sum_{j=0}^{L-i} \sum_{k=1}^{K} (i + j) p_{i.j.k},$$

$$Q_{prior} = \sum_{i=0}^{L} \sum_{j=0}^{L-i} \sum_{k=1}^{K} i p_{i.j.k},$$

$$Q_{unprior} = \sum_{i=0}^{L} \sum_{j=0}^{L-i} \sum_{k=1}^{K} j p_{i.j.k}.$$

3.2.3. Математические методы и процедуры

Попытка найти решение СУР (1)-(23) в аналитическом виде не удалась. Вследствие того, что задача решения СУР в аналитическом виде не ставилась, то было решено использовать численные методы для нахождения стационарного распределения.

Вследствие того, что диаграмма состояний и переходов трёхмерная для хранения переходных вероятностей необходимо использовать трёхмерную матрицу, что значительно усложняет проведение матричных операций и расчётов. По этой причине разработана специальная функция, названная функцией

формирования матрицы, которая переводит трёхмерный массив данных в двумерную форму (матрицу).

При поиске наиболее удобной функции формирования матрицы было замечено, что при специальном виде функции формирования матрицы, приведенном ниже, матрица переходных вероятностей «Q» принимает блочный трехдиагональный вид:

$$Q=\begin{pmatrix} B_0 & A_1 & 0 & \cdots & 0 & 0 & 0 \\ C_0 & B_1 & A_2 & \cdots & 0 & 0 & 0 \\ 0 & C_1 & B_2 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & B_{L-1} & A_L & 0 \\ 0 & 0 & 0 & \cdots & C_{L-1} & B_L & A_{L+1} \\ 0 & 0 & 0 & \cdots & 0 & C_L & B_{L+1} \end{pmatrix}. \qquad (24)$$

При использовании матричных методов расчёта применялись материалы книги [115].

Матрица «Q» имеет модульную структуру и состоит из матриц, делящихся на 3 типа «A,B,C». Блоки типов «A,B,C», в свою очередь, также являются модульными структурами и состоят из блоков, формируемых по принципам, описанным далее.

Алгоритм формирования матрицы специального вида, приводящий матрицу переходных вероятностей «Q» в блочный трёхдиагональный вид, приведён ниже. Состояния системы переводятся в одномерный вектор. Двухмерность обеспечивается комплексом значения функции и номером уравнения из СУР.

```
Ввод: i, j, k, K
Вывод: FFM
1: If i=0 And j=0 And k=0 Then
2:     FFM=1
3: Else
4:     If i=0 And j=0 Then
5:         FFM=k+1
6:     Else
```

```
7:        If i=0 And j>0 Then
8:           FFM=1+M · Σ_{x=0}^{j} x+k
9:        Else
10:          If i>0 And j=0 Then
11:             FFM=1+K · i+K · Σ_{x=0}^{i} x+k
12:          Else
13:             If i >0 And j>0 Then
14:                FFM = FFM(0, i+j, K , K)+K · (i-1)+k
15:             End If
16:          End If
17:       End If
18:    End If
19: End If
20: Return FFM
```

Правила формирования матрицы переходных вероятностей

Блок $B_0 = [-\lambda_1 - \lambda_2]$; блок B_1 имеет размер $K \times K$ и следующий вид:

$$B_1 = \begin{pmatrix} -\lambda_1 - \lambda_2 - \mu * & \mu * & 0 & \cdots & 0 & 0 \\ 0 & -\lambda_1 - \lambda_2 - \mu & \mu & \cdots & 0 & 0 \\ 0 & 0 & -\lambda_1 - \lambda_2 - \mu & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \mu & 0 \\ 0 & 0 & 0 & \cdots & -\lambda_1 - \lambda_2 - \mu & \mu \\ 0 & 0 & 0 & \cdots & 0 & -\lambda_1 - \lambda_2 - \mu \end{pmatrix}.$$

Блоки $B_i, i = \overline{2, L}$ имеют размер $(i \cdot K) \times (i \cdot K)$ и состоят из i блоков B_1, стоящих на диагонали. Например, блок $B_3 = \begin{pmatrix} B_1 & 0 & 0 \\ 0 & B_1 & 0 \\ 0 & 0 & B_1 \end{pmatrix}$, где 0 представляют собой нулевые матрицы соответствующего размера.

Последний диагональный блок B_{L+1} имеет размер $(L+1)\cdot K \times (L+1)\cdot K$ и состоит из L подблоков B'_{L+1} размера $K \times K$, размещённых по диагонали и имеющих вид

$$B'_{L+1} = \begin{pmatrix} -\lambda_1 - \mu * & \mu * & 0 & \cdots & 0 & 0 \\ 0 & -\lambda_1 - \mu & \mu & \cdots & 0 & 0 \\ 0 & 0 & -\lambda_1 - \mu & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \mu & 0 \\ 0 & 0 & 0 & \cdots & -\lambda_1 - \mu & \mu \\ 0 & 0 & 0 & \cdots & 0 & -\lambda_1 - \mu \end{pmatrix}$$

и блока размера $K \times K$, стоящего в последней позиции диагонали, имеющего следующий вид:

$$\begin{pmatrix} -\mu * & \mu * & 0 & \cdots & 0 & 0 & 0 \\ 0 & -\mu & \mu & \cdots & 0 & 0 & 0 \\ 0 & 0 & -\mu & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\mu & \mu & 0 \\ 0 & 0 & 0 & \cdots & 0 & -\mu & \mu \\ 0 & 0 & 0 & \cdots & 0 & 0 & -\mu \end{pmatrix},$$

кроме того, над диагональными подблоками блока B_{L+1} находятся диагональные блоки размера $K \times K$, у которых на диагоналях стоят λ_1.

Блок C_0 размера $K \times 1$ состоит из μ, стоящей внизу блока, и $(K-1)$ нулей.

Блок $\tilde{C}_0$ размера $K \times K$, в левом нижнем углу которого стоит μ, а все остальные элементы нули. Тогда:

$$C_1 = \begin{pmatrix} \tilde{C}_0 \\ \tilde{C}_0 \end{pmatrix};\ C_2 = \begin{pmatrix} C_1 & 0 \\ 0 & \tilde{C}_0 \end{pmatrix} = \begin{pmatrix} \tilde{C}_0 & 0 \\ \tilde{C}_0 & 0 \\ 0 & \tilde{C}_0 \end{pmatrix};\ C_3 = \begin{pmatrix} C_1 & 0 & 0 \\ 0 & \tilde{C}_0 & 0 \\ 0 & 0 & \tilde{C}_0 \end{pmatrix} \text{ и т.д.}$$

Блок A_1 имеет размер $1\times K$, состоит из элемента $(\lambda_1+\lambda_2)$, стоящего слева, и $(K-1)$ нулей справа.

Введем блоки Λ_1 и Λ_2 – диагональные матрицы размера $K\times K$, на диагонали которых размещены λ_1 и λ_2 соответственно. Тогда

$$A_2=\begin{pmatrix}\Lambda_2 & \Lambda_1\end{pmatrix};\ A_3=\begin{pmatrix}\Lambda_2 & \Lambda_1 & 0\\ 0 & \Lambda_2 & \Lambda_1\end{pmatrix};\ A_4=\begin{pmatrix}\Lambda_2 & \Lambda_1 & 0 & 0\\ 0 & \Lambda_2 & \Lambda_1 & 0\\ 0 & 0 & \Lambda_2 & \Lambda_1\end{pmatrix}\ \text{и т.д.}$$

Используя представленную выше информацию, можно полностью сформировать матрицу переходных вероятностей.

Сформировав матрицу переходных вероятностей, имеем систему уравнений равновесия:

$$\begin{cases}\overline{x_0}^T B_0+\overline{x_1}^T C_0=0\\ \overline{x_{k-1}}^T A_k+\overline{x_k}^T B_k+\overline{x_{k+1}}^T C_k=0,\ 1\le k\le L\\ \overline{x_L}^T A_{L+1}+\overline{x_{L+1}}^T B_{L+1}=0\end{cases}\qquad(25)$$

Исходя из вида матрицы переходных вероятностей, воспользуемся матричным методом UL-разложение. Матрица Q представима в виде:

$$Q=\begin{pmatrix}I & -R_0 & 0 & \cdots & 0 & 0 & 0\\ 0 & I & -R_1 & \cdots & 0 & 0 & 0\\ 0 & 0 & I & \cdots & 0 & 0 & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots\\ 0 & 0 & 0 & \cdots & I & -R_{L-1} & 0\\ 0 & 0 & 0 & \cdots & 0 & I & -R_L\\ 0 & 0 & 0 & \cdots & 0 & 0 & I\end{pmatrix}\begin{pmatrix}\Phi_0 & 0 & 0 & \cdots & 0 & 0 & 0\\ C_0 & \Phi_1 & 0 & \cdots & 0 & 0 & 0\\ 0 & C_1 & \Phi_2 & \cdots & 0 & 0 & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots\\ 0 & 0 & 0 & \cdots & \Phi_{L-1} & 0 & 0\\ 0 & 0 & 0 & \cdots & C_{L-1} & \Phi_L & 0\\ 0 & 0 & 0 & \cdots & 0 & C_L & \Phi_{L+1}\end{pmatrix}.$$

Для нее выполняются следующие рекуррентные формулы:

$$\Phi_{L+1}=B_{L+1},$$
$$R_r=-A_{r+1}\Phi_{r+1}^{-1},$$
$$\Phi_r=B_r+R_rC_r,$$
$$\textit{при}\ r=L,L-1,\ldots,0.$$

Решение системы (25) запишем в матрично-мультипликативном виде:

$$\overline{x_k}^T = \overline{x_0}^T R_0 R_1 ... R_{k-1}, 1 \le k \le L+1,$$

где $\overline{x_0}^T$ может быть получено из следующего уравнения:

$$\overline{x_0}^T (\mathrm{B}_0 + R_0 \cdot C_0) = \overline{0}^T.$$

После нахождения всех векторов $\overline{x_k}^T$ необходимо просуммировать все значения, содержащиеся в них, и с помощью полученного числа провести нормировку значений векторов $\overline{x_k}^T$. Затем необходимо сложить все вектора в один в порядке их индекса «k».

Получившийся вектор содержит значения всех стационарных вероятностей состояний исследуемой СМО. Значения стационарных вероятностей расположены в векторе не по очерёдности, а в соответствии с функцией формирования матрицы, с которой можно ознакомиться на странице 69, в статье автора диссертации [117] или в приложении Г (функция представлена на языке Wolfram Mathematica).

3.3. Оценка результатов моделирования в системах компьютерной алгебры и математического моделирования

3.3.1. Общие сведения

При выборе системы компьютерной алгебры и математического моделирования рассматривались 3 программных продукта: PTC Mathcad Prime 3.0, Matlab R2014a, Wolfram Mathematica 10. Среда Mathcad была исключена из-за недостаточного уровня предоставляемого функционала, точности и производительности. Сравнение Matlab и Mathematica показало, что в Mathematica удобнее производить символьные вычисления, что упрощает прототипирование и отладку кода. В процессе разработки математической модели было проверено несколько способов расчётов искомых показателей, которые в результате оказались хуже, чем способ, представленный в подразделе 3.2, без прототипирование эта работа заняла бы заметно больше времени. Учитывая изначальное отсутствие опыта использования Matlab и Mathematica и факт, что

обе среды платные, было решено проводить исследование на Wolfram Mathematica 10.0.1.0 [118], которая позволяла получить во временном (триальном) режиме полнофункциональную среду расчётов.

Использованный для расчётов код математической модели на языке Wolfram Mathematica представлен в приложении Г.

Математическая модель оперирует большими объёмами данных, хранящимися в матричной форме, а также производит алгебраические и матричные операции. Этот факт создаёт высокую нагрузку на оперативную память персонального компьютера. При расчётах использовался персональный компьютер, имеющий 16 Гбайт свободной оперативной памяти с файлом подкачки 64 Гбайт.

3.3.2. Исходные данные

За исходные данные, с учётом ограничения аппаратных ресурсов, были выбраны следующие значения: количество этапов обслуживания $K = 30$, количество мест в очереди $L = 30$, время обслуживания распределено экспоненциально с параметром $\mu^* = 2 \cdot 10^7, \mu = 3.77 \cdot 10^5$. Значения μ^*, μ взяты из работы [20] в связи с проведёнными в ней экспериментальными измерениями. Предельный поток трафика, который может обслужить МЭ, равен 246931 пакет в секунду, эту точку будем называть «точкой отказа». Общая динамика системы сохраняется при подборе подходящих соотношений $K, L, \mu^*, \mu.$

3.3.3. Сценарии поведения трафика на входе модели

Для проверки корректности функционирования математической модели необходимо поставить её в кардинально различные условия поведения входящего трафика. Рассматриваются 3 сценария поведения трафика.

Первый сценарий – поток приоритетного трафика постоянный, поток неприоритетного трафика растет, превышая обслуживающую способность МЭ. При работе устройств на реальной сети наиболее вероятен именно этот сценарий. Он может произойти при появлении «взрывного интереса» к информации (например, оперативной информации, важных новостей, «вирусных вбросах»)

какого-либо ресурса сети, в связи с чем, произойдёт лавинообразное повышение интенсивности неприоритетного трафика к этому ресурсу сети. В том числе подобное может появиться при ошибочной конфигурации оборудования сети или его выхода из строя с последующим перераспределением нагрузки.

Второй сценарий – поток неприоритетного трафика постоянный, поток приоритетного трафика растёт, превышая обслуживающую способность МЭ. Подобный сценарий маловероятен. Такое может случиться при broadcast-шторме, если его пакеты будут иметь приоритет выше, чем besteffort, что маловероятно, однако, может случиться с сигнальным (управляющим) трафиком сети или при некорректной настройке оборудования. Обязательным условием этого события должны быть некорректная настройка механизмов ограничения и сглаживания трафика или их отключение.

Третий сценарий – поток приоритетного и не приоритетного трафика равномерно нарастает, превышая обслуживающую способность МЭ. Такой сценарий маловероятен, но представляет интерес для оценки работы модели.

3.3.4. Результаты расчётов

Первый сценарий. Для первого сценария дополнительные исходные данные по суммарному входящему потоку трафика ($\lambda_1 + \lambda_2$) выглядят следующим образом $\lambda_1 = 40 \cdot 10^3 = const$, $\lambda_2 = 110 \cdot 10^3, 120 \cdot 10^3, 130 \cdot 10^3, \ldots, 280 \cdot 10^3$.

Рисунок 3.1 демонстрирует динамику изменения интенсивности обслуженного потока пакетов в зависимости от интенсивности поступающего потока пакетов.

По рисунку 3.1 хорошо видно, в какой-то момент времени МЭ перестаёт обслуживать трафик, то есть достигает точки отказа. Поток приоритетных пакетов не несёт потерь при повышении потока неприоритетных пакетов. Это также подтверждает корректность работы модели.

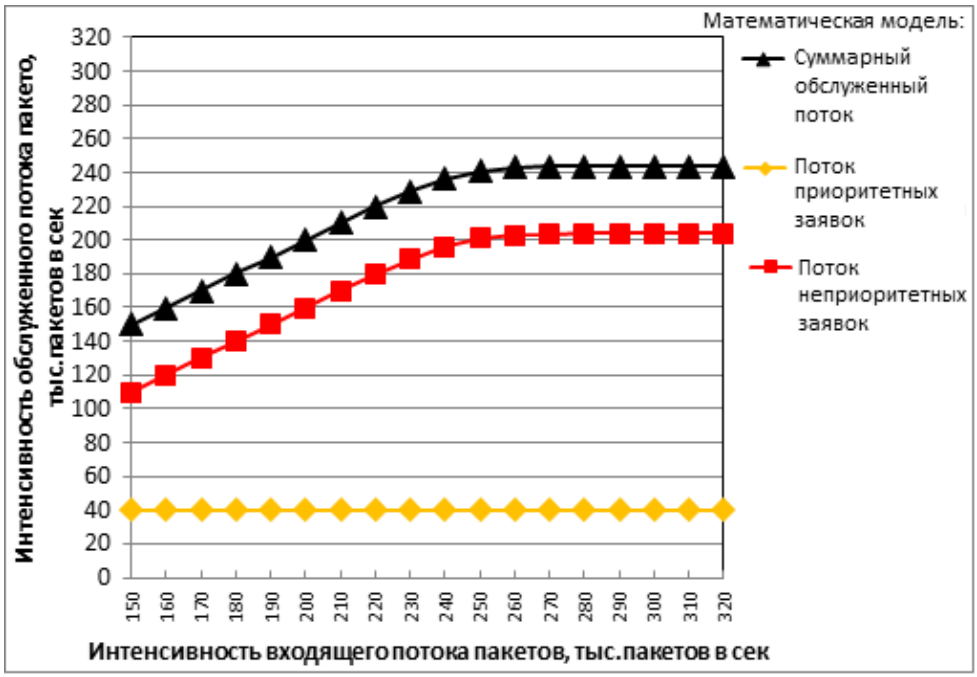


Рисунок 3.1 – Интенсивность обслуженного трафика (1-ый сценарий)

На рисунке 3.2 показана зависимость потерь неприоритетных пакетов от интенсивности поступающего потока пакетов. Потери начинают проявляться несколько ранее достижения точки отказа вследствие экспоненциального характера обслуживания и поступающего потока пакетов. После прохождения точки отказа рост потерь приобретает линейный характер, зависящий от соотношения входящих потоков пакетов. График потерь приоритетных пакетов не представлен потому, что у потока приоритетных пакетов отсутствуют потери.

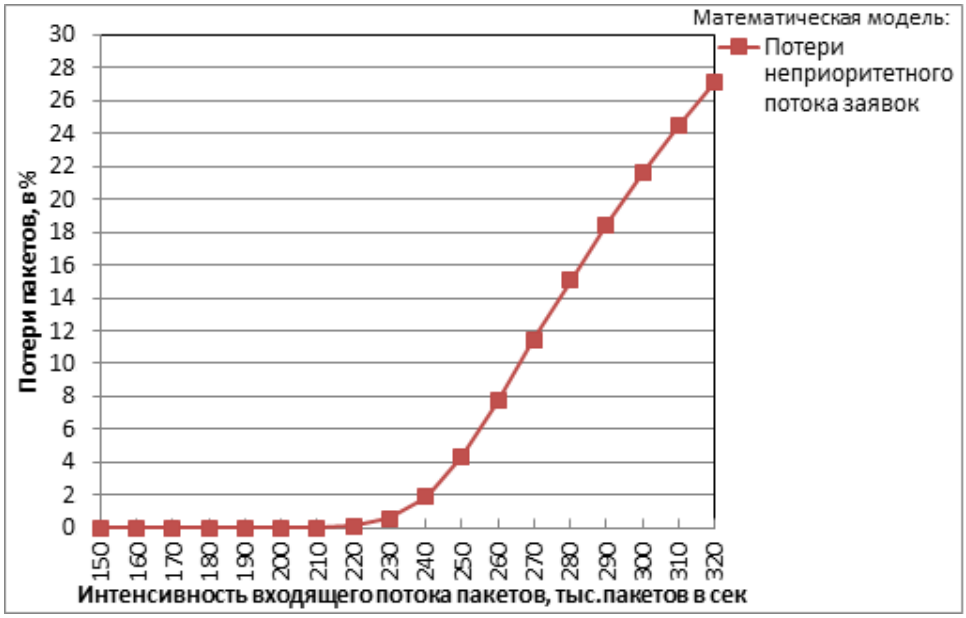


Рисунок 3.2 – Потери пакетов (1-ый сценарий)

Рисунок 3.3 демонстрирует динамику заполнения очереди в зависимости от интенсивности поступающего потока пакетов. Из графика видно, что очередь в основном, занята неприоритетными пакетами. При максимальной интенсивности входящего потока пакетов среднее количество приоритетных пакетов в очереди равно 0,13947 пакета. Очередь не заполняется полностью по причине экспоненциального характера времени обслуживания и входящих потоков пакетов.

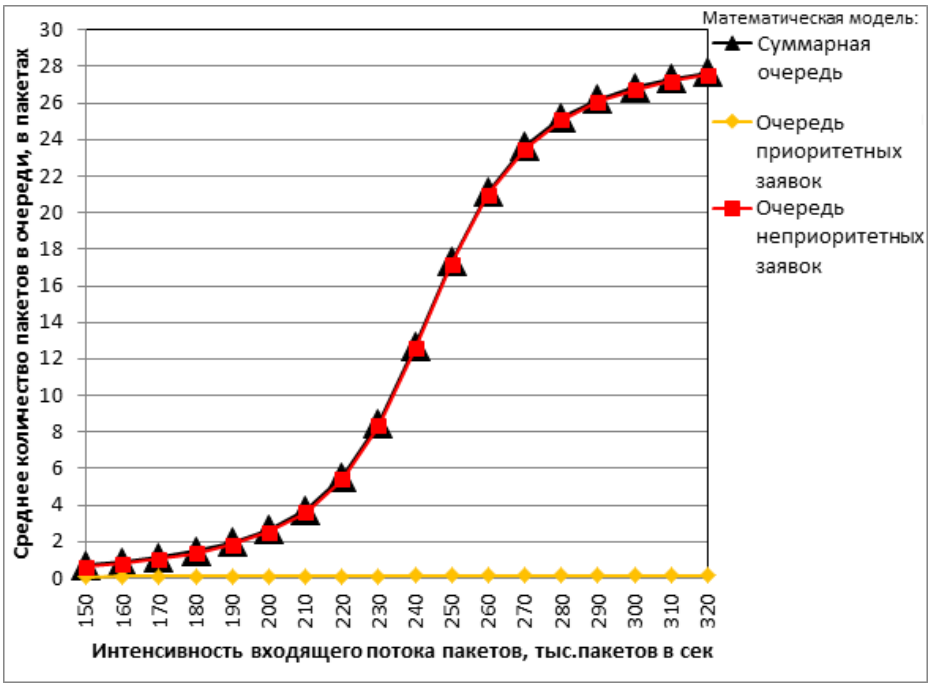


Рисунок 3.3 – Заполнение пакетной очереди (1-ый сценарий)

Второй сценарий. Для второго сценария дополнительные исходные данные по суммарному входящему потоку трафика ($\lambda_1 + \lambda_2$) выглядят следующим образом $\lambda_1 = 110 \cdot 10^3$, $120 \cdot 10^3$, $130 \cdot 10^3$, ... , $280 \cdot 10^3$, $\lambda_2 = 40 \cdot 10^3 = const$.

Рисунок 3.4 демонстрирует динамику изменения интенсивности обслуженного потока пакетов в зависимости от интенсивности поступающего потока пакетов.

Из рисунка 3.4 видно, что в момент, когда МЭ достигает точки отказа, объём обслуженных неприоритетных пакетов падает, так как их вытесняют приоритетные пакеты.

Рисунок 3.5 демонстрирует динамику изменения потерь в зависимости от интенсивности поступающего потока пакетов.

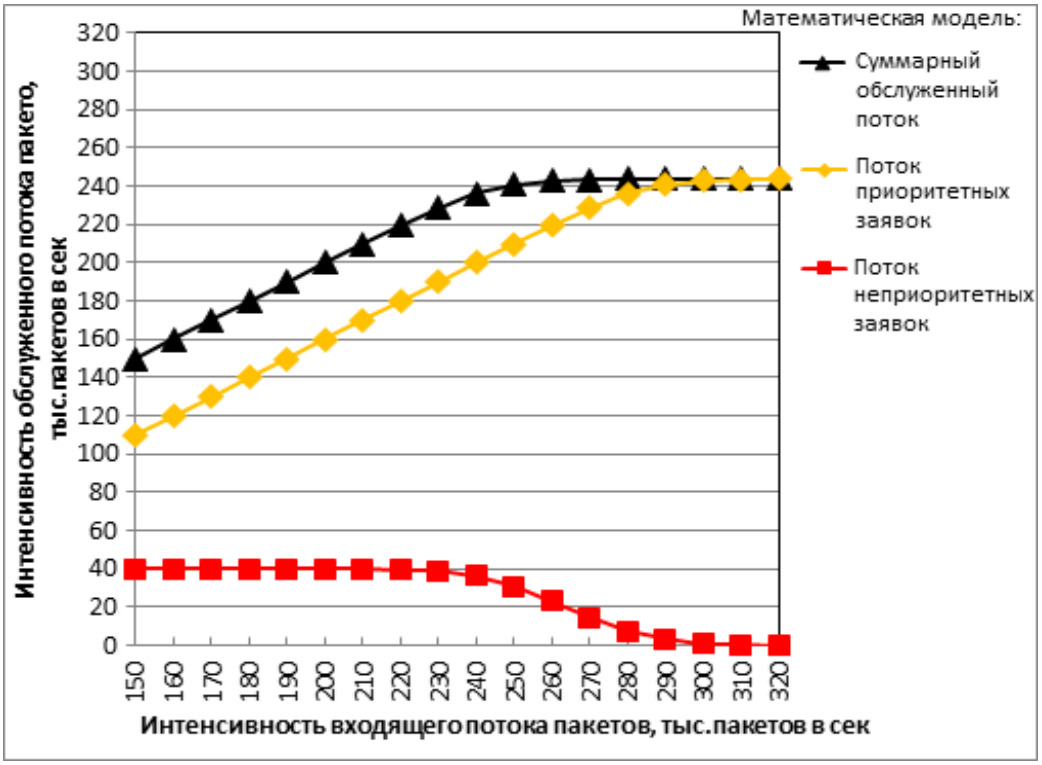


Рисунок 3.4 – Интенсивность обслуженного трафика (2-ой сценарий)

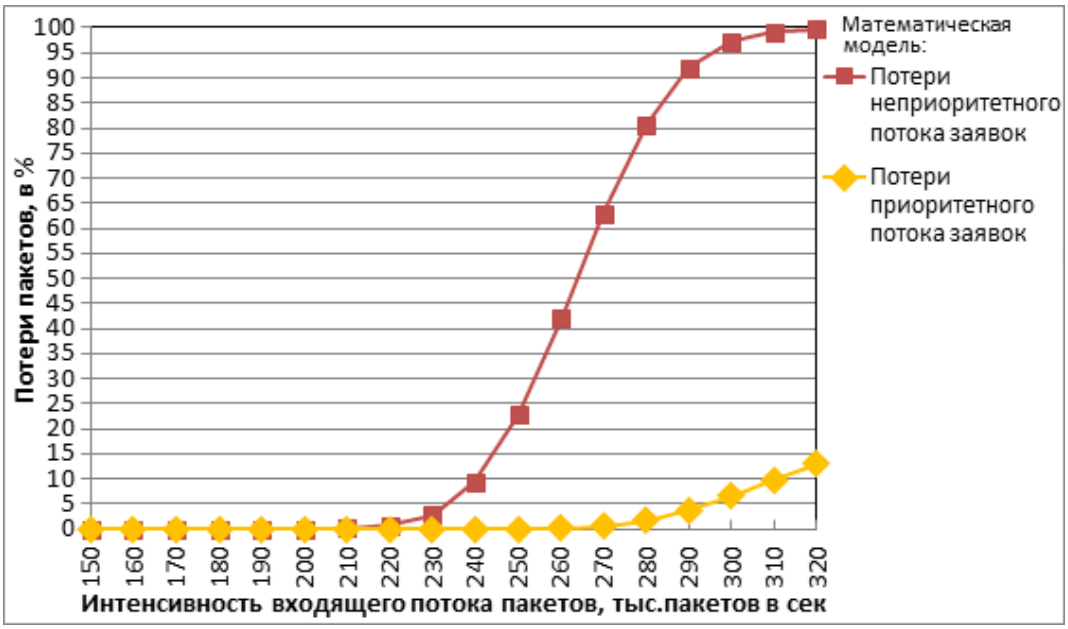


Рисунок 3.5 – Потери пакетов (2-ой сценарий)

Рисунок 3.6 демонстрирует динамику заполнения очереди в зависимости от интенсивности поступающего потока пакетов.

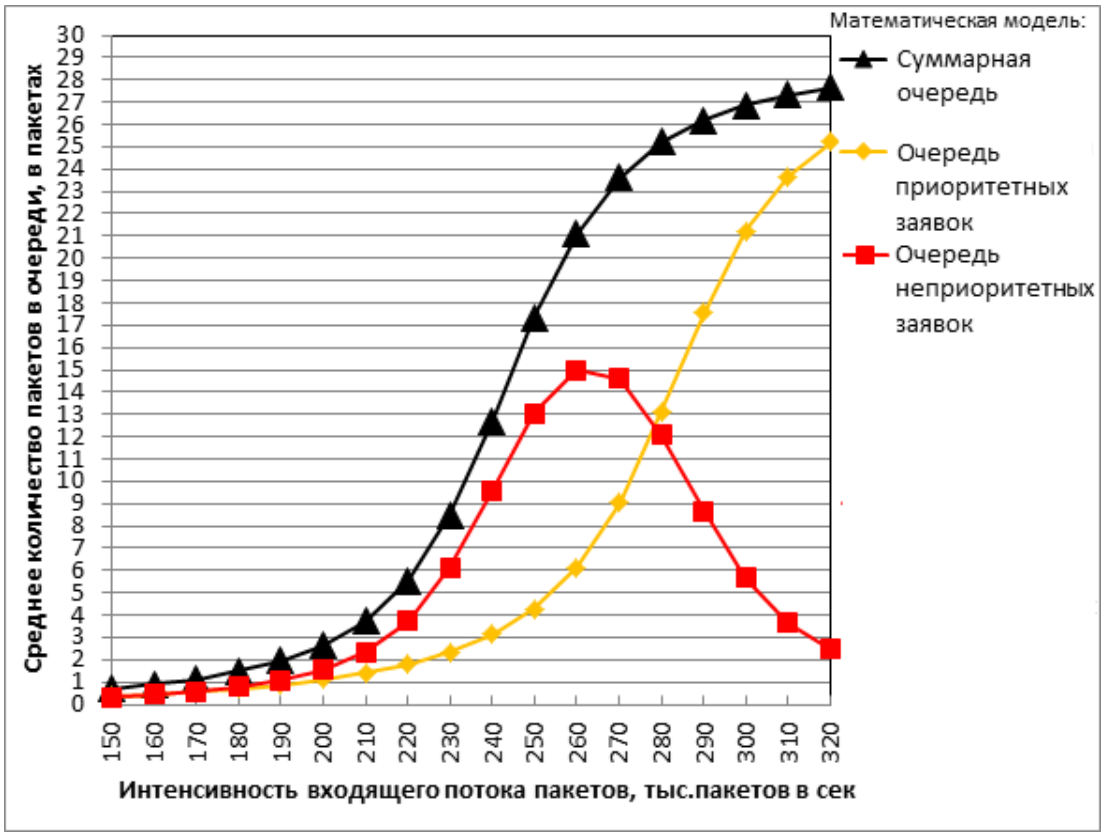


Рисунок 3.6 – Заполнение пакетной очереди (2-ой сценарий)

Рисунки 3.4, 3.5 и 3.6 наглядно демонстрируют, что большой поток приоритетных пакетов вытесняет неприоритетные пакеты из СМО, что ведёт к росту их потерь вплоть до 99.7888%.

Из рисунка 3.7 видно, что при росте интенсивности входящего потока приоритетных пакетов в очереди начинают копиться неприоритетные пакеты. Это происходит ввиду того, что приоритетные пакеты обслуживаются первыми и скапливаются в очереди по мере роста интенсивности входящего потока пакетов. Неприоритетные пакеты начинают скапливаться в очереди, даже при учёте того, что интенсивность их входящего потока не меняется, по причине того, что они не могут попасть на обслуживание и покинуть СМО. Когда интенсивность входящего потока приоритетных пакетов достигает точки отказа, происходит полное прекращение обслуживания неприоритетных пакетов, сопровождающееся их вытеснением из очереди.

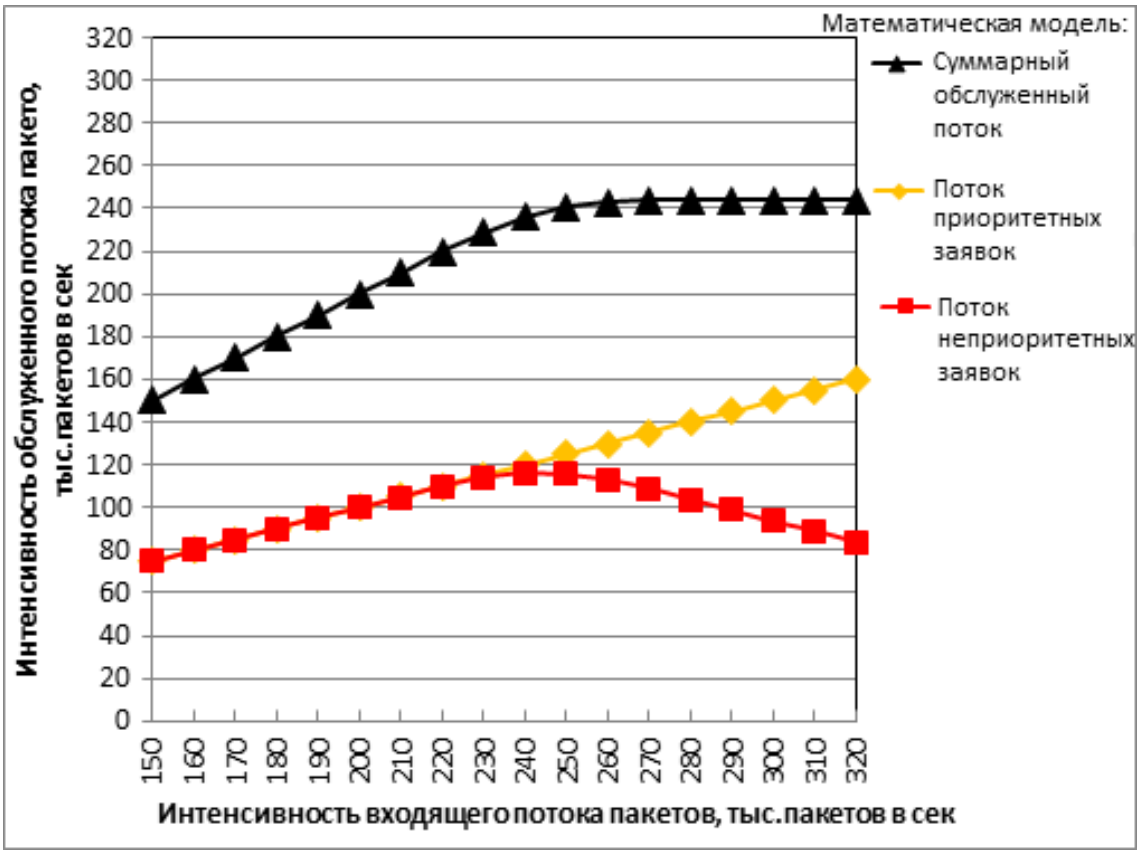


Рисунок 3.7 – Интенсивность обслуженного трафика (3-ий сценарий)

Третий сценарий

Для третьего сценария дополнительные исходные данные по входящим потокам трафика выглядят следующим образом $\lambda_1 = \lambda_2 = 75 \cdot 10^3,\ 80 \cdot 10^3,\ 85 \cdot 10^3, \ldots, 160 \cdot 10^3$.

Рисунок 3.7 демонстрирует динамику изменения интенсивности обслуженного потока пакетов в зависимости от интенсивности поступающего потока пакетов. Из рисунка 3.7 видно, что до достижения точки отказа оба входящих потока пакетов обслуживаются полностью, то есть без потерь. При приближении к точке отказа поток приоритетных пакетов начинает вытеснять неприоритетные пакеты из СМО. Если продлить график до того момента, когда интенсивность потока приоритетных пакетов достигнет точки отказа, то сценарий 3 вырождается в сценарий 2.

Рисунок 3.8 демонстрирует динамику изменения потерь в зависимости от интенсивности поступающего потока пакетов.

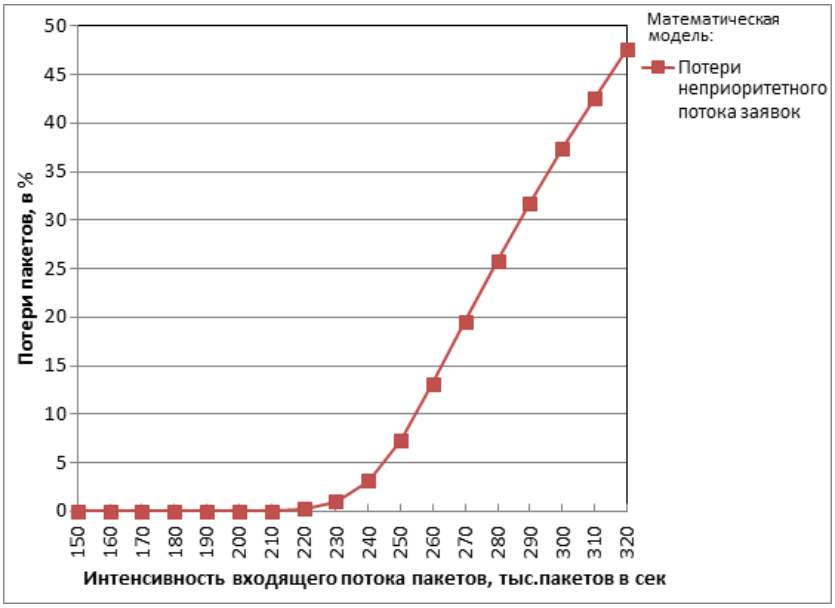


Рисунок 3.8 – Потери пакетов (3-ий сценарий)

Из рисунка 3.8 видно, что до достижения точки отказа оба входящих потока пакетов обслуживаются полностью, то есть без потерь. При приближении к точке отказа начинают появляться потери неприоритетных пакетов. До достижения точки отказа суммарным входящим потоком, динамика заполнения очереди схожа со сценарием 1.

Рисунок 3.9 демонстрирует динамику заполнения очереди в зависимости от интенсивности поступающего потока пакетов.

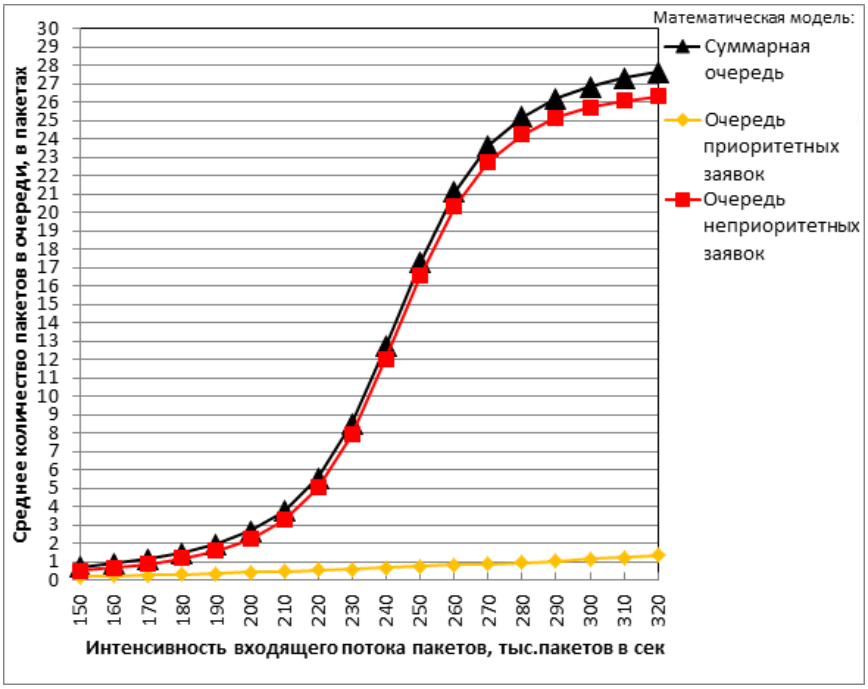


Рисунок 3.9 – Заполнение пакетной очереди (3-ий сценарий)

При дальнейшем росте интенсивности потока приоритетных пакетов очередь заполняется так же, как и при сценарии 2, это отражено на рисунке 3.9.

Краткий вывод

Применение приоритизации трафика позволяет снизить потери и величину очередь для приоритетного трафика. В отсутствие приоритизации обслуживания трафика потери стремились бы к одному и тому же процентному соотношению, а величина очереди – к соотношению интенсивностей входящих потоков пакетов.

По результатам раздела разработан математический аппарат, позволяющий получить численные значений показателей производительности МЭ, функционирующего в условиях приоритизации обслуживания трафика. Результаты раздела опубликованы в [117].

Выводы раздела

1. Входящие потоки (приоритетный/неприоритетный) приняты простейшими вследствие того, что они формируются при взаимном наложении большого числа малых независимых ординарных потоков с различным последействием. Это объясняется большим числом разнородных клиентских устройств в рамках рассматриваемых ЛВС торговых центров или бизнес-центров.

Функционирование межсетевого экрана в условиях приоритизации трафика может быть представлено в виде однолинейной системы массового обслуживания типа $\overrightarrow{M} / M / 1 / L / f_1^{22}$ с ограниченным накопителем пакетов, комбинированными потерями и смешанными приоритетами обслуживания. Диаграмма состояний и переходов представляет собой трёхмерную структуру, формируемую тремя степенями свободы: количеством приоритетных пакетов в очереди, количеством неприоритетных пакетов в очереди и порядковым номером операции, выполняемой в данный момент над обслуживаемым пакетом.

Для решения СУР (1)-(23) трёхмерной марковской модели функционирования межсетевого экрана разработана процедура перевода трехмерной матрицы переходных вероятностей в двумерную матрицу, позволяющая применить эффективный вычислительный метод расчёта

стационарных вероятностей, основанный на применении блочных треугольных разложений.

2. В качестве среды (программы) компьютерной алгебры и математического моделирования использована Wolfram Mathematica 10.0.1.0 ввиду удобства отладки модели с использованием продвинутого аппарата символьных вычислений.

3. Параметры межсетевого экрана, выбранные в качестве исходных данных, близки к реальным и заимствованы из работы специалистов IEEE [20], которые проводили стендовые испытания и измерения параметров обслуживания межсетевого экрана типа IPTables/NetFilter, рассматриваемого в рамках настоящей модели. Количество правил фильтрации, которое необходимо пройти пакету, подобрано для варианта с расстановкой правил внутри базы правил на основе статистики их совпадения (не идеальной). Параметры использованы в рамках дельнейших расчётов, на их основе произведено представление результатов.

РАЗДЕЛ 4. РАЗРАБОТКА ИМИТАЦИОННОЙ МОДЕЛИ ФУНКЦИОНИРОВАНИЯ МЕЖСЕТЕВОГО ЭКРАНА В УСЛОВИЯХ ПРИОРИТИЗАЦИИ ОБСЛУЖИВАНИЯ ТРАФИКА

4.1. Постановка задачи на разработку имитационной модели

4.1.1. Содержательная постановка задачи на разработку имитационной модели

Имитационная модель воспроизводит процесс обслуживания, принятый в математической модели (раздел 3) и расширяет функционал модели.

На вход имитационной модели поступает два потока пакетов: приоритетный и неприоритетный с интенсивностями λ_1 и λ_2. Каждый поток состоит из пакетов, время между приходом которых может быть постоянной величиной или случайной величиной, распределённой по экспоненциальному или равномерному законам. Потоки можно лишать свойства приоритетности, в этом случае один из них не будет иметь приоритета над вторым. Доступны следующие вариации дисциплин обслуживания:

– постановка пакетов в очередь может осуществляться с абсолютным приоритетом приоритетных пакетов или в соответствии с дисциплиной FIFO, то есть без приоритета постановки в очередь.

– пакеты могут поступать из очереди на обслуживание с относительным приоритетом приоритетных пакетов или в соответствии с дисциплиной FIFO, то есть без приоритета постановки на обслуживание.

Время обслуживания пакета может быть постоянной величиной или случайной величиной, распределённой по экспоненциальному или равномерному законам.

Имитационная модель позволяет воспроизводить полученные результаты моделирования за счёт свойства детерминированности генераторов псевдослучайных чисел, которые далее буду именовать *генераторами случайных чисел* (ГСЧ).

Все величины, зависящие от времени, в имитационной модели имеют размерность на абстрактную единицу времени. Это подразумевает, что пользователь имитационной модели должен вводить значения в одинаковой размерности, например, все временные характеристики в наносекундах или в микросекундах и т.п. Выходные величины будут иметь ту же размерность, что и входные величины. Такой подход усложняет работу пользователя, но позволяет не использовать внутреннего приведения величин к единой размерности, что положительно сказывается на точности вычислений.

Пакет, поступающий из очереди на обслуживание, продолжает занимать место в очереди до окончания обслуживания. Это имитирует занятие пакетом памяти, выделенной под очереди, в течение процесса его обработки. По окончании процесса обслуживания пакет покидает очередь и СМО в целом.

Методы реализации имитационной модели в коде, обеспечивают высокий уровень масштабируемости проекта, что, в свою очередь, позволяет добавлять и расширять функционал имитационной модели без значительных изменений существующей архитектуры проекта.

4.1.2. Стадии разработки имитационной модели

Выделим следующие стадии разработки имитационной модели:

- постановка задачи, поиск способа реализации имитационной модели:
  - разработка постановки задачи: Мусатов В.К.;
  - консультации: проф. Пшеничников А.П., аспирант Щербанская А.А.;
- разработка алгоритма имитационной модели: Мусатов В.К.;
- воплощение имитационной модели в программном коде:
  - ведущий разработчик: инженер-программист Аликин С.С.;
  - разработчик: Мусатов В.К. (участие в разработке механизмов сбора и анализа статистики, графического интерфейса пользователя, механизмов обработки событий, а также внедрении ГСЧ);
- поиск ошибок, отладка алгоритма, корректировка кода:

- разработка тестов, поиск ошибок, откладка, корректировка «периферийного» кода модели: Мусатов В.К.;
- корректировка кода модели: инженер-программист Аликин С.С.;

– проведение имитационного моделирования: Мусатов В.К.;

– сравнительный анализ результатов имитационного и математического моделирования, агрегация результатов моделирования:

- выполнение сравнительного анализа, агрегация результатов: Мусатов В.К.;
- консультации: проф. Пшеничинов А.П., Щербанская А.А.;

– выводы по результатам имитационного моделирования:

- разработка выводов: Мусатов В.К.;
- консультации: проф. Пшеничинов А.П.

4.1.3. Вводимые в имитационную модель исходные данные

Исходные данные, вводимые в имитационную модель, представлены в таблице 4.1.

Таблица 4.1 – Исходные данные для имитационной модели

Наименование параметра	**Тип данных, Возможное значение**	**Единица измерен.**
Параметры входящих потоков пакетов		
Интенсивность приоритетного входящего потока пакетов	Double [0;MaxValue]	пакет в условную единицу времени (УЕВ)
Интенсивность неприоритетного входящего потока пакетов	Double [0;MaxValue]	пакет в УЕВ
Параметр отключение приоритизации трафика. Если параметр равен 0, то будет приоритетный и неприоритетный потоки пакетов. Если параметр равен 1, то оба потока будут иметь одинаковый приоритет постановки в очередь и на обслуживание (дисциплина FIFO).	Bool [0,1]	–

Наименование параметра	Тип данных, Возможное значение	Единица измерен.
Параметры МЭ, влияющие на его производительность		
Среднее время, затрачиваемое на действия над пакетом, предшествующие процессу фильтрации	Double [0;MaxValue]	УЕВ
Среднее время, затрачиваемое на процесс проверки пакета одним правилом фильтрации	Double [0;MaxValue]	УЕВ
Количество правил фильтрации	Integer [0;MaxValue]	правило
Размер пакетной очереди (буфера памяти, выделяемого на работу очередей)	Integer [0;MaxValue]	пакет
Общие параметры работы ГСЧ		
Закон распределения вероятностей ГСЧ	ENUM [0,1] Равномерный, показа-тельный	–
Алгоритм работы ГСЧ	См. пункт 4.3.4	–
Случайное стартовое число ГСЧ (англ. Seed)	Integer [0;MaxValue]	–
Параметры ГСЧ с экспоненциальным распределением		
Расчёт времени прихода пакетов с помощью ГСЧ. Если параметр отключен, то время прихода пакетов будет постоянной величиной.	Bool [0,1]	–
Расчёт времени обслуживания пакетов с помощью ГСЧ. Если параметр отключен, то время обслуживания пакетов будет постоянной величиной.	Bool [0,1]	–
Параметры ГСЧ с равномерным распределением		
Процент отклонения значений равномерного ГСЧ для времени прихода пакетов. При выставлении значения отклонении равным 0%, время прихода пакетов будет постоянной величиной.	Double [0;100]	процент
Процент отклонения значений равномерного ГСЧ для времени обслуживания пакетов. При выставлении значения отклонении равным 0%, время обслуживание пакетов будет постоянной величиной.	Double [0;100]	процент
Параметры моделирования		
Модельное время	Double [0;MaxValue]	УЕВ

Наименование параметра	Тип данных, Возможное значение	Единица измерен.
Количество реализаций/объём выборки	Integer [0;MaxValue]	штук
Уровень доверия для расчёта доверительного интервала	Double [0;100]	процент
Значение t-критерия Стьюдента	Double [0;MaxValue]	–

4.1.4. Выводимые имитационной моделью результаты

Все результаты моделирования рассчитываются и выводятся отдельно для приоритетных и неприоритетных потоков пакетов за исключением общего количества пакетов, прибывших на вход модели, и количества неприоритетных пакетов, вытесненных из очереди приходом приоритетного пакета.

Результаты имитационного моделирования, выводимые при окончании расчёта, представлены в таблице 4.2.

Таблица 4.2 – Результаты имитационного моделирования

Наименование результата	Тип данных, Возможное значение	Единица измерен.
Количественные характеристики прихода пакетов:		
Количество приоритетных/неприоритетных пакетов, пришедших в систему	Integer [0;MaxValue]	пакет
Общее количество пакетов, пришедших в систему	Integer [0;MaxValue]	пакет
Количественные характеристики обслуживания пакетов		
Количество приоритетных/неприоритетных пакетов, обслуженных с ожиданием в очереди	Integer [0;MaxValue]	пакет
Количество приоритетных/неприоритетных пакетов, обслуженных без ожидания в очереди	Integer [0;MaxValue]	пакет
Количество приоритетных/неприоритетных пакетов, покинувших СМО, обнаружив очередь переполненной	Integer [0;MaxValue]	пакет
Количество неприоритетных пакетов, вытесненных из очереди приходом приоритетных пакетов	Integer [0;MaxValue]	пакет
Количество приоритетны/неприоритетных пакетов, присутствующих в очереди и на обслуживании на момент истечения модельного времени	Integer [0;MaxValue]	пакет
Среднее количество приоритетных/неприоритетных пакетов в очереди	Double [0;MaxValue]	пакет

Наименование результата	Тип данных, Возможное значение	Единица измерен.
Процент потерь приоритетных/неприоритетных пакетов	Double [0;100]	процент
Временные характеристики обслуживания		
Среднее время нахождения приоритетны/неприоритетных пакетов в очереди	Double [0;MaxValue]	УЕВ
Среднее время нахождения обслуженных приоритетны/неприоритетных пакетов в очереди для всех обслуженных пакетов	Double [0;MaxValue]	УЕВ
Среднее время нахождения обслуженных приоритетны/неприоритетных пакетов в очереди только для пакетов, задержанных в очереди	Double [0;MaxValue]	УЕВ
Среднее время обработки приоритетны/неприоритетных пакетов (время в очереди не учитывается, отражает работу ГСЧ)	Double [0;MaxValue]	УЕВ

4.1.5. Выбор способа реализации имитационной модели

Для воспроизведения принятого процесса функционирования МЭ необходимо выбрать общий подход к построению имитационной модели.

Три основных подхода к имитационному моделированию [33, 119, 120]:

– агентное моделирование;

– дискретно-событийное моделирование;

– системная динамика.

Наилучшим подходом для реализации поставленной задачи является дискретно-событийное моделирование вследствие того, что перечень возможных событий, возникающих в рамках процесса моделирования функционирования МЭ, ограничен, а появление событий подчиняется строгим логическим правилам. Эти факты позволяют привязать события к определённому модельному времени и чётко понимать, как и когда будет реагировать имитационная модель на различные события, тем самым обеспечивая детерминированность процесса обслуживания.

Возможность приобретения платных сред моделирования в рамках проведения настоящей работы отсутствует, в связи с чем была рассмотрена одна из наиболее популярных условно бесплатных средств моделирования,

используемых для симуляции процессов, протекающих в СМО – *GPSS World* (англ. General Purpose Simulation System, рус. система моделирования общего назначения). Попытка использовать GPSS для выполнения имитационного моделирования была неудачной вследствие того, что реализовать принятую в разделе 3 модель обслуживания не удалось. Также сказывается низкая степень знания этой среды моделирования. Что также стало причиной отказа от моделирования с использованием Network Simulator 2 и 3 (NS2, NS3).

По этим причинам было принято решение использовать языки программирования общего назначения для разработки имитационной модели МЭ. Учитывая, что планировалось разрабатывать имитационную модель для запуска в среде окружения Microsoft Windows, и отсутствует необходимость в поддержке кроссплатформенности имитационной модели, решено использовать в качестве языка программирования C#. В рамках разработки имитационной модели использование C# предоставляет следующие преимущества:

– упрощённый синтаксис C# по сравнению с C и C++ понижает шанс допустить алгоритмические ошибки в принципиально сложном коде;

– высокая репрезентативность кода упрощает анализ исходных кодов сторонними разработчиками и пользователями;

– упрощение разработки и отладки приложения за счёт применения в C# «*управляемого кода*» (англ. managed code) [121], основными достоинством которого являются современные методы управления памятью, улучшенная безопасность;

– поддержка «*сборки мусора*» (англ. garbage Collection) [122] байт-кодом, в который компилируются исходные коды C#.

4.2. Разработка алгоритма имитационной модели

4.2.1. Дискретно-событийное моделирование: особенность работы с событиями и временными метками

В качестве основы для разработки алгоритма дискретно-событийной модели использовалась информация из источников [119, 120 123].

Процесс дискретно-событийного моделирования представляется как хронологическая последовательность событий. Каждое событие происходит в определенный момент времени и приводит к изменению состояния модели, а также может приводить к появлению новых событий.

Для обеспечения статистической достоверности результатов необходимо многократное повторение процесса функционирования МЭ при заданных условиях и случайных стартовых значениях ГСЧ (англ. generator seed). Каждое такое повторение будем называть реализацией. Одна реализация длиться от нуля и до значения «модельного времени» – T_{model}, которое выставляется пользователем. Если обратиться к модели в произвольный момент времени в процессе моделирования, то она будет находиться в каком-то состоянии, в которое она попала при наступлении события, произошедшего на отрезке $[0\ ;\ T_{\text{model}}]$. Время появления этого события будем называть «текущим модельным временем».

Основной сложностью разработки алгоритма является совокупность двух факторов: первый – возможность обработки пакетов из очереди, второй – непрерывность процесса обработки пакета, занимающего некоторый временной интервал. В течение обработки пакета в систему могут приходить другие пакеты, чей приход может изменить состояние очереди, породить другие события и повлиять на выбор следующего пакета, отправляющегося из очереди на обработку. Основным принципом дискретно-событийного моделирования является поочерёдное наступление событий, что приводит к необходимости создания временных меток, относительно которых будет существовать возможность, продолжая обработку текущего пакета, реагировать на другие события, возникающие в модели до окончания его обработки. Важно понимать, что обработка пакета не прерывается.

Все события, которые могут возникнуть в течение обработки какого-либо пакета, порождаются приходом пакета в систему, и как следствие, приход пакета принят за временную метку наравне с истечением модельного

времени. Рассмотрим пример диаграммы состояний модели (рисунок 4.1), иллюстрирующую обработку событий с применением временных меток

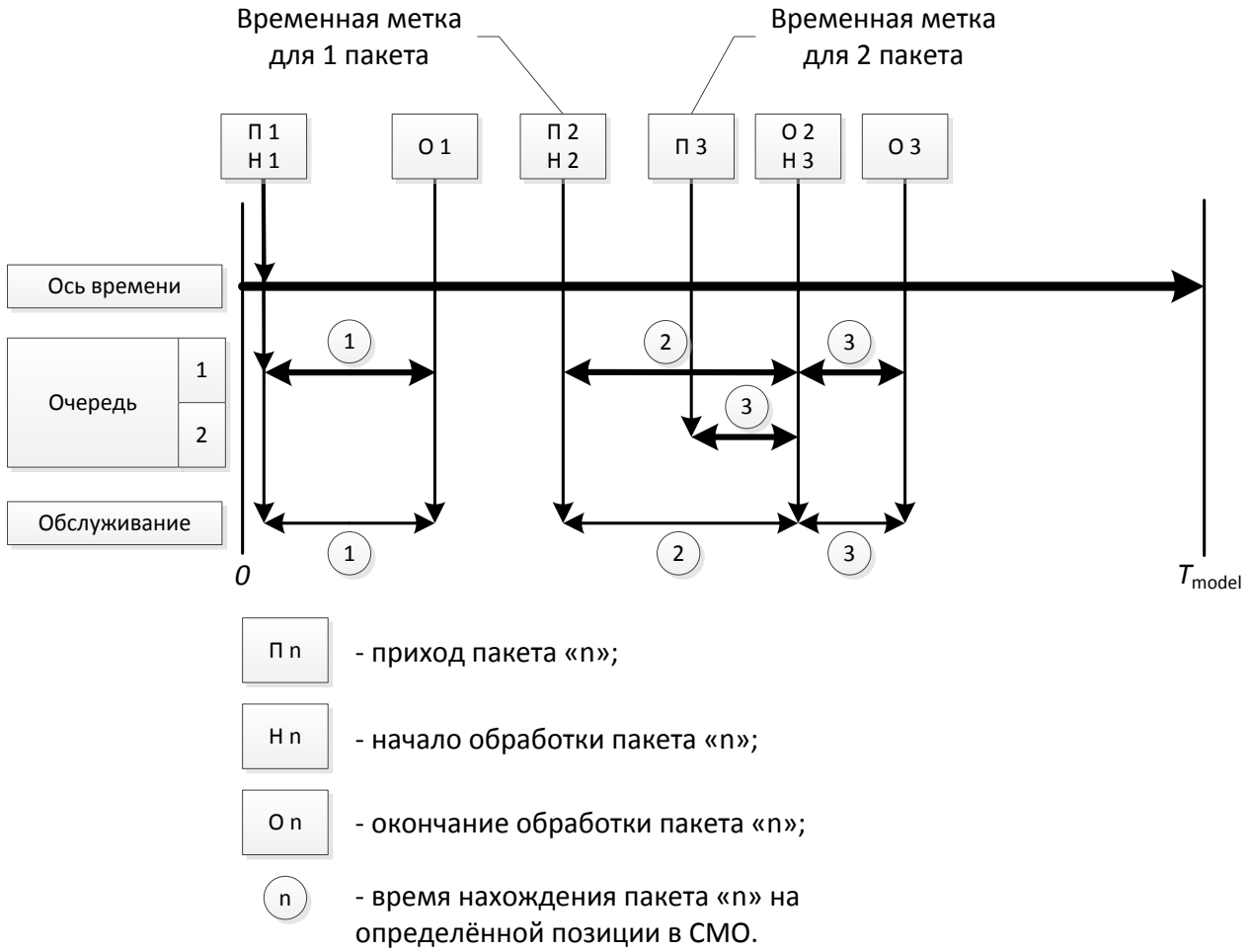


Рисунок 4.1 – Иллюстрация работы временных меток

На рисунке отражены ось времени $[0\ ;\ T_{\text{model}}]$, очередь с двумя позициями, напротив которых стрелками указаны интервалы времени, которые проводят пакеты на определённых позициях очереди, а также предусмотрено место для линий, отражающих продолжительность обслуживания пакетов. Событие с маркировкой «П 1» означает приход первого пакета. Событие «Н 1» означает начало обслуживание первого пакета. Событие «О 1» означает окончание обслуживание первого пакета. Другие события маркированы по аналогии.

При поступление в модель первого пакета наступают событиях «П 1», «Н 1». Пакет попадает в очередь и сразу поступает на обслуживание, однако, система не знает, успеет ли она обслужить первый пакет до поступления второго.

По этой причине необходимо выработать время прихода второго пакета «П 2» и считать его временной меткой.

Рассчитав время окончания обработки первого пакета, проверяем, вышло ли оно за временную метку «П 2». Не обнаружив проблем, заканчиваем обработку первого пакета событием «О 1». Далее система простаивает до наступления события «П 2». Когда приходит второй пакет, он помещается в очередь и появляется необходимость создания новой временной метки. Создаётся третий пакет «П 3» и соответствующая ему временная метка. Начинается обслуживание второго пакета, и при этом обнаруживается, что до окончания обслуживания в модель придёт третий пакет. До завершения обслуживания второго пакета модель обрабатывает событие «П 3», помещая пакет в очередь на позицию 2. После чего продолжает обслуживание второго пакета. Дальнейшая обработка ведётся по аналогии.

Любой пакет, который уже попал в систему к заданному моменту времени, будет называться «попавшим в систему». Пакет, играющий роль временной метки, будет называться «ожидающим прихода».

Данная схема иллюстрирует реакцию системы при определённых условиях, что достаточно для общего понимания принципов обработки событий с использованием временных меток.

4.2.2. Алгоритм функционирования имитационной модели

Алгоритм функционирования имитационной модели, представленный в данном пункте, является обобщённым и предоставляет возможность воспроизвести его на других языках программирования. Такой тип алгоритмов в некоторых источниках называют «моделирующим алгоритмом». Помимо блоков алгоритма, отражающих принятый процесс обслуживания, в алгоритм включены блоки ввода/вывода данных, контроля количества выполненных реализаций и подсчёта статистики.

Условно разделим алгоритм на 3 цикла: базовый, внешний и внутренний. В рамках каждого цикла выполняется несколько уникальных задач. Внутренний

цикл алгоритма (рисунок 4.2) предназначен для имитации обработки пакетов и работы с очередью. Базовый цикл алгоритма (рисунок 4.3) предназначен для ввода/вывода данных, выполнения заданного количества реализаций, подсчёта результатов моделирования. Внешний цикл алгоритма (рисунок 4.4) предназначен для генерации пакетов, установки и контроля временных ограничителей (меток времени), а также работы с очередью.

Элементы алгоритма, имеющие затемнённый фон, обозначают старт программы (базовый цикл), внешнего и внутреннего циклов. Круглые метки с цифровыми обозначениями отмечают блок-схемы, после выполнения которых производится инкрементация различных счётчиков и переменных, используемых для сбора статистической информации.

После запуска модели производится ввод исходных данных и запуск процесс моделирования по команде пользователя. После запуска процесса моделирования производится однократная очистка всех переменных, действующих внутри модели.

Далее выполняется структура базового цикла, внутри которого происходит управление реализациями и подсчёт статистики по результатам каждой реализации. Затем устанавливается значение первой реализации, впоследствии при возврате к этому блоку число реализаций каждый раз будет инкрементироваться на 1 до достижения заданного количества реализаций. Счётчик реализаций инкрементируется после прохождения блока, помеченного «1» на рисунке 4.3.

В начале каждой реализации необходимо очистить все переменные, действующие в рамках одной реализации (локальные переменные) и очередь. При расчёте с сохранением состояния очереди, необходимо восстановить её состояние, а по её состоянию выставить счётчики в базовые значения. Результаты реализации необходимо хранить отдельно и очищаться после перезапуска процесса моделирования или при выключении программы моделирования.

Затем выполняется переход во внешний цикл, в котором моделируются процессы функционирования в рамках одной реализации. Чтобы упростить цикл в

первый проход по нему будет создан только «ожидающий прихода» пакет, который в конце цикла станет «попавшим в систему» в момент его постановки в очередь. Со второго прохода по внешнему циклу система попадёт в состояние «П 1», описанное ранее на примере в пункте 4.2.1. При каждом последующем проходе по внешнему циклу модели будет создаваться один «ожидающий прихода» пакет, выполняющий роль временной метки до истечения модельного времени.

Затем происходит переход во внутренний цикл для обработки пакетов. Сначала проверяется наличие пакета на обработке (обработка события «О 1», рассмотренное на примере в пункте 4.2.1), если он есть, то модель продолжит его обработку, если его нет, то обработчик обратится к очереди и попытается получить из неё пакет в соответствии с принятой дисциплиной обслуживания.

В случае если будет обнаружено, что до окончания обработки пакета должен прийти пакет «ожидающий прихода», то будет вызвано прерывание внутреннего цикла алгоритма, после чего произойдёт возврат к внешнему циклу. Модель же будет помнить о том, что на обработке уже находится пакет. Обработчик внутреннего цикла будет брать пакеты из очереди на обработку, пока не кончатся пакеты в очереди, или не будет достигнута временная метка. После выполнения блока, помеченного «4», инкрементируется счётчик обслуженных пакетов, в зависимости от приоритета обслуженного пакета, а также рассчитывается его временные характеристики.

При возвращении во внешний цикл алгоритма будут производиться следующие операции: проверка на окончание реализации по модельному времени, попытка помещения пакета «ожидающего прихода» в очередь (после этот пакет станет «попавшим в систему»). Если модельное время не истекло, то модель попытается поместить «ожидающий прихода» пакет в очередь в соответствии с принятой дисциплиной постановки в очередь и внешний цикл повторится.

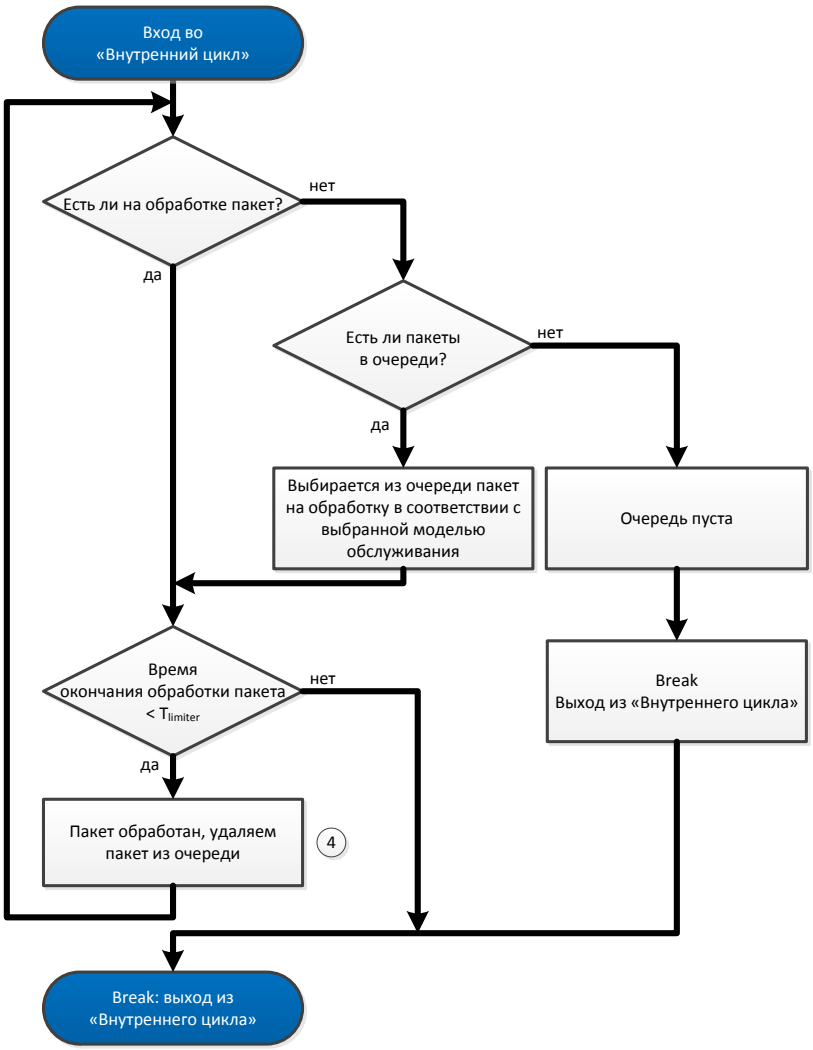


Рисунок 4.2 – Внутренний цикл

Если модельное время истекло, то будет вызвано прерывание внешнего цикла и система вернётся в базовый цикл алгоритма.

После получения результата от попытки поместить пакет в очередь (блок помечен «2») необходимо произвести инкрементацию счётчиков в зависимости от совершённых действий. Первым инкрементируется счётчик пришедших в систему пакетов (приоритетных или неприоритетных), в зависимости от действий, произведённых при постановке в очередь, могут быть инкрементированы счётчики потерянных пакетов или вытесненных неприоритетных пакетов при приходе приоритетного пакета (блок помечен «3»).

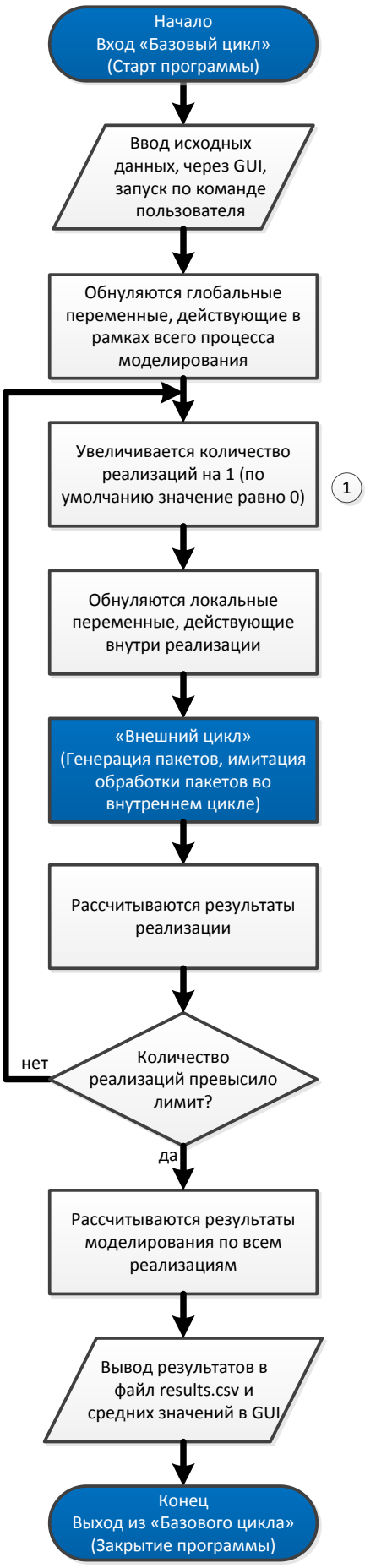


Рисунок 4.3 – Базовый цикл

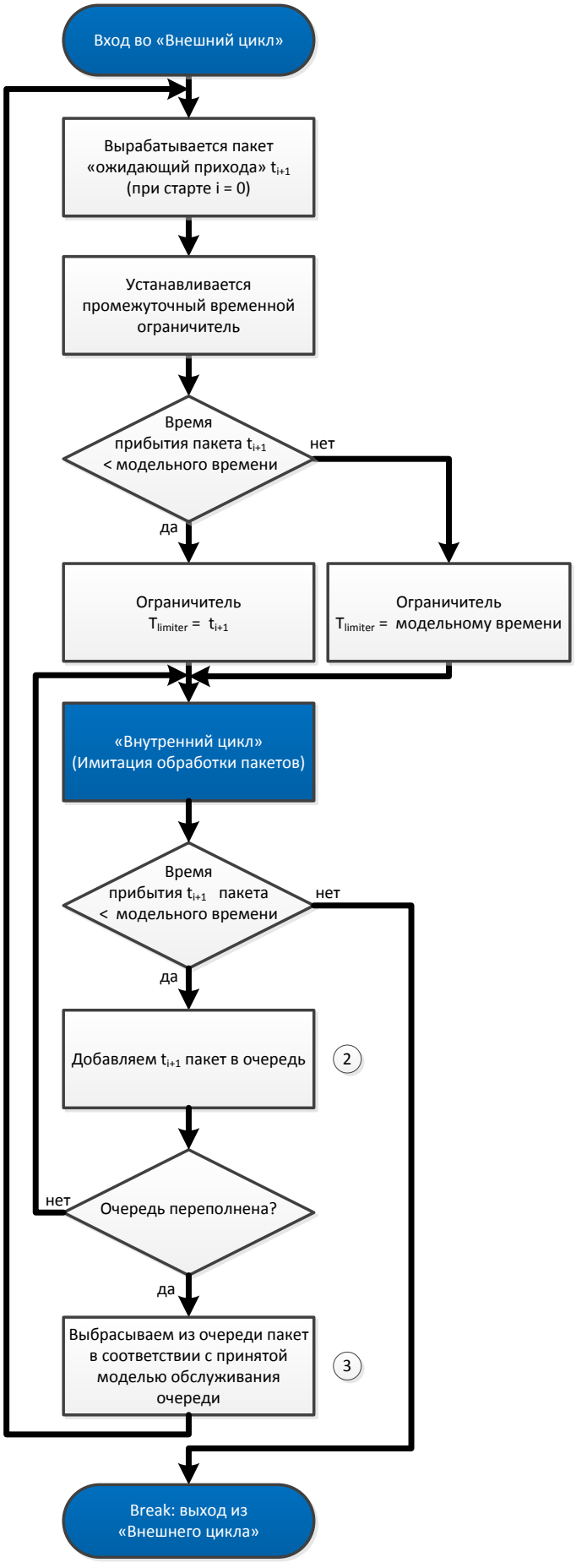


Рисунок 4.4 – Внешний цикл

При потерях пакетов также необходимо рассчитать их время нахождения в очереди и инкрементировать соответствующие переменные, хранящие временные характеристики (пункт 4.3.4).

Возврат в базовый цикл алгоритма происходит при истечении модельного времени одной из реализаций, что потребует рассчитать результаты реализации. Если заданное количество реализаций не достигнуто, то система вернётся к циклической структуре базового цикла и начнёт моделирование следующей реализации. При достижении заданного количества реализаций будет произведён расчёт статистики по всем реализациям. По окончании расчёта производится вывод результатов.

Важным моментом является ввод модели в стационарный режим. Вследствие того, что разработка «автоматического» алгоритма проверки вхождения в стационарный режим достаточно сложная задача, был выбран более простой подход. Для ввода в стационарный режим необходимо провести расчёт и по результатам оценить вошла ли модель в стационарный режим, если вошла, то не изменяя параметры модели необходимо провести расчёт на тех же исходных данных с переносом состояния очереди из предыдущего расчёта. Как результат модель сразу будет иметь заполненную очередь, фактически находясь в стационарном режиме.

Алгоритм, представленный в таблице 4.3, предусматривает выполнение некоторого комплекса операций над пакетами.

Таблица 4.3 – Операции, выполняемые над пакетом

Наименование выполняемой операции	**Приоритет пакета**
Постановка пакета в очередь	Приоритетный Неприоритетный
Отказ в постановке пакета в очередь в связи с переполнением очереди	Приоритетный Неприоритетный
Отказ в постановке пакета в очередь в связи с переполнением очереди и невозможностью освобождения места в очереди	Приоритетный
Исключение пакета из очереди для освобождения места для приоритетного пакета	Неприоритетный

Наименование выполняемой операции	Приоритет пакета
Поиск пакета в очереди для его передачи на обслуживание	Приоритетный Неприоритетный
Постановка пакета на обслуживание	Приоритетный Неприоритетный
Исключение пакета из очереди после окончания обслуживания	Приоритетный Неприоритетный

Описанный комплекс операций позволяет полностью реализовать принятую модель обслуживания.

4.3. Реализация алгоритма имитационной модели

4.3.1. Общая информация

Имитационная модель реализована на языке C# в среде Microsoft Visual Studio на основе Microsoft.NET 4.0.

В процессе разработки было решено обеспечить имитационную модель *графическим пользовательским интерфейсом* (англ. graphic user interface, GUI). Для пользовательского интерфейса использовалась платформа WPF 4.0. Она входит в Microsoft.NET 4.0, в дополнение при разработке пользовательского интерфейса использовался паттерн *MVVM* (model – view – view model).

Для реализации ГСЧ использовались штатная библиотека System.Random, входящая в состав Microsoft.NET 4.0 и сторонняя библиотека Math.NET Numeric, распространяемая по лицензии MIT/X11. Информация о ГСЧ приведена в пункте 4.3.4.

При разработке применялся ряд паттернов проектирования, из которых стоит выделить следующие:

– *dependency injection* (рус. внедрение зависимостей);
– *abstract Factory* (рус. абстрактная фабрика);
– *command* (рус. команда);
– *facade* (рус. фасад);
– *flyweight* (рус. легковесный повторно используемый объект);
– *null object* (рус. нулевой объект);

– *object pool* (рус. пул объектов);

– *singleton* (рус. одиночка).

Описание задач, для которых применялись те или иные паттерны проектирования приведены в приложении Д.

Описание паттернов проектирования приведено в [124].

4.3.2. Сторонние библиотеки, применённые при разработке имитационной модели

При разработке имитационной модели использовались сторонние библиотеки. Все использованные библиотеки являются свободно распространяемыми.

Применены библиотеки двух типов: первый – библиотеки Microsoft в том числе: *стандартные библиотеки классов* (англ. base class library) .NET Framework и Microsoft.Practices.Unity, второй – библиотеки прочих разработчиков.

На основе стандартных классовых библиотек реализуется основная часть имитационной модели. По этой причине не будут выделяться их конкретные роли в рамках проекта.

Библиотека Microsoft.Practices.Unity использована для реализации паттерна внедрения зависимостей.

В проекте использовано несколько библиотек сторонних разработчиков: NLog [125] для ведения журналов событий и MathNet.Numerics [126] для ГСЧ и ИСЧ, из состава которой используется два пространства имён:

– MathNet.Numerics.Random;

– MathNet.Numerics.Distributions.

4.3.3. Типы используемых данных

В имитационной модели используются следующие типы данных:

– bool: используется для операций математической логики, диапазон данных [0 ; l], информация об этом типе приведена в [127];

– Integer 32: диапазон данных [–2147483684 ; 2147483647], хранит целочисленные переменны, информация об этом типе данных в [128];

– ENUM: диапазон значений задаётся пользователем, по умолчанию хранится в переменной Integer, информация о этом типе [129];

– Double: диапазон данных $\pm 1.7 \cdot 10^{308} ... \pm 5.0 \cdot 10^{-324}$ в зависимости от точности, составляющей 15-16 знаков, хранит 64 разрядные значения [130];

– Long: используется для целочисленных переменных, которые не попадают в диапазон Integer 32, покрывает диапазон [–9223372036854775808 ; 9223372036854775807], хранит 64 разрядные значения [131];

– String: используется для текстовых данных при вводе/выводе данных и текста в GUI. Вводимые исходные данные конвертируются в типы данных, описанные выше [132].

В рамках разработки было проведено сравнение результатов расчётов при использовании форматов double и decimal для хранения нецелочисленных переменных с плавающей запятой. Значение переменной в формате decimal занимает вдвое больше памяти, чем в формате double, при этом имеет меньший диапазон значений $[-7.9 \cdot 10^{-28};\ 7.9 \cdot 10^{28} / (10^{0-28})]$, но большую точность расчётов.

При проведении тестов двух сборок программы моделирования (на основе double и decimal) не было выявлено значимых расхождений результатов моделирования. При этом ограниченный диапазон значений переменных, предоставляемый decimal, теоретически мог создать ошибку переполнения переменной. В дополнение к вышесказанному надо учесть, что переменные double обрабатываются заметно быстрее, чем переменные decimal. Современные процессоры не обладают встроенной поддержкой операций с числами формата decimal, это требует выполнения нескольких инструкций процессора даже для выполнения простейших операций сложения. По описанным причинам переменные типа decimal в модели не используются.

4.3.4. Внутренние переменные, структуры и их реализация

Пакеты. На вход модели поступают пакеты. Пакетам присвоены несколько свойств, представленных в таблице 4.4.

Таблица 4.4 – Свойства пакетов

Наименование свойства	**Тип данных**
Время прихода пакета	Double
Время обработки пакета	Double
Приоритет пакета	ENUM: 0 – неприоритетный пакет; 1 – приоритетный пакет.
Статус пакета	ENUM: 0 – пакет не обслужен; 1 – пакет ожидает в очереди; 2 – обслужен 3 – пакет сброшен из-за переполнения очереди; 4 – неприоритетный пакет вытеснен приходом приоритетного пакета.

Четыре свойства пакета охватывают все необходимые для моделирования состояния пакета кроме случая, при котором модельное время истекло и осталось некоторое количество пакетов в очереди и один на обслуживании, такие пакеты именуются *неучтёнными* (англ. unaccounted). Количество таких пакетов рассчитывается математически.

Очередь. Принятая модель обслуживания требует реализовать следующие операции с очередью:

– постановку в очередь на определённую позицию в соответствии с заданными критериями (приоритет, очерёдность прихода среди пакетов с одинаковым приоритетом);

– поиск пакета в очереди по заданным критериям;

– сброс пакета из очереди;

– передача пакета на обслуживание;

– перемещение пакета между позициями в очереди;

– определение размера очереди.

Для минимизации ресурсов на поддержание механизма функционирования очередей решено:

– разделить очередь на две структуры: приоритетную и неприоритетную очереди (с контролем общего объёма очереди);

– реализовывать обе очереди как динамическую структуру данных – *связный список* (англ. Linked list).

Длина связного списка – это характеристика, описывающая количество элементов, находящихся в нём. Например, если длина очереди равна 0, то значит, что в ней нет пакетов. Это позволяет не производить поиск пакетов внутри самой структуры, именуемой «очередь». Внутри каждой очереди пакеты обслуживаются по дисциплине FIFO, однако, из неприоритетной очереди периодически необходимо изымать последний пришедший пакет (сценарий прихода приоритетного пакета при переполнении очереди). По описанным причинам для реализации каждой из очередей не подходят структуры организации данных, называемые *очередь* (англ. queue) и *стек* (англ. stack).

Каждый раз при приходе пакета в очередь фиксируется количество пакетов, которое содержится в очереди на момент прихода пакета. Данная характеристика является статистически важной и рассматривается далее в этом пункте.

Генераторы случайных чисел. В имитационной модели реализованы ГСЧ, функционирующие по следующим алгоритмам: алгоритму Вихря Мерсенна (Mersenne Twister 19937 generator) [133] и модифицированному алгоритму Donald E. Knuth's, используемого в основе класса System.Random в языке C# с .Net Framework 1.1 [134, 135]. Алгоритм Вихря Мерсена реализован в заимствованной библиотеке MathNet.Numerics.

Экспоненциальный (показательный) закон распределения вероятностей реализован с помощью заимствованной библиотеки MathNet.Numerics. Равномерный закон распределения вероятностей реализован самостоятельно.

Сбор статистической информации и её агрегация. Первые нагрузочные тесты показали, что хранить всю статистическую информацию по каждому

объекту типа пакет невозможно из-за большого требуемого объёма памяти. При хранении информации о 50 000 000 экземпляров пакетов и промежуточной статистической информации, программа занимает у операционной системы около 8.5 Гбайт оперативной памяти. Для решения задачи сбора временных характеристик функционирования МЭ было решено использовать набор инкрементируемых переменных, например, после завершения обслуживания каждого пакета его время обслуживания прибавляется к некой переменной, которую в конце расчёта можно разделить на общее количество обработанных пакетов, получив среднее время обработки пакета. Набор таких переменных позволяет эффективно использовать оперативную память, ограниченную реалиями современных персональных ЭВМ. После того, как пакет покидает модель, его персональная информация удаляться. Персонифицированная информация о пакетах хранится, пока они находятся в модели. Подобный подход позволяет имитировать СМО с «бесконечной» очередью, при условии, что входящий поток пакетов превышает обслуживающую способность МЭ в разумных пределах и не произошло переполнение очереди.

Вторым элементом сбора статистики являются счётчики. Счётчики хранят в себе целочисленные данные такие, как количество пакетов, над которыми были совершены те или иные действия. Сбор статистики осуществляется с использованием переменных и счётчиков, описанных в таблице 4.5.

Таблица 4.5 – Переменные и счётчики, используемые для сбора статистики

Наименование переменной/счётчика	**Тип данных**	**Условное обозначение**
Количество пакетов, пришедших на вход модели:		
Общее количество пакетов	long	$Cincome$
Количество приоритетных пакетов	long	$Cincome_{priority}$
Количество неприоритетных пакетов	long	$Cincome_{unpriority}$
Количество обслуженных пакетов:		
Общее количество обслуженных пакетов	long	CPr
Количество обслуженных приоритетных пакетов	long	$CPr_{priority}$
Количество обслуженных неприоритетных пакетов	long	$CPr_{unpriority}$
Количество приоритетных пакетов, обслуженных с ожиданием в очереди	long	$CPrW_{priority}$

Наименование переменной/счётчика	Тип данных	Условное обозначение
Количество неприоритетных пакетов, обслуженных с ожиданием в очереди	long	$CPrW_{unpriority}$
Количество приоритетных пакетов, обслуженных без ожидания в очереди	long	$CPrWo_{priority}$
Количество неприоритетных пакетов, обслуженных без ожидания в очереди	long	$CPrWo_{unpriority}$
Количество пакетов, покинувших СМО:		
Общее количество пакетов, покинувших СМО, обнаружив очередь переполненной	long	$CLoss$
Количество приоритетных пакетов, покинувших СМО, обнаружив очередь переполненной	long	$CLoss_{priority}$
Количество неприоритетных пакетов, покинувших СМО, обнаружив очередь переполненной	long	$CLoss_{unpriority}$
Количество неприоритетных пакетов, вытесненных из очереди приходом приоритетного пакета	long	$CKnockout$
Количество пакетов, присутствующих в очереди и на обслуживании на момент истечения модельного времени:		
Количество приоритетных пакетов	long	$Cunacc_{priority}$
Количество неприоритетных пакетов	long	$CQunacc_{unpriority}$
Количество пакетов в очереди на момент осуществления постановки пакета в очереди:		
Количество приоритетных пакетов в очереди	long	$CQueue_{priority}$
Количество неприоритетных пакетов в очереди	long	$CQueue_{unpriority}$
Время нахождения пакетов в очереди:		
Время нахождения обслуженных приоритетных пакетов в очереди	double	$TQueue_{priority}$
Время нахождения обслуженных неприоритетных пакетов в очереди	double	$TQueue_{unpriority}$
Время нахождения в очереди неприоритетных пакетов вытесненных из очереди приходом приоритетного пакета	double	$TKnockout$
Время обработки пакетов:		
Время обработки приоритетных пакетов	double	$TPr_{priority}$
Время обработки неприоритетных пакетов	double	$TPr_{unpriority}$
Время обработки и нахождения пакетов в очереди:		
Время обработки и нахождения приоритетных пакетов в очереди	double	$TQP_{priority}$
Время обработки и нахождения неприоритетных пакетов в очереди	double	$TQP_{unpriority}$

Получив значения переменных, хранящих временные характеристики, и данные счётчиков, можно рассчитать средние значения временных показателей производительности МЭ, а также среднее количество пакетов в очереди.

Среднее количество пакетов в очереди рассчитывается по следующим формулам:

$$QueueAvr_{priority} = CQueue_{priority} / Cincome;$$

$$QueueAvr_{unpriority} = CQueue_{unpriority} / Cincome;$$

$$QueueAvr = QueueAvr_{priority} + QueueAvr_{unpriority}.$$

Среднее время нахождения пакетов в очереди на общее количество пришедших пакетов рассчитывается по следующим формулам:

$$TAvrQperPacket_{priority} = \frac{TQueue_{priority}}{Cincome_{priority} - Cunacc_{priority}};$$

$$TAvrQperPacket_{unpriority} = \frac{TQueue_{unpriority}}{Cincome_{unpriority} - Cunacc_{unpriority}}.$$

Среднее время нахождения обслуженных пакетов в очереди рассчитывается по следующим формулам:

$$TAvrQperServedPacket_{priority} = \frac{TQueue_{priority}}{CPr_{priority}};$$

$$TAvrQperServedPacket_{unpriority} = \frac{TQueue_{unpriority}}{CPr_{unpriority}}.$$

Среднее время нахождения обслуженных пакетов в очереди для пакетов, задержанных в очереди, рассчитывается по следующим формулам:

$$TAvrQperQueueServedPacket_{priority} = \frac{TQueue_{priority}}{CPrW_{priority}};$$

$$TAvrQperQueueServedPacket_{unpriority} = \frac{TQueue_{unpriority}}{CPrW_{unpriority}}.$$

Среднее время нахождения неприоритетного пакета в очереди (учитывает как обслуженные пакеты, так и вытесненные пакеты) рассчитывается по формуле:

$$TAvrQinSystem_{unpriority} = \frac{TQueue_{unpriority} + TKnockout}{CPr_{unpriority} + CKnockout}.$$

Среднее время обслуживания пакетов рассчитывается по следующим формулам:

$$TAvrProcessing_{priority} = \frac{TPr_{priority}}{CPr_{priority}};$$

$$TAvrProcessing_{unpriority} = \frac{TPr_{unpriority}}{CPr_{unpriority}}.$$

Величины среднего времени нахождения в очереди и обслуживания пакетов рассчитываются по следующим формулам:

$$TAvrQP_{priority} = \frac{TQP_{priority}}{CPr_{priority}};$$

$$TAvrQP_{unpriority} = \frac{TQP_{unpriority}}{CPr_{unpriority} + CKnockout}.$$

Процент потерь пакетов рассчитывается по следующим формулам:

$$Ploss_{priority} = \frac{CLoss_{priority}}{Cincome_{priority}} \cdot 100;$$

$$Ploss_{unpriority} = \frac{CLoss_{unpriority} + CKnockout}{Cincome_{unpriority}} \cdot 100.$$

4.3.5. Внедрение зависимостей

Внедрение зависимостей (англ. Dependency injection) [136] – процесс предоставления программному компоненту внешней зависимости. Процесс является специфичной формой «*инверсии управления*» (англ. Inversion of control, IoC), когда она применяется к управлению зависимостями. Для реализации паттерна внедрения зависимостей использовалась библиотека Microsoft Unity 2.0, а именно её IoC-контейнер Unity.

Внедрение зависимостей применяется при реализации пакетных фабрик, выполняющих задачу создания пакетов, и в ГСЧ. В обоих случаях используется паттерн типа abstract factory.

Применение внедрения зависимостей в пакетных фабриках. Пакетная фабрика создаёт объекты (пакеты), имеющие 4 свойства (таблица 4.4. в пункте 4.3.4). Значения трёх из четырёх свойств могут быть постоянными или случайными величинами, подлежащими созданию с использованием ГСЧ. Значение четвёртого свойства при создании нового пакета всегда является постоянной величиной. В результате имеем 8 различных комбинаций значений трёх свойств пакетов. Учитывая данный факт, было принято решение реализовать пакетную фабрику с использованием шаблона abstract factory, которая инициализируется на основе входных параметров модели.

Применение внедрения зависимостей при работе ГСЧ. В рамках процесса моделирования функционируют ГСЧ, реализующие два различных закона распределения вероятностей: равномерный и экспоненциальный.

ГСЧ, вырабатывающий приоритет пакета, реализует равномерный закон. ГСЧ, вырабатывающие время прихода и обработки пакетов, реализуют на выбор экспоненциальный или равномерный законы распределения вероятностей. Каждый тип ГСЧ требует определённых входных данных для своего функционирования. Для упрощения работы с различными типами ГСЧ, а также уменьшения количества различных методов, используемых для вызовов ГСЧ различных типов, используется паттерн внедрение зависимостей.

Основным преимуществом использование внедрения зависимостей является упрощение последующего добавление ГСЧ, реализующих другие законы распределения вероятностей за счёт обобщения метода вызова ГСЧ.

4.4. Проверка функционирования имитационной модели, сравнение результатов имитационного и математического моделирования

4.4.1. Способ проверки функционирования имитационной модели

Использование общепринятых методов и процедур при разработке математической модели позволяет утверждать, что полученные с её помощью результаты имеют высокую степень достоверности, но несут в себе погрешность вычислений. Факт совпадение результатов имитационного и математического моделирования (с достаточной точностью) доказывает, что имитационная модель функционирует корректно.

4.4.2. Исходные данные

Для проверки имитационной модели необходимо использовать те же параметры МЭ, которые применялись в математической модели (раздел 3).

Дополнительными параметрами имитационного моделирования являются:

1. Модельное время – 30 секунд. Время вхождения системы в стационарный режим зависит от соотношения входящих потоков пакетов и характеристик МЭ, влияющих на обслуживание пакетов. В нашем случае время вхождения системы в стационарный режим наступает в течение 1 – 5 секунд.
2. Количество реализаций – 30.
3. Уровень доверия к результатам – 95%.
4. Алгоритм работы ГСЧ – Mersenne Twister 1998.
5. Время между приходом пакетов и время обслуживания пактов является случайными величинами, распределёнными по экспоненциальному закону.

4.4.3. Доверительный интервал

В имитационной модели автоматически рассчитывается предел погрешности в соответствии с распределением Стьюдента и требует к вводу t-критерий Стьюдента и уровень доверия. По умолчанию они соответствуют 95% уровню доверия. Уровень доверия вводится для напоминания о принятом уровне доверия, в расчётах же используется t-критерий Стьюдента, подобранный для 95% уровня доверия при объёме выборки, равной 30.

Результаты измерений на указанных исходных данных имеют предел погрешности от 0,01 до 1 % от исследуемых средних значений показателей производительности. Это объясняется тем, что при заданных исходных данных в рамках одной реализации в систему поступает от 6,5 до 9,5 миллионов пакетов, а при создании каждого пакета происходит 3 обращения к ГСЧ. Как следствие, результаты внутри каждой реализации в значительной степени усредняются, а предел погрешности становиться мал. По описанным причинам доверительный интервал не представлен на графиках.

4.4.4. Сравнение результатов моделирования

Сравнение производится для двух сценариев поведения трафика.

Первый сценарий – интенсивность потока приоритетного трафика постоянная, интенсивность потока неприоритетного трафика растет, превышая обслуживающую способность МЭ.

Второй сценарий – интенсивность потока неприоритетного трафика постоянная, интенсивность потока приоритетного трафика растёт, превышая обслуживающую способность МЭ.

Первый сценарий поведения трафика. Для первого сценария исходные данные по входящим потокам трафика заданы следующим образом: $\lambda_1 = 40 \cdot 10^3 = const$, $\lambda_2 = 110 \cdot 10^3, 120 \cdot 10^3, \ldots, 280 \cdot 10^3$ пакетов в секунду.

Рисунок 4.5 демонстрирует динамику изменения интенсивности обслуженного потока пакетов в зависимости от интенсивности поступающего потока пакетов.

Рисунок 4.6 демонстрирует динамику изменения потерь в зависимости от интенсивности поступающего потока пакетов.

При превышении интенсивности суммарного входящего потока пакетов пропускной способности МЭ имеются потери неприоритетных пакетов (рисунок 4.6).

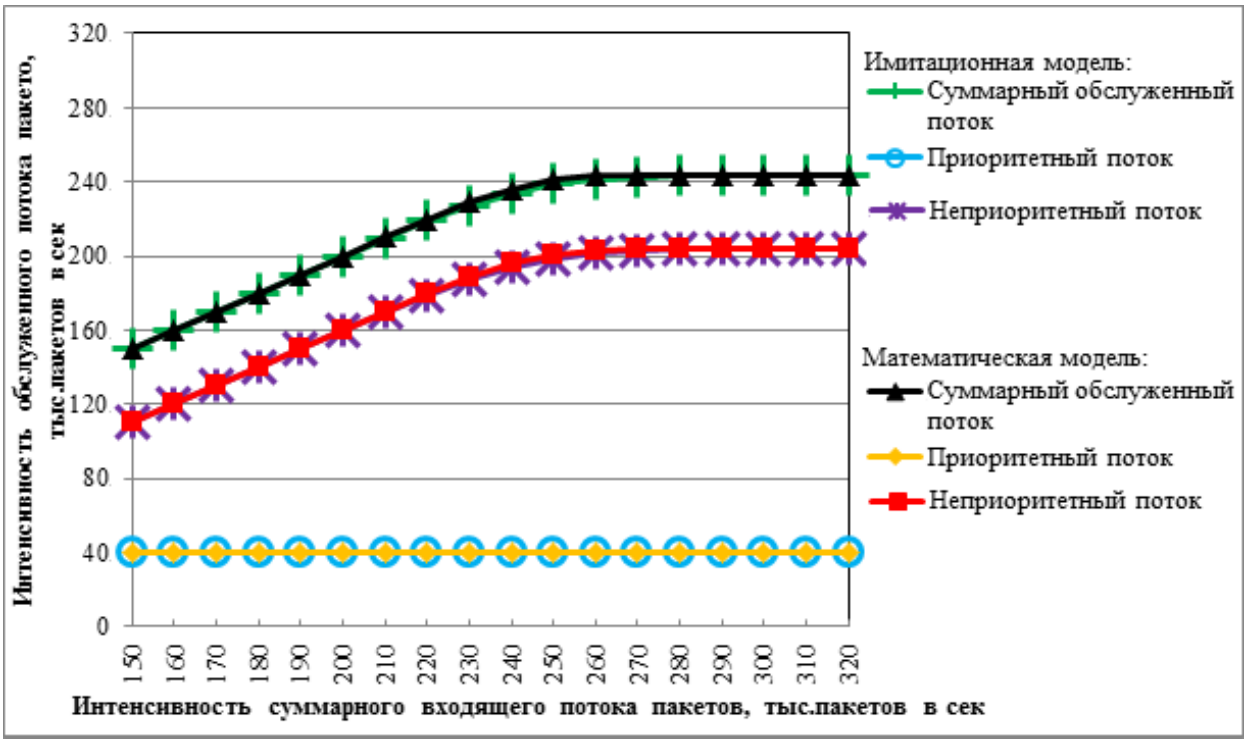


Рисунок 4.5 – Интенсивность обслуженного трафика (сценарий 1)

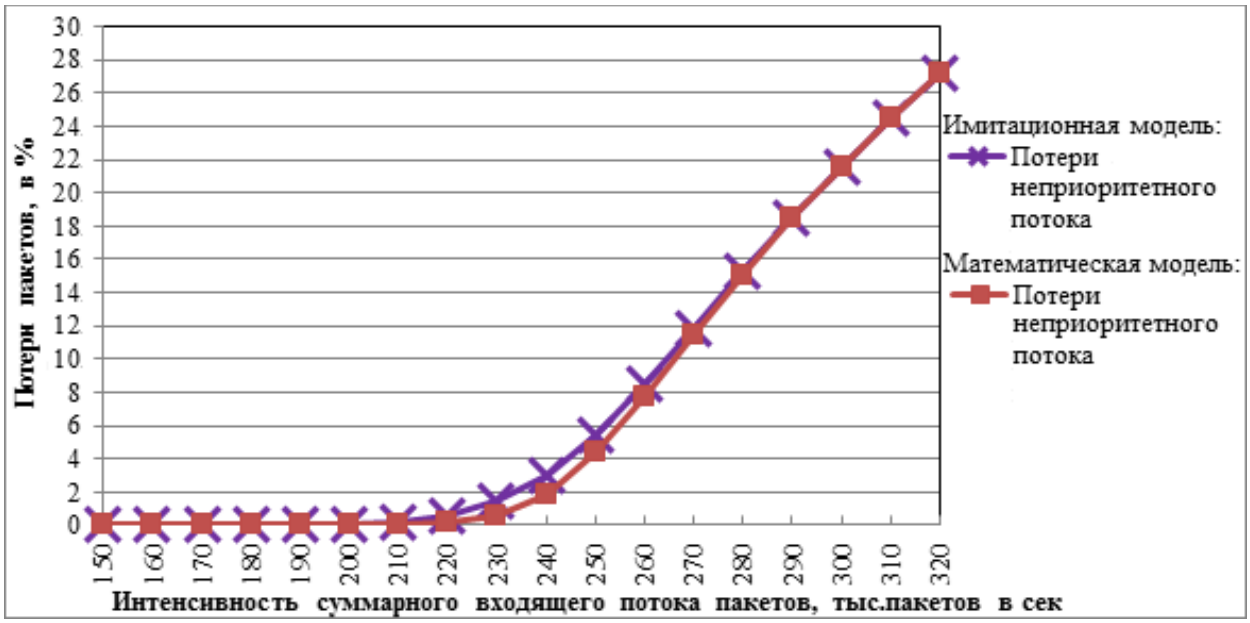


Рисунок 4.6 – Потери пакетов (сценарий 1)

Приоритетные пакеты не несут потерь при первом сценарии поведения трафика ввиду того, что интенсивность входящего потока приоритетных пакетов не превышает пропускную способность МЭ. При имитационном моделировании неприоритетные пакеты терпят несколько большие потери на отрезке ±10%

значения интенсивности потока пакетов в точке отказа, составляющие $\approx 1-2\%$ абсолютного значения потерь.

Различие зависимостей заполнения пакетной очереди находится в пределах 1 занятого места в очереди. График не приведен.

Второй сценарий поведения трафика. Для второго сценария исходные данные по входящим потокам пакетов заданы следующим образом: $\lambda_1 = 110 \cdot 10^3, 120 \cdot 10^3, \ldots, 280 \cdot 10^3$, $\lambda_2 = 40 \cdot 10^3 = const$ пакетов в секунду.

Рисунок 4.7 демонстрирует динамику изменения интенсивности обслуженного потока пакетов в зависимости от интенсивности поступающего потока пакетов.

Рисунок 4.8 демонстрирует динамику изменения потерь в зависимости от интенсивности поступающего потока пакетов.

После превышения интенсивности суммарного входящего потока пропускной способности МЭ имеют место потери неприоритетных пакетов, а после превышения пропускной способности МЭ интенсивностью приоритетного потока имеют место потери приоритетных пакетов (рисунок 4.8). В конечном итоге потери неприоритетных пакетов достигают $\approx 100\%$. При имитационном моделировании неприоритетные пакеты терпят несколько большие потери на отрезке $\pm 10\%$ значения интенсивности потока пакетов в точке отказа, составляющие $\approx 1-2\%$ абсолютного значения потерь.

Рисунок 4.9 демонстрирует динамику заполнения пакетной очереди в зависимости от интенсивности поступающего потока пакетов. Прослеживается общая динамика заполнения очереди. Количество неприоритетных пакетов в очереди растёт до тех пор, пока суммарная интенсивность входящих потоков трафика не достигнет точки отказа, а затем их количество начинает уменьшаться, так как их «вытесняют» из очереди приоритетные пакеты. Разница между результатами моделирования находится в пределах 3 – 11% от общего объёма очереди (30 пакетов при приведённых условиях эксперимента).

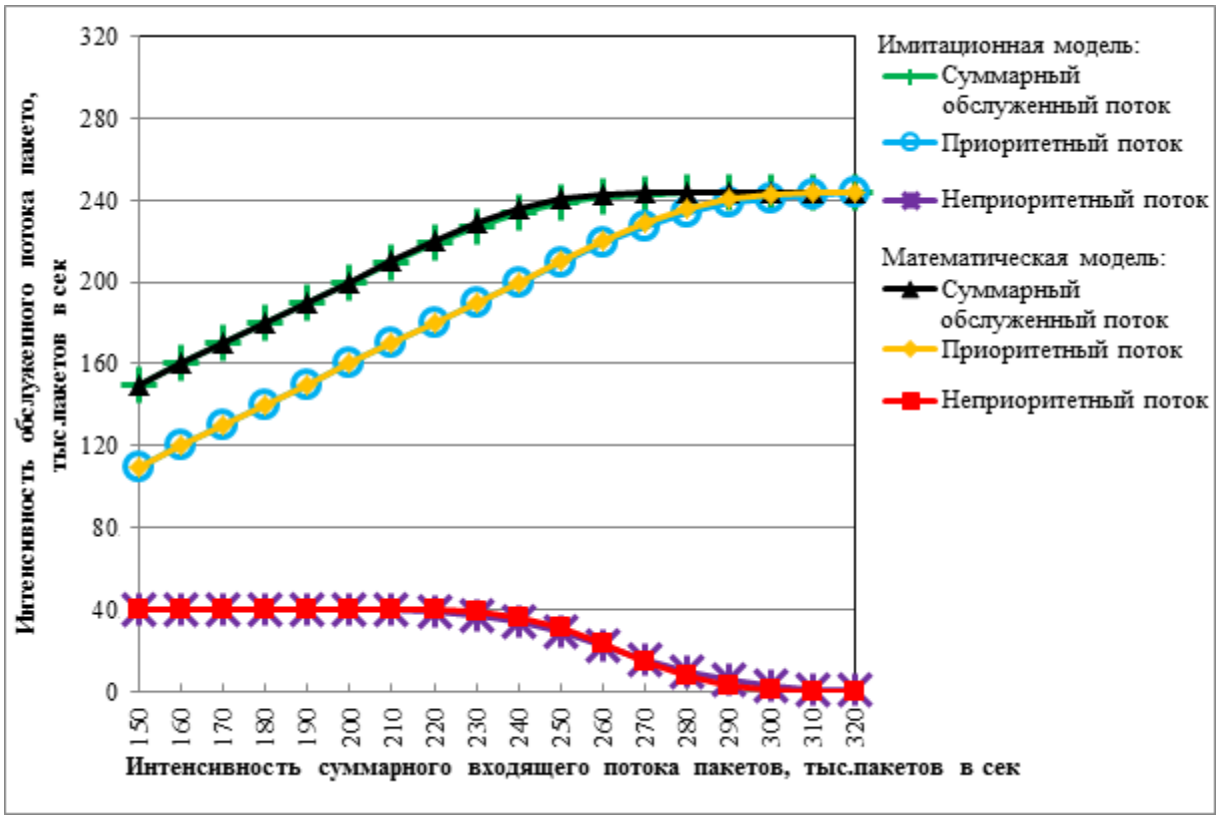


Рисунок 4.7 – Интенсивность обслуженного трафика (сценарий 2)

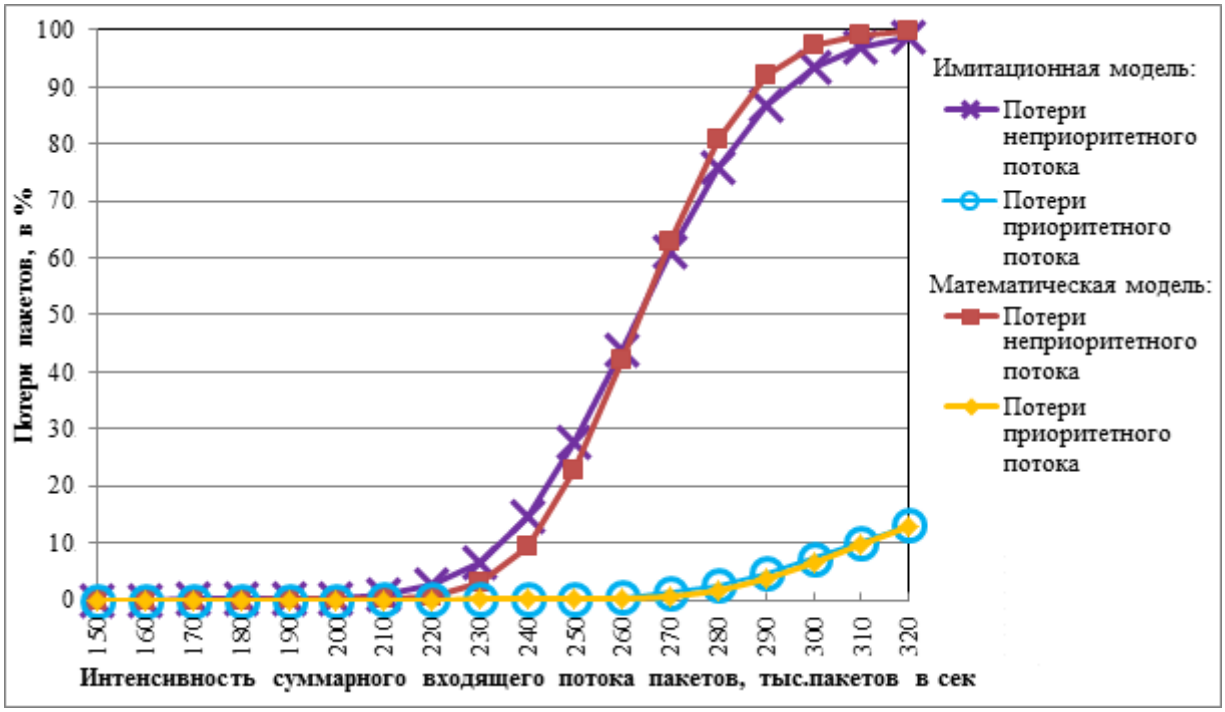


Рисунок 4.8 – Потери пакетов (сценарий 2)

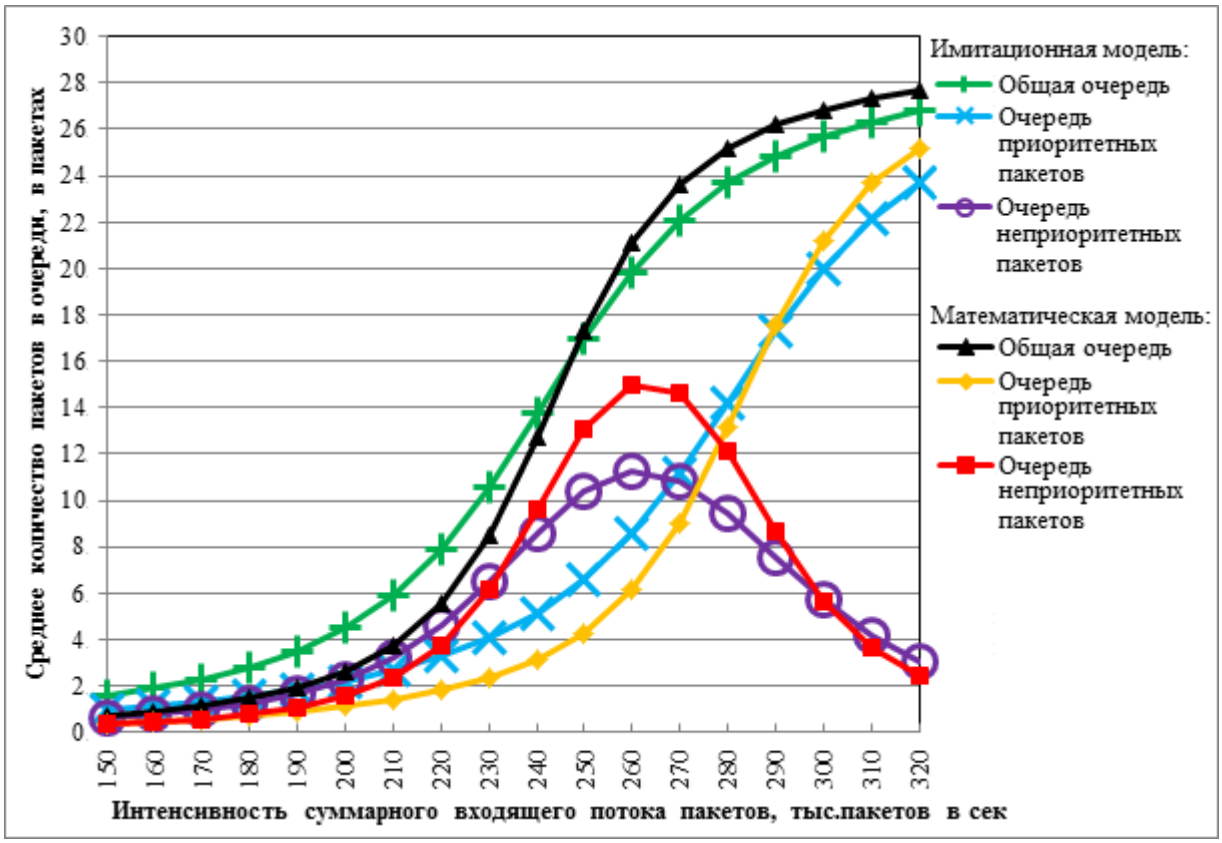


Рисунок 4.9 – Заполнение пакетной очереди (сценарий 2)

Результаты имитационного и математического моделирования, представленные в подразделе 4.4, совпадают в достаточной мере, для того чтобы говорить об их взаимозаменяемости при проведении расчётов показателей производительности МЭ, функционирующих в условиях приоритизации обслуживания трафика (с учётом принятых допущений). Результаты имитационного моделирования подтвердили соответствие математической модели поставленной задаче моделирования процессов функционирования МЭ в условиях приоритизации обслуживания трафика. Результаты раздела опубликованы в статье [137].

Выводы раздела

1. Постановка задачи на разработку имитационной модели функционирования межсетевого экрана в условиях приоритизации обслуживания трафика позволяет разработанной имитационной модели воспроизводить процесс обслуживания, принятый в математической модели для сравнения результатов моделирования.

2. Для реализации имитационной модели выбран высокоуровневый язык программирования общего назначения C#. В качестве основного подхода к имитационному моделированию выбрано дискретно-событийное моделирование. Это объясняется тем, что перечень возможных событий, возникающих в рамках процесса моделирования функционирования межсетевого экрана, ограничен, а появление событий подчиняется строгим логическим правилам

3. Сравнительный анализ результатов математического и имитационного моделирования при одинаковых исходных данных показал различия интенсивностей суммарного потока пакетов в пределах 1 – 2%, а различие среднего заполнения очереди – в пределах 3 – 11% от общего объёма очереди. Результаты сравнения позволяют подтвердить соответствие математической и имитационной моделей поставленным задачам на их разработку, что, в частности, говорит о корректности имитационной модели и позволяет производить с её использованием дополнительные исследования, приведённые в разделе 5.

РАЗДЕЛ 5. ОЦЕНКА ФУНКЦИОНИРОВАНИЯ МЕЖСЕТЕВОГО ЭКРАНА В УСЛОВИЯХ ПРИОРИТИЗАЦИИ ОБСЛУЖИВАНИЯ ТРАФИКА ПРИ ИЗМЕНЕНИИ ХАРАКТЕРИСТИК ОБСЛУЖИВАНИЯ

5.1. Моделирование функционирования межсетевого экрана в условиях приоритизации обслуживания трафика при условии постоянного времени обслуживания

5.1.1. Снимаемое допущение, причины его снятия и цель эксперимента

Имитационная модель позволяет снять допущение, связанное с тем, что в математической модели время обслуживания заявок является случайной величиной распределённой по экспоненциальному закону.

В разделе 1 и 2 несколько раз упоминалось о влияние высокой степень аппаратной поддержки функций обработки пакетов на обслуживание. Обработка такого типа основана на использовании *интегральных схем специального назначения* [138, 139], *сетевых процессоров* [138, 140]. Функции, обрабатываемые подобным аппаратным обеспечением, вносят задержку обслуживания пакетов, близкую к постоянной величине, нежели случайной величиной, распределённой по экспоненциальному закону.

Целью эксперимента является проверка различия в показателях производительности МЭ при постоянном и экспоненциальном временах обслуживания пакетов.

В дополнение приведём графические зависимости времени нахождения пакетов в очереди от интенсивности суммарного входящего потока пакетов.

5.1.2. Исходные данные

В рамках эксперимента рассматривается два сценария поведения трафика: рост интенсивности приоритетного потока пакетов при постоянно потоке неприоритетных пакетов и рост интенсивности потока неприоритетных пакетов при постоянном потоке приоритетных пакетов.

Величины входных параметров аналогичны тем, что использованы ранее и представлены в подразделах 3.3 и 4.4.

5.1.3. Результаты моделирования

Первый сценарий поведения трафика. Для первого сценария исходные данные по входящим потокам пакетов заданы следующим образом: $\lambda_1 = 40 \cdot 10^3 = const$, $\lambda_2 = 110 \cdot 10^3, 120 \cdot 10^3, \ldots, 280 \cdot 10^3$ пакетов в секунду.

Рисунок 5.1 демонстрирует динамику изменения интенсивности обслуженного потока пакетов в зависимости от интенсивности поступающего потока пакетов. Значительного различия в интенсивностях обслуженных потоков пакетов нет, однако, вблизи точки отказа интенсивность потока неприоритетных пакетов при постоянно времени обслуживания несколько выше (достигается различие в 2%).

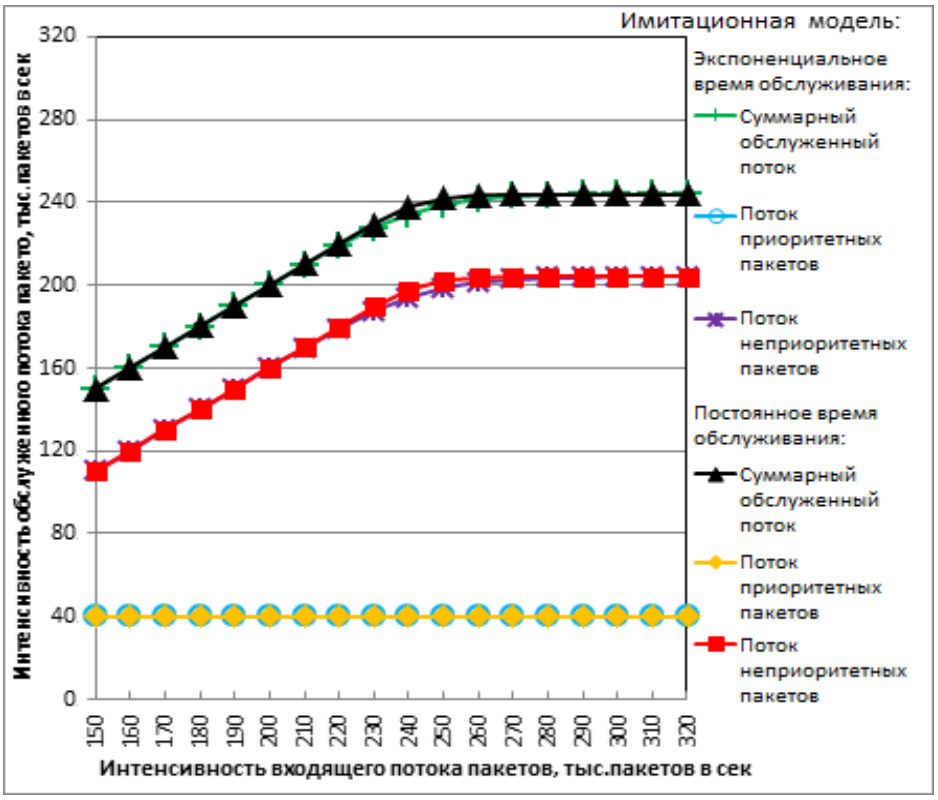


Рисунок 5.1 – Интенсивность обслуженного трафика (сценарий 1)

Рисунок 5.2 демонстрирует динамику изменения потерь пакетов в зависимости от интенсивности поступающего потока пакетов.

Вблизи точки отказа, то есть ±15% от интенсивности обслуженного потока пакетов заметны различия в потерях пакетов, составляющие до 1,5% в

абсолютном значении потерь. При увеличении интенсивности входящего потока пакетов значение потерь сравнивается.

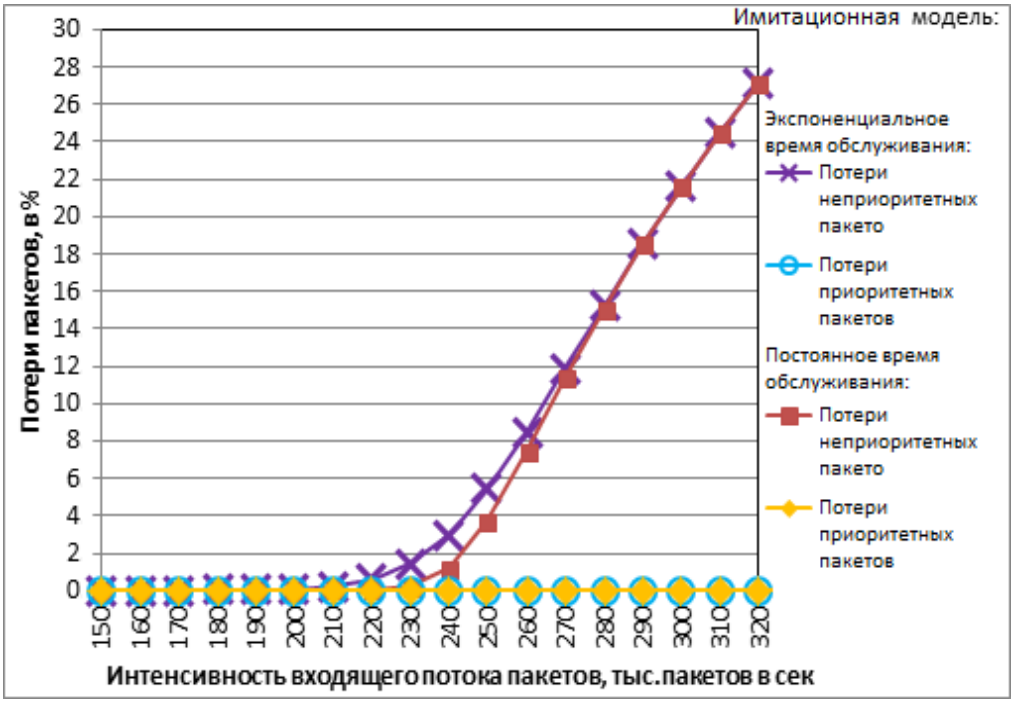


Рисунок 5.2 – Потери пакетов (сценарий 1)

Рисунок 5.3 демонстрирует динамику заполнения пакетной очереди в зависимости от интенсивности поступающего потока пакетов.

Рисунок 5.4 демонстрирует динамику изменения времени нахождения пакетов в очереди от интенсивности поступающего потока пакетов.

Графики заполнения очереди и времени нахождения пакетов в ней взаимосвязаны. По этой причине их можно описывать совместно. Пересечение графиков происходит в точке отказа. При отсутствии потерь, то есть до достижения точки отказа, время нахождения пакетов в очереди при постоянной длительности обслуживания и при экспоненциальной длительности обслуживания различается приблизительно вдвое. Чем ближе система к точке отказа, тем менее заметны различия во времени нахождения обслуженных пакетов в очереди.

С точки зрения математики увеличение времени нахождения в очереди объясняется тем, что оно пропорционально дисперсии времени обслуживания. Дисперсия времени обслуживания равна 0 при постоянном времени

обслуживания. Эти результаты подтверждаются в работах Л. Клейнрока [141]. Однако математические методы не объясняют поведение времени обслуживания за точкой отказа.

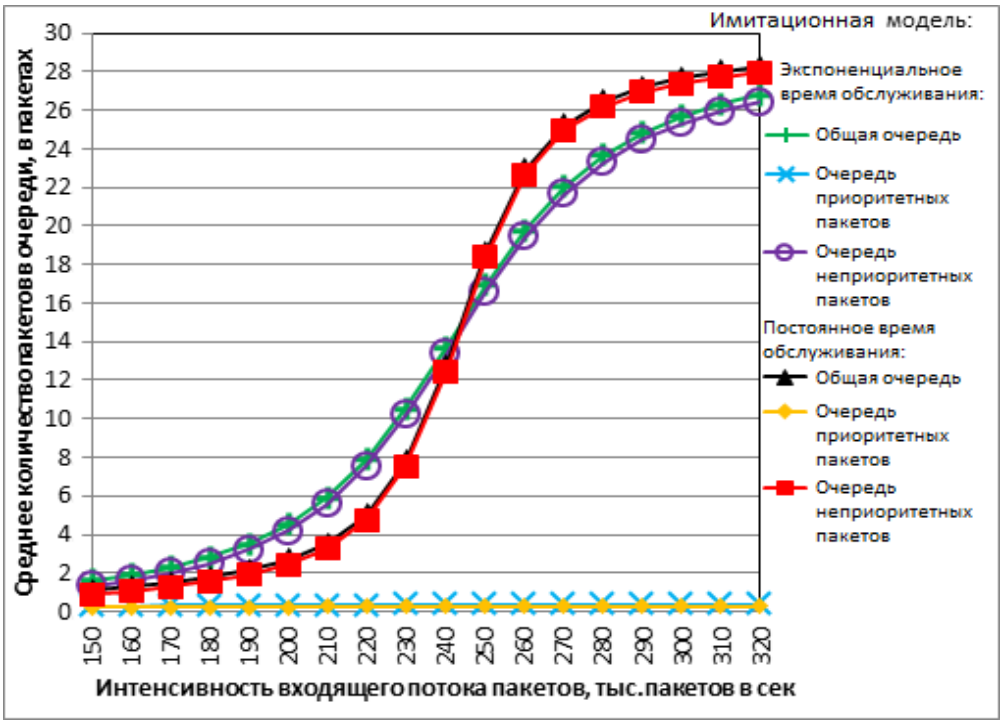


Рисунок 5.3 – Заполнение пакетной очереди (сценарий 1)

Если рассматривать эти результаты с физической точки зрения, то можно сказать следующее: до приближения к точке отказа, когда очередь почти пуста, экспоненциальное время обслуживания будет периодически сильно замедлять и ускорять обслуживание. При значительном ускорении времени обслуживания, слабо загруженная очередь может опустеть, что не окажет значительного влияния на задержку пакетов. Значительное увеличение времени обслуживания может привести к заполнению очереди, при этом время обслуживания пакетов заметно возрастёт. На это может накладываться экспоненциально распределённое время прихода пакетов, которое также способно в этот момент прислать большее число пакетов. При приближении к точке отказа очередь начинает заполняться сильнее, а влияние ускорения обслуживания будет расти. Время в очереди при экспоненциальном и постоянном временах обслуживания будет стремиться сравнятся. После преодоления точки отказа ситуация изменится, большее влияние на время в очереди будет оказывать уменьшение времени обслуживания.

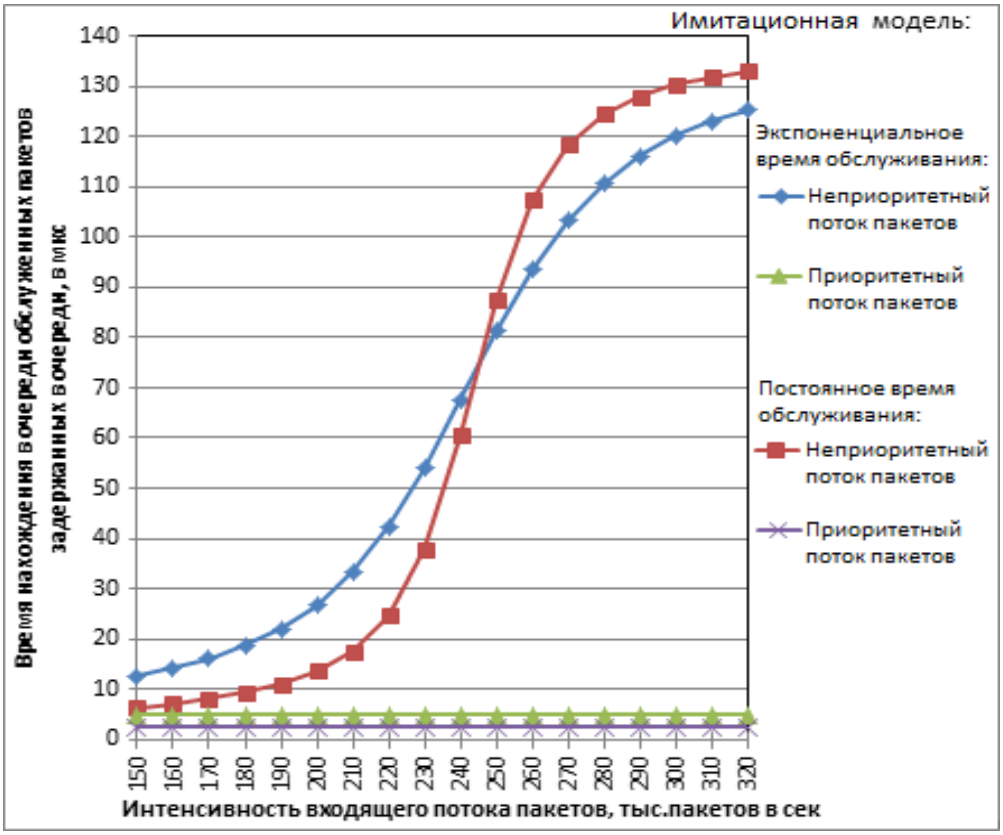


Рисунок 5.4 – Время нахождения в очередь обслуженных пакетов (сценарий 1)

Второй сценарий поведения трафика. Для второго сценария исходные данные по входящим потокам пакетов заданы следующим образом: $\lambda_1 = 110 \cdot 10^3,\ 120 \cdot 10^3,\ \ldots,\ 280 \cdot 10^3,\ \lambda_2 = 40 \cdot 10^3 = const$ пакетов в секунду.

Рисунок 5.5 демонстрирует динамику изменения интенсивности обслуженного потока пакетов в зависимости от интенсивности поступающего потока пакетов. Динамика системы сохраняется как при постоянном, так и при экспоненциальном временах обслуживания.

Рисунок 5.6 демонстрирует динамику заполнения очереди в зависимости от интенсивности поступающего потока пакетов. Вблизи точки отказа, то есть $\pm 10\%$ от интенсивности обслуженного потока пакетов заметны различия в потерях пакетов, составляющие $\approx 1\%$.

Рисунок 5.7 демонстрирует динамику заполнения пакетной очереди от интенсивности поступающего потока пакетов.

Рисунок 5.8 демонстрирует динамику изменения времени нахождения пакетов в очереди от интенсивности поступающего потока пакетов.

Прослеживается общая динамика заполнения очереди. Объяснение поведения кривых аналогично тому, что приведено для рисунков 5.3 и 5.4 ранее.

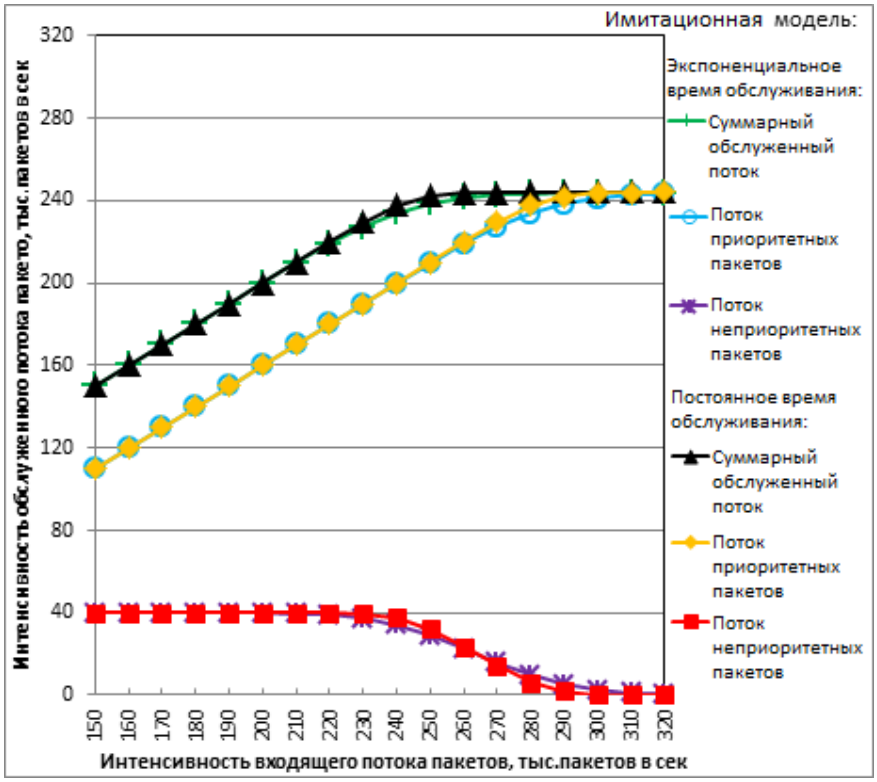


Рисунок 5.5 – Интенсивность обслуженного трафика (сценарий 2)

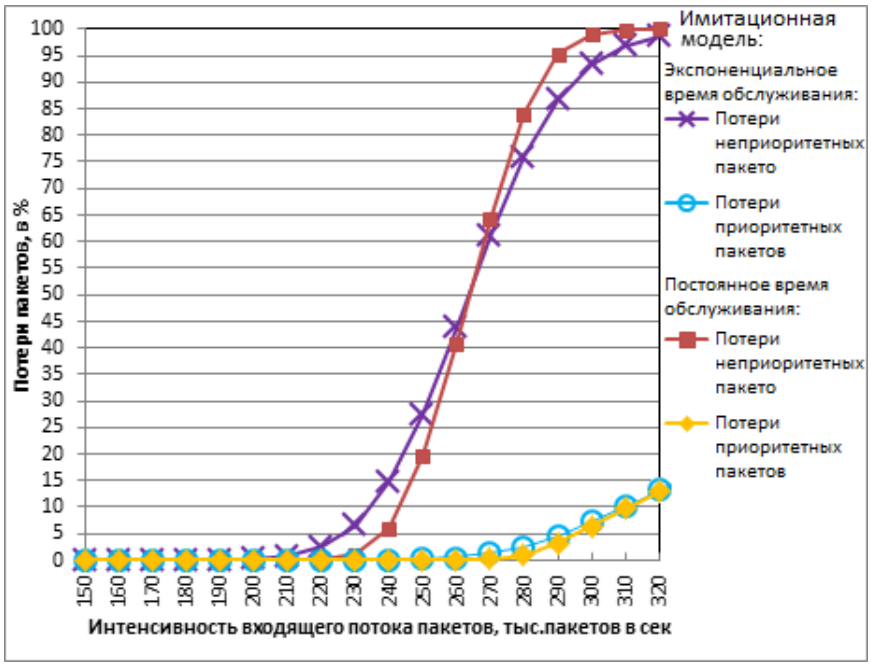


Рисунок 5.6 – Потери пакетов (сценарий 2)

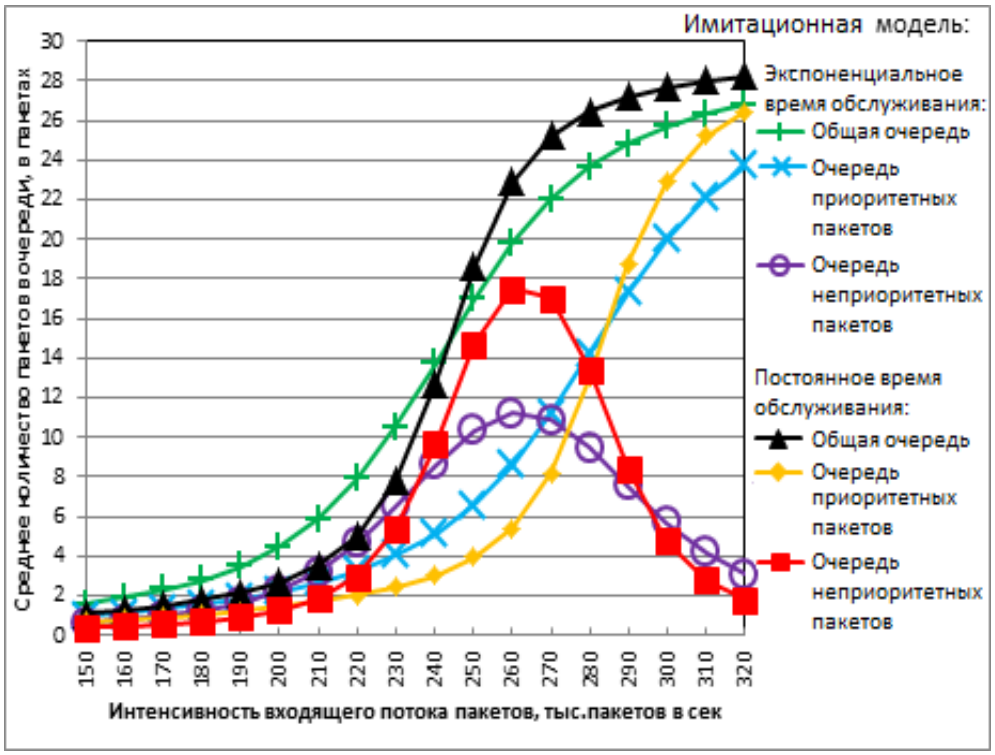


Рисунок 5.7 – Заполнение пакетной очереди (сценарий 2)

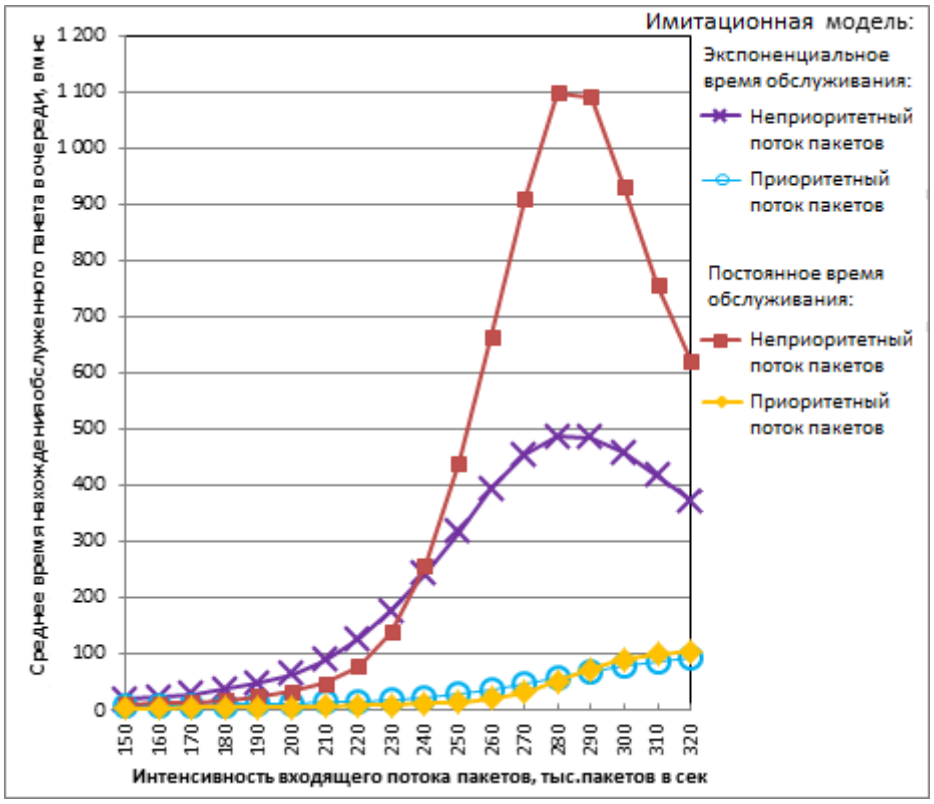


Рисунок 5.8 – Время нахождения в очереди обслуженных пакетов (сценарий 2)

Краткие выводы. Сравнение времени нахождения в очереди обслуженных пакетов (рисунки 5.4, 5.8) при случайной длительности обслуживания, распределённой по экспоненциальному закону вероятностей, и при постоянной

длительности обслуживания показало, что при отсутствии потерь пакетов и при постоянной длительности обслуживания задержка в очереди примерно в два раза меньше, чем при случайной длительности обслуживания, что соответствует результатам, приведенным Л. Клейнроком в [141].

При наличии потерь зависимость становиться обратной. Наличие экспоненциального времени обслуживания повышает вероятность частичного освобождения очереди от пакетов, которая влияет на время обслуживания. Обратная ситуация с перегрузкой несколько увеличивает потери пакетов, однако не приводит к увеличению времени обслуживания. Этим объясняется поведение графиков временных характеристик обслуживания.

5.2. Исследование влияния изменения размера очереди и количества правил фильтрации на показатели производительности межсетевого экрана, функционирующего в условиях приоритизации обслуживания трафика

5.2.1. Исходные данные

В рамках эксперимента рассматривается два сценария поведения трафика: первый – рост интенсивности потока неприоритетных пакетов при постоянном потоке приоритетных пакетов и второй – рост интенсивности приоритетного потока пакетов при постоянно потоке неприоритетных пакетов. Входные параметры МЭ аналогичны предыдущим экспериментам (подразделы 4.4, 5.1).

Эксперимент разделён надвое, то есть в первом случае переменной величиной является максимальный размер очереди, а во втором случае изменяется позиция правила фильтрации, на котором происходит совпадение.

В первом случае увеличим максимальный размер очереди в 2, 4, 8 и 32 раза, каждый раз повторяя весь расчёт. Результаты первого эксперимента при втором сценарии поведения трафика не показали интересных результатов и далее представлены не будут. Расчёты по второму эксперименту показали линейный рост задержки в обслуживании. Результаты второго эксперимента не имеют дополнительных особенностей относительно результатов первого эксперимента.

Во втором эксперименте меняется именно позиция правила, на котором происходит совпадение, т.е. максимальный размер базы правил не рассматривается. Уточним, что при работе экспоненциального ГСЧ периодически могут появляться крайне большие величины времени обслуживания, которые будут соответствовать правилу, стоящему на более глубокой позиции базы, чем рассматриваемое.

5.2.2. Результаты первого эксперимента

Первый сценарий поведения трафика. Для первого сценария исходные данные по входящим потокам пакетов заданы следующим образом: $\lambda_1 = 40 \cdot 10^3 = const$, $\lambda_2 = 110 \cdot 10^3$,$120 \cdot 10^3$, ... , $280 \cdot 10^3$ пакетов в секунду.

На рисунке 5.9 продемонстрированы кривые заполнения очереди неприоритетными пакетами ($QueueAvr_{unpriority}$) от интенсивности поступающего потока пакетов. Ось ординат представлена в логарифмическом формате (основание логарифма равно 2).

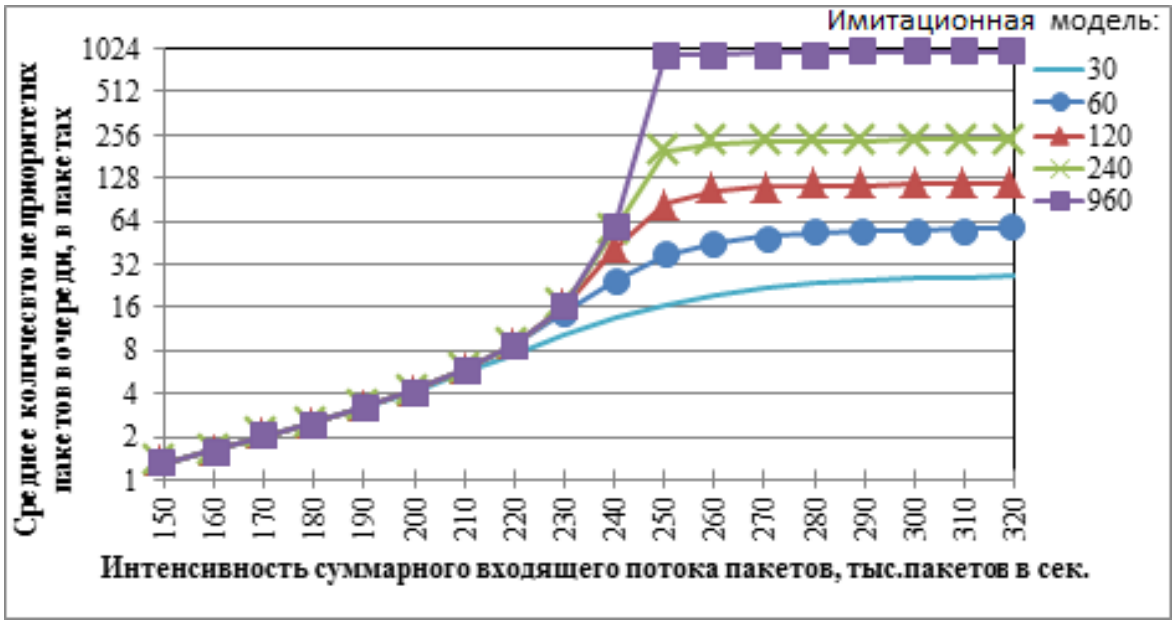


Рисунок 5.9 – Заполнение очереди неприоритетными пакетами (сценарий 1)

Уточним, что в зависимости от интенсивности входящего потока неприоритетных пакетов среднее количество приоритетных в очереди составляет $\approx 0.28 - 0.361$. Среднее количество приоритетных пакетов в очереди не зависит от размера очереди. Изменение среднего количества приоритетных пакетов в

очереди от интенсивности входящего потока неприоритетных пакетов объясняется повышением вероятности того, что система занята обслуживанием неприоритетного пакета в момент попадания приоритетного пакета в очередь. Тем самым приоритетный пакет будет ожидать в очереди до окончания обслуживания неприоритетного пакета.

Рисунок 5.10 демонстрирует динамику заполнения очереди в зависимости от интенсивности поступающего потока пакетов. Для расчёта использовались величина $QueueAvr_{unpriority}$, описанная в пункте 4.3.4. Расчёт проведен по формулам, поточечно:

$$30 = QueueAvr_{unpriority}(30) / 30,$$
$$60 = QueueAvr_{unpriority}(60) / 60,$$
$$...$$
$$960 = QueueAvr_{unpriority}(960) / 960.$$

Из рисунка 5.10 видно, что чем больше максимальный размер очереди, тем более «предсказуемо» ведёт себя модель. При моделировании появляется достаточно высокая вероятность того, что характеристики части пакетов, попадающих в систему, будут отличаться от их математического ожидания. Это значит, что чем больше в системе максимальный размер очереди, тем ниже вероятность сброса пакетов из-за частого прихода пакетов с долгим временем обслуживания. То есть большая очередь будет сглаживать появление таких событий, буферизируя пакеты.

До того, как интенсивность входящего потока пакетов достигнет значения в 90% от предельной интенсивности обслуживания, абсолютное значение среднего количества пакетов в очереди не зависит от количества мест в очереди.

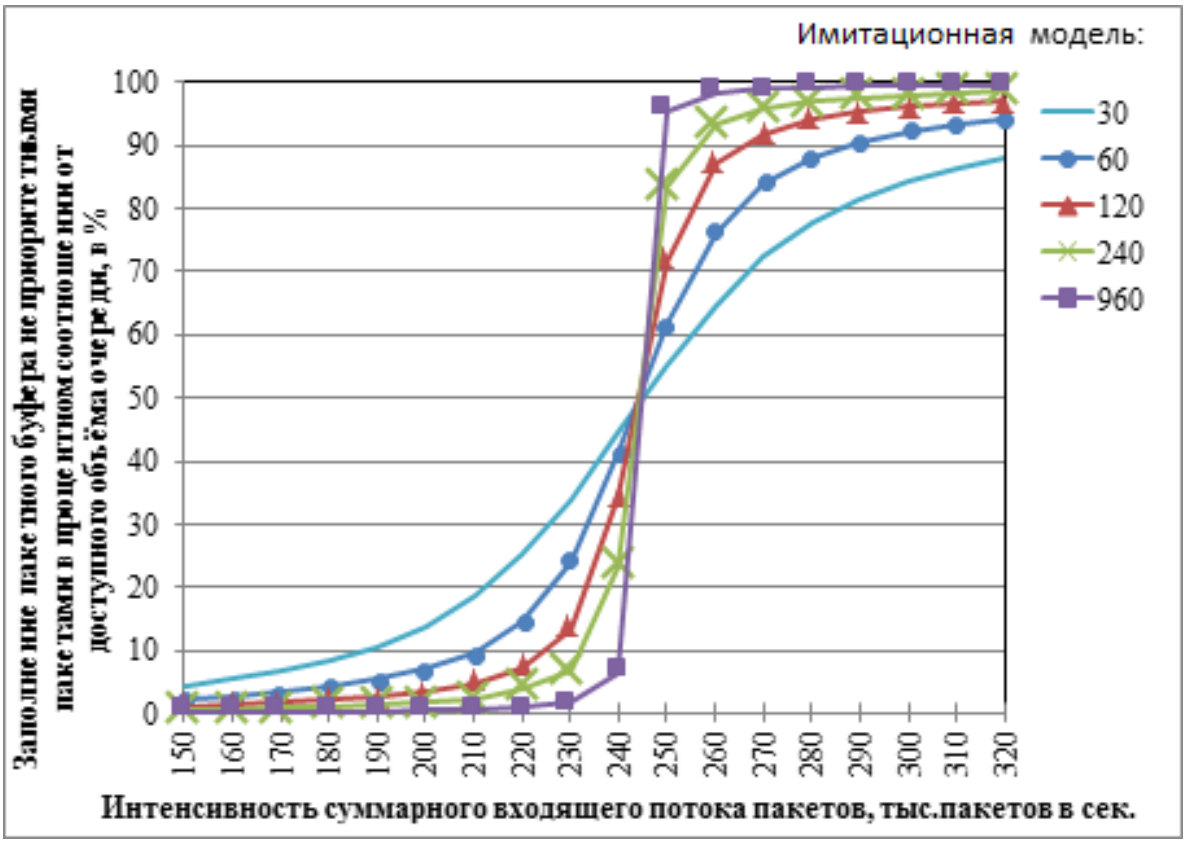


Рисунок 5.10 – Заполнение очереди неприоритетными пакетами в процентном соотношении от доступного объёма очереди (сценарий 1)

На рисунке 5.11 показаны соотношения количества обслуженных неприоритетных пакетов, задержанных в очереди, от интенсивности поступающего потока пакетов. Для расчёта использовались величина $CPrW_{unpriority}$, описанная в пункте 4.3.4. Расчёт проведен по формулам, поточечно:

$$30 = CPrW_{unpriority}(30) / CPrW_{unpriority}(30) = 1,$$

$$60 = CPrW_{unpriority}(60) / CPrW_{unpriority}(30),$$

...

$$960 = CPrW_{unpriority}(960) / CPrW_{unpriority}(30).$$

Из рисунка 5.11 видно, что увеличение максимального размера очереди приводит к тому, что при приближении к точке отказа МЭ обслуживает большее количество пакетов с очередью. Чем меньше величина очереди, тем больше потерь будет терпеть система из-за случайных бросков трафика. При повышении интенсивности после точки отказа разница в количестве неприоритетных пакетов, обслуженных с очередь, для различных размеров очереди уменьшается, так как

система в принципе достигла предела своих возможностей. Однако из-за продолжающихся колебаний интенсивности входящего потока пакетов соотношения не сразу выравниваются.

На рисунке 5.12 показано соотношение неприоритетных пакетов обслуженных без задержки в очереди (заставших систему пустой) от интенсивности поступающего потока пакетов. Для расчёта использовались величина $CPrWo_{unpriority}$, описанная в пункте 4.3.4. Расчёт проведен по формулам, поточечно:

$$30 = CPrWo_{unpriority}(30) / CPrWo_{unpriority}(30) = 1,$$
$$60 = CPrWo_{unpriority}(60) / CPrWo_{unpriority}(30),$$
$$\ldots$$
$$960 = CPrWo_{unpriority}(960) / CPrWo_{unpriority}(30).$$

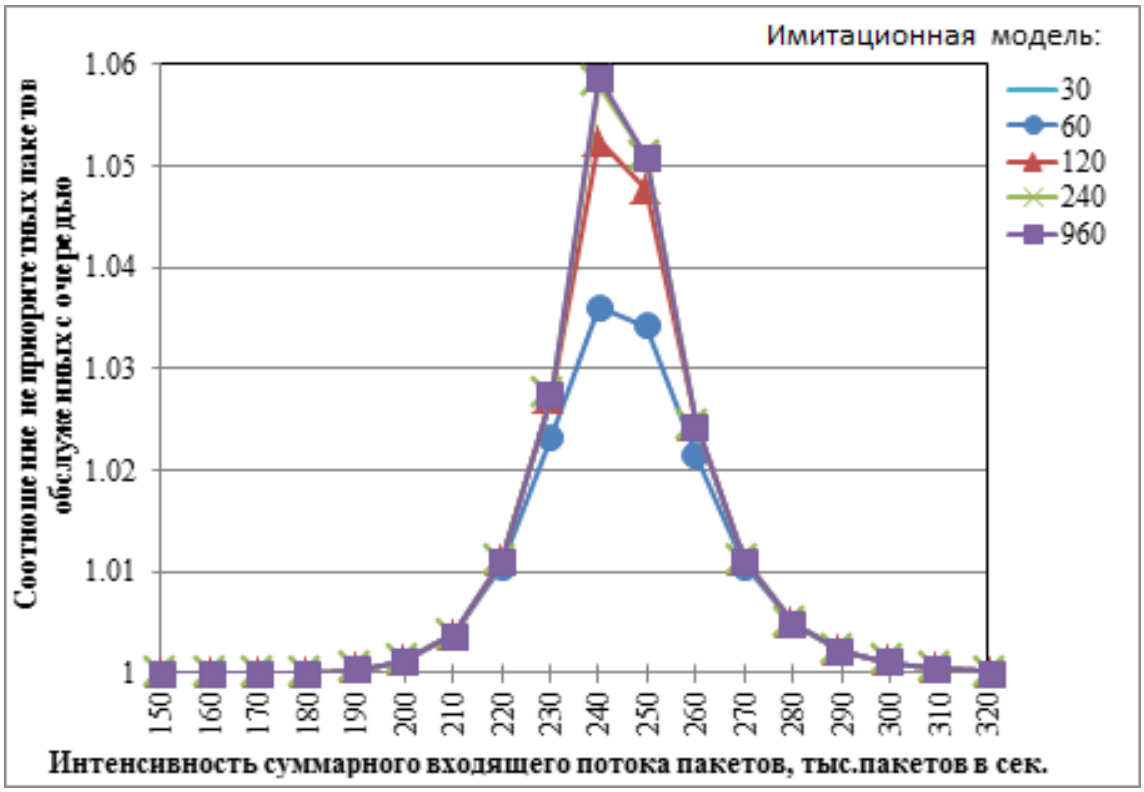


Рисунок 5.11 – Соотношение неприоритетных пакетов, обслуженных с очередью (сценарий 1)

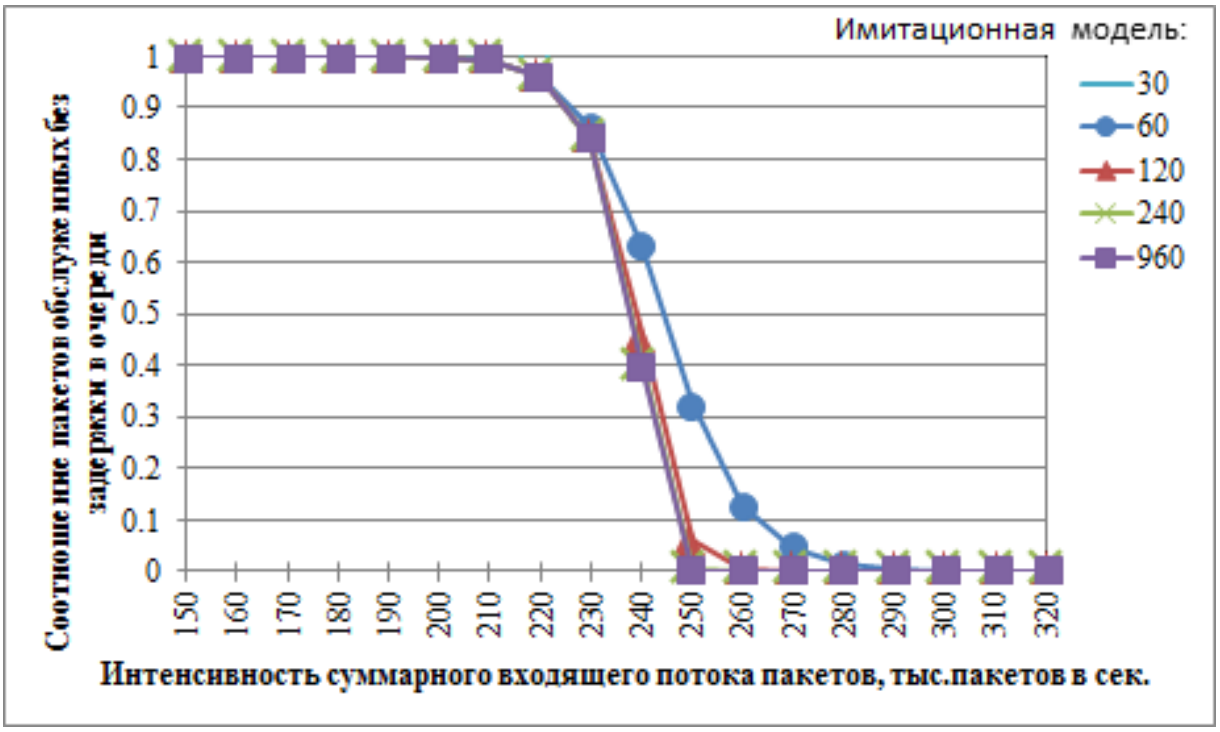


Рисунок 5.12 – Соотношение пакетов, обслуженных без задержки в очереди (сценарий 1)

Из рисунка 5.12 видно, что по достижении точки отказа почти во всех системах исчезает вероятность появления неприоритетных пакетов обслуженных без задержки в очереди, что аналогично тому, что на момент прихода пакета, система была пуста.

На рисунке 5.13 продемонстрированы графики задержек в очереди обслуженных неприоритетных пакетов, которые были задержаны в очереди, от интенсивности поступающего потока пакетов. К таким пакетам относятся все неприоритетные пакеты, которые попали в очередь и были обслужены, пакеты заставшие систему пустой и сразу попавшие на обслуживание не учитываются. Ось ординат представлена в логарифмическом масштабе. Для расчёта использовались величина $TAvrQperQueueServedPacket_{unpriority}$, описанная в пункте 4.3.4.

По графику видно, что увеличение максимального объёма очереди приводит к значительным задержкам пакетов. Соотношение этих задержек стремится к соотношению размеров очередей, но не достигает их, смотри описание к рисунку 5.9.

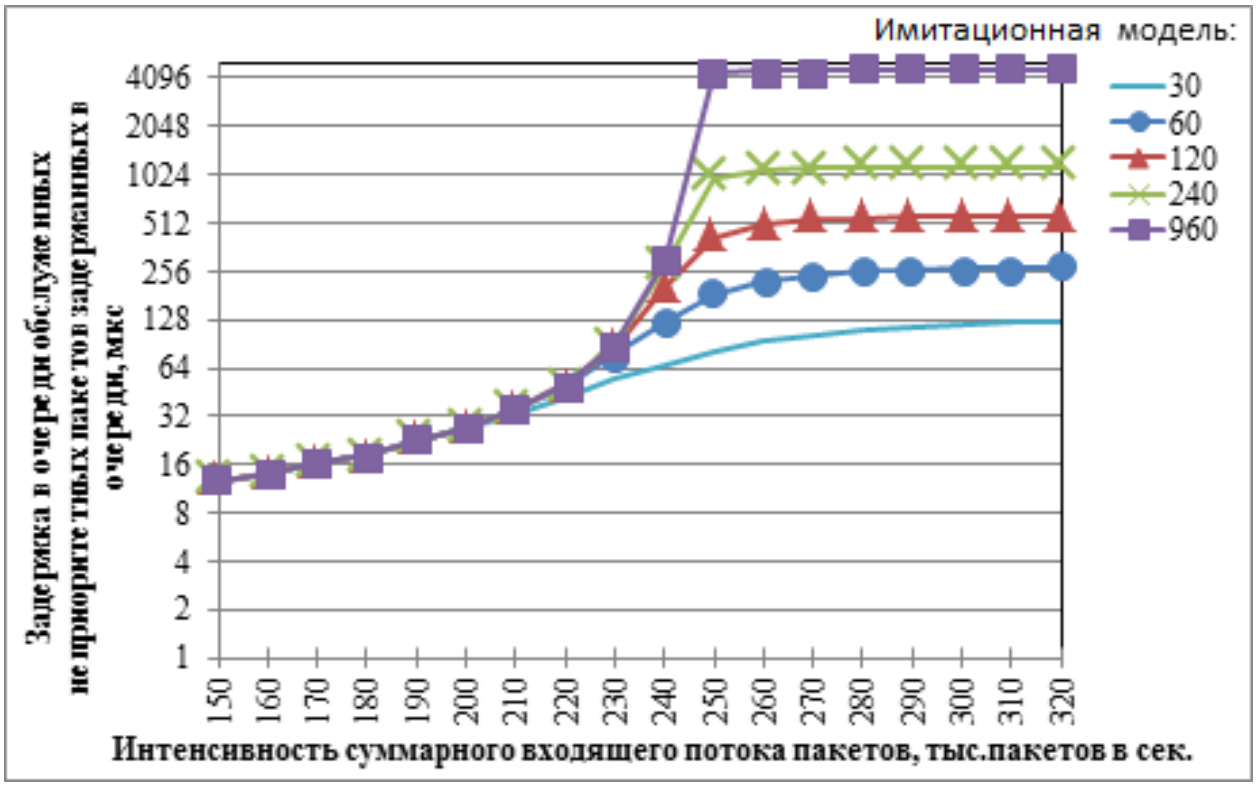


Рисунок 5.13 – Задержка в очереди обслуженных неприоритетных пакетов, задержанных в очереди (сценарий 1)

На рисунке 5.14 продемонстрированы соотношения времени нахождения обслуженных задержанных в очереди неприоритетных пакетов от интенсивности поступающего потока пакетов. К таким пакетам относятся все неприоритетные пакеты, которые были обслужены и находились в очереди ненулевое время.

Результаты, представленные на графике 5.14, нормированы базовыми значениями эксперимента, то есть значениями системы при очереди в 30 мест. График нормирующих значений даёт линию на уровне «1».

График демонстрирует две особенности поведения системы. Первая особенность – резкий относительный рост задержки при достижении точки отказа, отражаемый на схеме пиками при интенсивности 250 тыс. пакетов в секунду. Вторая особенность – стремление значения соотношения задержек выйти на уровень соотношения максимального размера каждой из очередей.

Для расчёта использовалась величина $TAvrQperQueueServedPacket_{unpriority}$, описанная в пункте 4.3.4. Расчёт проведен по формулам, поточечно:

$$30 = \frac{TAvrQperQueueServedPacket_{unpriority}(30)}{TAvrQperQueueServedPacket_{unpriority}(30)} = 1,$$

$$60 = T\frac{TAvrQperQueueServedPacket_{unpriority}(60)}{TAvrQperQueueServedPacket_{unpriority}(30)},$$

...

$$960 = \frac{TAvrQperQueueServedPacket_{unpriority}(960)}{TAvrQperQueueServedPacket_{unpriority}(30)}.$$

Первая особенность объясняется тем, что увеличение максимального размера очереди снижает потери в точке отказа, то есть пакеты, попавшие в расширенную очередь, не будут потеряны, но будут находиться в очереди значительно большее время по сравнению с теми, которые попали в очередь меньшего размера. Стоит понимать, что в точке отказа скорость заполнения очереди достаточно невысокая, так как интенсивность входящего потока пакетов незначительно превышает интенсивности обслуживания. Если интенсивность входящего потока трафика значительно выше интенсивности обслуживания (приблизительно более 20%), то такого яркого относительного броска задержки не было бы. Из рисунка 5.11 хорошо видно, насколько больше пакетов обслуживается с ожиданием в очереди вблизи точки отказа при увеличении максимального размера очереди.

Вторая особенность (соотношение задержек) объясняется тем, что пакетам необходимо ждать большее время в очереди, если она имеет больший объём. Если бы приоритета обслуживания не было, то эта зависимость была бы линейной. Однако значение соотношения задержек отличается от соотношения размера очереди и никогда не будет достигнута. Например, при текущих характеристиках обслуживания, если провести дополнительный расчёт для точки с интенсивностью суммарного входящего потока пакетов равной 540 тысяч пакетов в секунду, соотношения будут следующими: 2,0764, 4,2296, 8,5356, 34,3732, а не 2, 4, 8 и 32 раза соответственно, то есть на 3,82%, 5,74%, 6,69%, 7,416% больше

ожидаемых величин. Дополнительная задержка объясняется тем фактом, что при увеличении объёма очереди неприоритетные пакеты, попавшие в очередь, за своё время нахождения в ней застанут большее число событий прихода приоритетных пакетов, которые, в свою очередь, попав в систему, будут приостанавливать продвижение на обслуживание неприоритетных пакетов. Чем больше пакетов в очереди, тем больше приоритетных пакетов придёт за время нахождения неприоритетного обслуженного пакета в очереди.

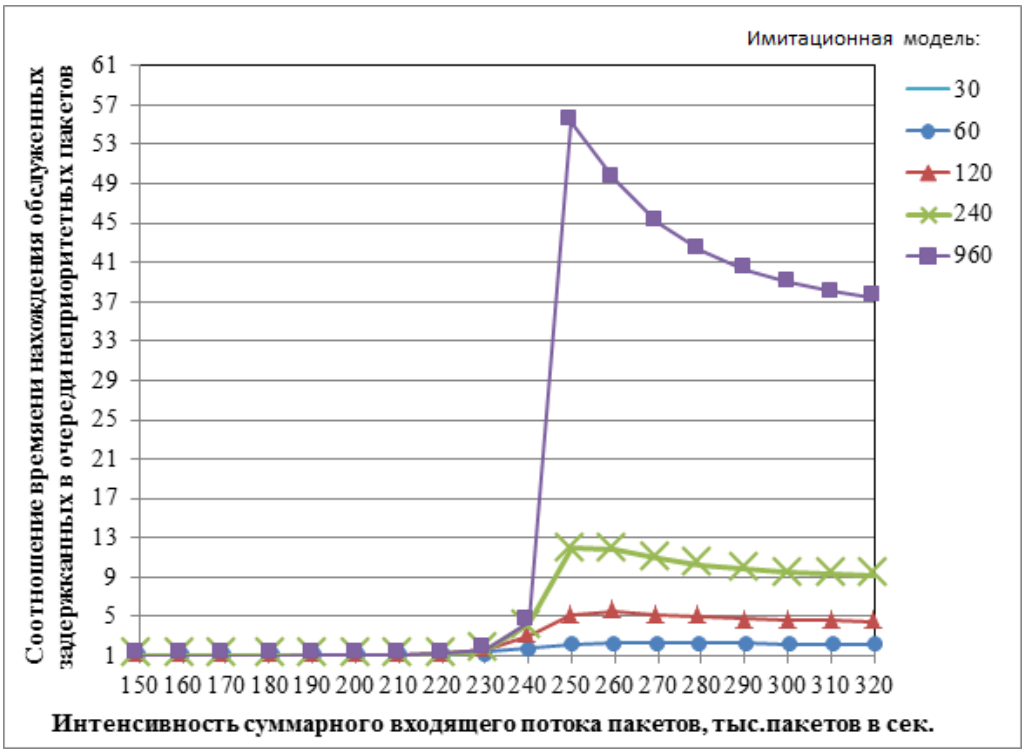


Рисунок 5.14 – Соотношение задержек в очереди обслуженных неприоритетных пакетов, задержанных в очереди (сценарий 1)

Увеличение максимального объёма очереди приводит к более медленному её заполнению и отсутствию ранних отказов в обслуживании, что приводит к увеличению средней задержки и, как следствие, соотношения задержек. На этот фактор сильно влияет соотношение интенсивности обслуживания и прихода пакетов, то есть если за время обслуживания одного пакета очередь будет заполняться на значительный процент, то величина соотношения задержек будет изменяться в меньшую сторону.

Исследования также показали, что увеличение максимального объёма очереди приводит к линейному росту задержки неприоритетных пакетов от

прихода приоритетных пакетов. Рост соотношения объясняется тем, что с увеличением количества пакетов в очереди, неприоритетные пакеты чаще застают приход приоритетных пакетов, которые приходится пропускать на обслуживание. При текущих условиях эксперимента эта задержка в очереди, составляет около 16,4% от общей величины задержки неприоритетных пакетов в очереди в любой точке эксперимента. Относительное значение этой величины рассчитывается из соотношения интенсивности приоритетного потока пакетов к максимальной интенсивности обслуживания.

Выводы по результатам первого эксперимента

Для формирования выводов по результатам эксперимента стоит разделить поведение системы на 3 участка по соотношению интенсивности входящего потока пакетов к интенсивности обслуживания. Первый участок располагается по шкале интенсивности входящего потока пакетов от начала эксперимента до 80-85% интенсивности обслуживания, второй от 80-85% до 110-115% интенсивности обслуживания, третий от 110-115% и более интенсивности обслуживания. Первый участок характеризуется почти полным отсутствием потерь пакетов при всех размерах очереди. Второй участок характеризуется началом переполнения модели, достижением точки отказа и дальнейшим ростом интенсивности суммарного входящего потока трафика. Третий участок характеризуется значительными потерями пакетов и стабилизацией работы систем с очередями меньшего размера.

В течение первого участка нет значительных различий в качестве обслуживании при различных размерах очереди. Ближе к точке отказа в системе с очередью в 30 пакетов присутствуют сверхмалые потери, которых, можно было полностью избежать, увеличив максимальный объём очереди до 60 пакетов. Подобное решение незначительно нагрузило бы устройство и не привело к значительной задержке обслуживания, но если бы портов на устройстве было бы много, то это уже могло сказаться на загрузке оперативной памяти (пункт 2.1.3).

В течение второго участка до достижения точки отказа сохраняется правило, что чем меньше очередь, тем больше потери. Системы с малой очередью

плохо реагируют на броски трафика, не имея возможности буферизировать пакеты в очередь, тем самым появляются дополнительные потери (рисунок 5.11). При этом до достижения точки отказа увеличенная очередь работает хорошо, то есть она снижает потери и не приводит к значительному росту задержки. При преодолении точки отказа очереди большего размера по-прежнему имеют несколько меньшие потери, но пропорционально увеличивают задержку пакетов в очереди. То есть администратору МЭ необходимо понимать, какие типы трафика проходят сквозь устройство. Если присутствует достаточное количество приложений реального времени, то значительно увеличивать максимальный объём очереди нельзя, так как это приведёт к значительной задержке обслуживания неприоритетного трафика, однако, это повысит потери при бросках трафика.

В течении третьего участка высокая интенсивность входящего потока пакетов (более 115-120% интенсивности обслуживания) нивелирует эффект нестабильной работы систем с очередями малого размера, они начинают меньше простаивать из-за случайных спадов интенсивности входящего потока пакетов. Это приводит к тому, что вне зависимости от размера очереди потери систем становятся почти одинаковыми. При этом задержки в системе с малыми очередями заметно ниже. Вопросы подбора оптимального размера очереди в работе не рассматриваются, но и без их широкого освящения видно, что при значительной перегрузке очереди большого размера вредны.

Общее влияние приоритизации обслуживания на производительность является достаточно значительным. При текущих условиях эксперимента эта задержка в очереди составляет около 16,4% от общей величины задержки неприоритетных пакетов в очереди в любой точке эксперимента. Относительное значение этой величины рассчитывается из соотношения интенсивности приоритетного потока пакетов к максимальной интенсивности обслуживания.

Стоит считать, что применение механизмов QoS в МЭ абсолютно оправдана в ситуациях нормального функционирования, то есть без перегрузки пакетами, а в режиме перегрузки их необходимо применять обдуманно, чтобы не слишком

сильно навредить неприоритетным потокам трафика. Учитывая, что в работе рассматривался самый «жёсткий» из алгоритмов выбора пакетов для постановки на обслуживание, стоит считать, что он не только «опасен» в условиях бросков приоритетного трафика, что приводит к оккупации канала, но также значительно повышает задержку неприоритетного трафика при перегрузках устройства и больших размерах очереди.

В режиме значительной перегрузки стоит снизить размер очереди, чтобы иметь возможность несколько снизить задержку пакетов без ухудшения величины потерь.

5.2.3. Результаты второго эксперимента

Второй эксперимент – первый сценарий поведения трафика

Для первого сценария исходные данные по входящим потокам пакетов заданы следующим образом: $\lambda_1 = 40 \cdot 10^3 = const$, $\lambda_2 = 110 \cdot 10^3, 120 \cdot 10^3, \ldots, 280 \cdot 10^3$ пакетов в секунду.

В таблице 5.1 представлены значения максимально возможной интенсивности обслуживания в зависимости правила фильтрации, на котором происходит совпадение и выход из системы. Расчёт проведён на основе исходных данных. Расчёты показали соответствие результатов.

Таблица 5.1 – Зависимость максимально возможной интенсивности обслуживания от количества правил фильтрации

Правило, на котором происходит совпадение	**Приблизительная возможная интенсивность обслуживания, пакетов в сек.**
29	≈ 244000
58	≈ 180000
116	≈ 118000
232	≈ 70000
928	≈ 20000

Из таблицы 5.1 видно, что при совпадении, начиная уже со 116 правила, система находится в значительной перегрузке уже с первой точки эксперимента. При этом при совпадении на 928 правиле система неспособна обслужить даже приоритетный поток пакетов.

В таблице 5.2 представлено значение математического ожидания времени обслуживания пакетов ядром обработки пакетов и базой правил в зависимости от количества правил фильтрации.

Таблица 5.2 – Зависимость времени обслуживания пакетов от количества правил фильтрации

Правило, на котором происходит совпадение	**Математическое ожидание времени обслуживания**	**Соотношение времени обслуживания создаваемое:**	
		Правилами фильтрации	**Ядром обработки пакетов**
29	$\approx 4,099$ мкс.	35,4%	64,6%
58	$\approx 5,548$ мкс.	52,2%	47,8%
116	$\approx 8,448$ мкс.	68,6%	31,4%
232	$\approx 14,249$ мкс.	81,4%	18,6%
928	$\approx 49,048$ мкс.	94,6%	5,4%

Заметим, насколько быстро меняется соотношение внутри величины задержки обслуживания. Величина задержки обслуживания, вносимая одним правилом фильтрации, незначительна, но с увеличением их количества вносимая ими задержка обслуживания становится значительной. Эти выводы также были озвучены в [20]. Напомним, что время обслуживания не зависит от приоритета пакета в условиях текущего эксперимента.

Если вычесть из общего времени обслуживания постоянную составляющую (ядро обработки пакетов), то математическое ожидание времени обслуживания от правил фильтрации будет увеличиваться линейно с увеличением их количества.

Рисунок 5.16 демонстрирует заполнение очереди неприоритетными пакетами ($QueueAvr_{unpriority}$) от интенсивности поступающего потока пакетов.

На графике видно, что системы с совпадением на 29 и 58 правилах заполняются при нарастании интенсивности входящего потока пакетов. Оставшиеся варианты системы изначально находятся в режиме перегрузки. Неприоритетные пакеты в системе с совпадением на 928 правиле выдавлены из системы приоритетными пакетами, но так как они на короткие промежутки времени попадают в неё, то их среднее количество в ней ненулевое. Также

напомним, что очередь в системе с 30 позициями ведёт себя несколько нестабильно, что было описано в предыдущем пункте 5.2.2.

На рисунке 5.17 продемонстрированы кривые задержек неприоритетных пакетов ($TAvrQperQueueServedPacket_{unpriority}$) от интенсивности поступающего потока пакетов. Кривая для системы с совпадением на 928 правиле не продемонстрирована по причине того, что среднее значение задержки слишком хаотичное из-за малой выборки, а точнее малого количества обслуженных неприоритетных пакетов. Количество обслуженных неприоритетных пакетов колеблется в пределах от 0 до 15 в каждой из реализаций.

Из рисунка 5.17 видно, что в системах с совпадением на правилах в начале базы правил (малых базах правил: 29, 58) задержка пакетов в очереди нарастает плавно. Системы с большим числом правил фильтрации, при совпадении на 116, 232 правилах показывают почти постоянную задержку, это связанно с тем, что их интенсивность обслуживания как минимум на 20% меньше интенсивности входящего потока пакетов с первых точек кривых. Напомним что при таких условиях система «уравновешивается» и при дальнейшем увеличении интенсивности поступающего потока пакетов её показатели производительности не изменяются, за исключением потерь.

По данным таблицы 5.3 виден значительный рост задержки приоритетных пакетов в очереди при увеличении количества правил фильтрации, что объясняется увеличением времени обслуживания пакетов и как следствие перегрузкой МЭ. График задержек для приоритетных пакетов не представлен.

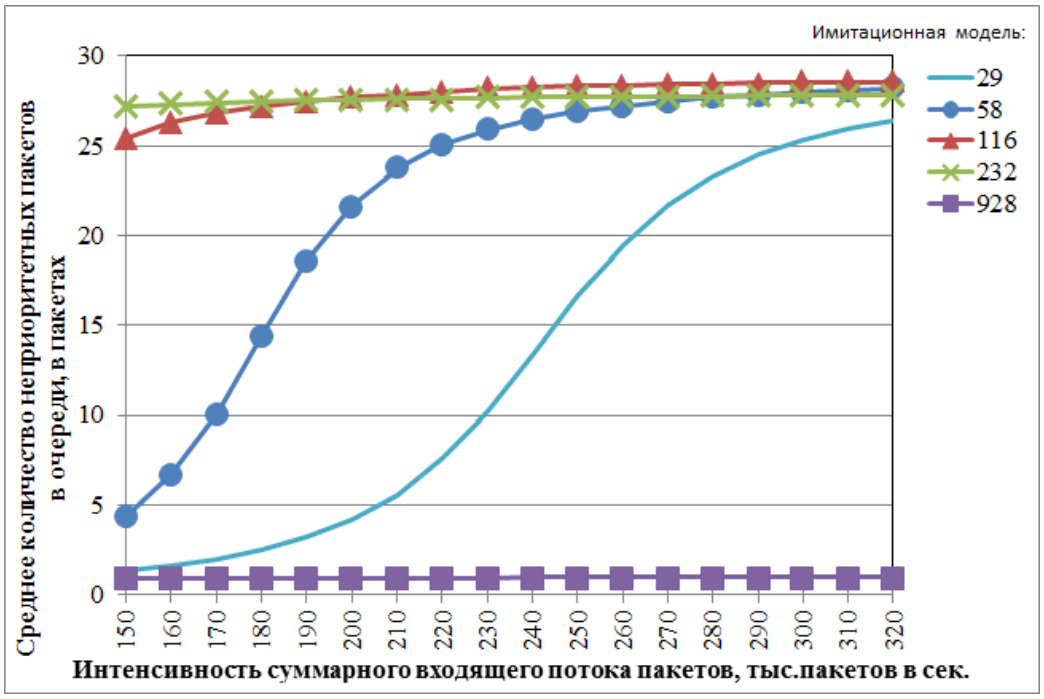
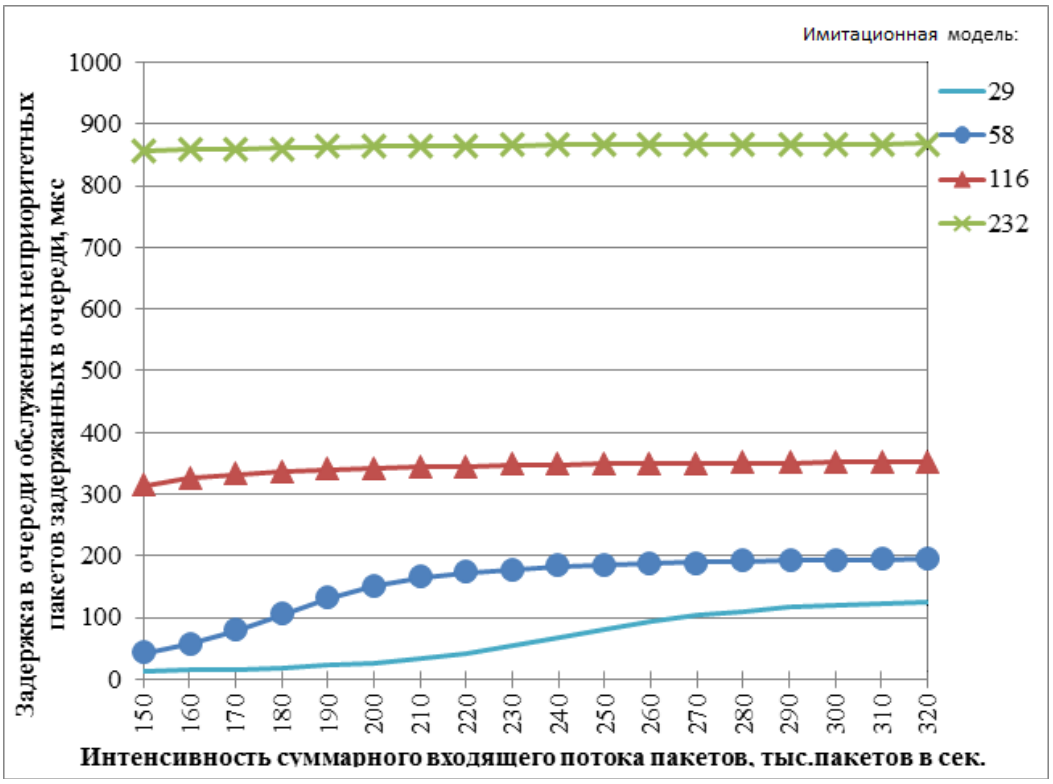


Рисунок 5.16 – Заполнение пакетной очереди неприоритетными пакетами (сценарий 1)



Рисунка 5.17 – Задержка в очереди обслуженных неприоритетных пакетов, задержанных в очереди (сценарий 1)

Таблица 5.3 – Время задержки в очереди обслуженных приоритетных пакетов задержанных в очереди

Правило, на котором происходит совпадение	**Время задержки в очереди обслуженных приоритетных пакетов, задержанных в очереди**	**Среднее количество приоритетных пакетов в очереди**
29	$\approx 4{,}902$ мкс.	$\approx 0{,}337$ пакетов
58	$\approx 7{,}132$ мкс.	$\approx 0{,}501$ пакетов
116	$\approx 12{,}76$ мкс.	$\approx 0{,}848$ пакетов
232	$\approx 33{,}167$ мкс.	$\approx 1{,}896$ пакетов
928	$\approx 1371{,}441$ мкс.	$\approx 28{,}963$ пакетов

Выводы по результатам второго эксперимента

Увеличение номера правила (позиции в базе), на котором происходит совпадение, в значительной мере увеличивает время обработки пакета. Эксплуатирующий персонал может пренебрегать этим, полагая, что время обработки одного правила незначительно по сравнению с другими операциями, проводимыми МЭ. Напомним, что характеристики обработки пакетов являются реальными и измерены на стенде, описанном в работе [20].

Само по себе увеличение времени обслуживания в значительной степени отражается на остальных показателях производительности устройства. Представим себе случай, при котором под определённую нагрузку было подобрано средство межсетевого экранирования с некоторым запасом по производительности. При длительной эксплуатации база правил фильтрации обычно увеличивается, если только среда функционирования не статичная. В свою очередь активные правила при некорректной эксплуатации могут попасть в конец базы правил, а это приведёт к тому, что устройство будет обслуживать меньше трафика и запаса производительности может не хватить.

Также крайне важным стоит считать тот факт, что при снижении максимальной пропускной способности будут изменяться соотношения интенсивностей приоритетного и неприоритетного входящих потоков с интенсивностью обслуживания. Это приведёт к тому, что на один обслуженный неприоритетный пакет придётся больше приоритетных пакетов и, следовательно,

неприоритетные пакеты будут задерживаться в очереди приоритетными пакетами больше времени. При максимальной интенсивности входящего трафика в 320 тыс. пакетов в секунду задержка в очереди неприоритетных пакетов будет вызвана приоритетными пакетами на: 16,4% при выходе с 29 правила (соответствует первому эксперименту), 22,2% при выходе с 58 правила, 33,7% при выходе с 116 правила, 57% для 232 правил. При выходе с 928 правила расчёт не возможен, обслуженных неприоритетных пакетов слишком мало.

5.3. Сравнительный анализ показателей производительности межсетевого экрана с включенными и выключенными механизмами приоритизации обслуживания трафика

5.3.1. Исходные данные

В рамках эксперимента рассматривается два сценария поведения трафика: первый – рост интенсивности потока неприоритетных пакетов при постоянном потоке приоритетных пакетов и второй – рост интенсивности приоритетного потока пакетов при постоянно потоке неприоритетных пакетов. Изменение параметров входящих потоков пакетов аналогично предыдущим экспериментам (подразделы 4.4, 5.1, 5.2).

В рамках эксперимента сравниваются результаты функционирования модели с включенными и выключенными механизмами приоритизации обслуживания трафика. При выключении приоритизации происходит следующее:

– приоритетные пакеты перестают вытеснять неприоритетные пакеты из очереди при её переполнении и сразу покидают МЭ необслуженными;

– при выборе пакета из очереди на обслуживание дисциплина обслуживания изменена на FIFO.

Для лучшего понимания представленного далее материала в таблице 5.4 представлены значения соотношений интенсивностей входящих потоков пакетов в каждой точке эксперимента.

Таблица 5.4 – Соотношения интенсивностей входящих потоков пакетов для обоих сценариев поведения трафика

Интенсивность входящих потоков пакетов, тыс. пакетов в секунду (процент от суммарной интенсивности входящего потока пакетов)	
Приоритетных/неприоритетных	**Неприоритетных/приоритетных**
40 (26.67 %)	110 (73.33 %)
40 (25.00 %)	120 (75.00 %)
40 (23.53 %)	130 (76.47 %)
40 (22.22 %)	140 (77.78 %)
40 (21.05 %)	150 (78.95 %)
40 (20.00 %)	160 (80.00 %)
40 (19.05 %)	170 (80.95 %)
40 (18.18 %)	180 (81.82 %)
40 (17.39 %)	190 (82.61 %)
40 (16.67 %)	200 (83.33 %)
40 (16.00 %)	210 (84.00 %)
40 (15.39 %)	220 (84.61 %)
40 (14.81 %)	230 (85.19 %)
40 (14.29 %)	240 (85.71 %)
40 (13.79 %)	250 (86.21 %)
40 (13.34 %)	260 (86.66 %)
40 (12.90 %)	270 (87.10 %)
40 (12.50 %)	280 (87.50 %)

В каждом из экспериментов один из типов трафика по отношению к другому занимает от 12.5 до 27% объёма входящего трафика.

5.3.2. Результаты моделирования

Первый сценарий поведения трафика. Для первого сценария исходные данные по входящим потокам пакетов заданы следующим образом: $\lambda_1 = 40 \cdot 10^3 = const$, $\lambda_2 = 110 \cdot 10^3, 120 \cdot 10^3, \ldots, 280 \cdot 10^3$ пакетов в секунду.

При рассмотрении результатов эксперимента было выявлено, что графические интерпретации данных не обеспечивают хорошее восприятие результатов. Результаты будут представлены в табличном виде.

В таблице 5.5 представлены результаты сравнения потерь пакетов для 1 сценария поведения трафика.

Таблица 5.5 – Сравнение потерь пакетов для 1 сценария поведения трафика

Интенсивность суммарного входящего потока пакетов	**Потери с QoS, в %**		**Потери без QoS, в %**	
	приор.	**не приор.**	**приор.**	**не приор.**
150	0	0	0	0
160	0	0	0	0
170	0	0	0	0
180	0	0,004	0,001	0,003
190	0	0,016	0,003	0,003
200	0	0,059	0,013	0,0121
210	0	0,194	0,047	0,0468
220	0	0,564	0,158	0,1574
230	0	1,412	0,464	0,4623
240	0	3,002	1,168	1,1648
250	0	5,425	2,502	2,503
260	0	8,482	4,559	4,5559
270	0	11,848	7,179	7,1753
280	0	15,252	10,084	10,0913
290	0	18,531	13,070	13,0709
300	0	21,610	15,975	15,9667
310	0	24,491	18,730	18,7257
320	0	27,177	21,314	21,3276

В таблице 5.6 представлены результаты сравнения потерь и времени нахождения пакетов в очереди для первого сценария поведения трафика.

Таблица 5.6 – Сравнение потерь и времени нахождения пакетов для первого сценария поведения трафика

Интенсивность суммарного входящего потока пакетов	**Задержка в очереди с QoS, в мкс**		**Задержка в очереди без QoS, в мкс**	
	приор.	**не приор.**	**приор.**	**не приор.**
150	4,905	12,741	10,651	10,652
160	4,906	14,256	11,917	11,918
170	4,904	16,186	13,531	13,532
180	4,907	18,715	15,650	15,643
190	4,904	22,081	18,460	18,466
200	4,905	26,791	22,413	22,409
210	4,904	33,323	27,894	27,897
220	4,903	42,367	35,520	35,521
230	4,905	53,999	45,365	45,360
240	4,904	67,521	56,828	56,836

Интенсивность суммарного входящего потока пакетов	Задержка в очереди с QoS, в мкс		Задержка в очереди без QoS, в мкс	
	приор.	не приор.	приор.	не приор.
250	4,905	81,276	68,493	68,497
260	4,903	93,452	78,823	78,834
270	4,904	103,242	87,132	87,139
280	4,904	110,664	93,399	93,398
290	4,904	116,132	98,004	97,991
300	4,905	120,086	101,348	101,361
310	4,905	123,060	103,871	103,865
320	4,906	125,316	105,783	105,790

По данным, представленным в таблице 5.5, 5.6 видно, что без применения механизмов приоритизации обслуживания потери и задержки обоих потоков пакетов стремятся сравняться в относительных значениях. При активной приоритизации трафика приоритетные пакеты получают значительно повышение качества обслуживания за счёт ухудшения параметров обслуживания неприоритетных пакетов.

Стоит понимать, что общее время занятия системы почти не зависит от наличия приоритизации обслуживания, то есть при переполнении система не может быть занята 31 секунду из 30. Значит при увеличении процента трафика, требующего приоритетного обслуживания, качество обслуживания неприоритетного трафика будет ухудшаться. В конечном итоге может произойти отказ обслуживания и лавинообразная перепосылка трафика (у протоколов с гарантированной доставкой).

5.4. Анализ результатов экспериментов

Результаты при постоянной длительности обслуживания. Сравнение времени нахождения в очереди обслуженных пакетов (рисунки 5.4, 5.8) при случайной длительности обслуживания, распределённой по экспоненциальному закону вероятностей, и при постоянной длительности обслуживания показало, что при отсутствии потерь пакетов и при постоянной длительности обслуживания задержка в очереди примерно в два раза меньше, чем при случайной

длительности обслуживания. Это соответствует результатам, приведенным в [141] и подтверждает корректность функционирования имитационной модели.

Имитационная модель поддерживает ряд дополнительных функций, которые не освещались в диссертации. Доступна возможность принять время между приходом пакетов за постоянную величину. Доступна возможность принять время между приходом пакетов и время обработки пактов, за случайную величину распределённые по равномерному закону распределения. Доступна возможность принять для каждого из входящих потоков (приоритетного, неприоритетного) своё собственное срабатывающее правило фильтрации.

Изменение максимального объёма очереди (буфера памяти, выделяемого на хранение пакетов) показало, что её увеличение положительно сказывается на показателях обслуживания до тех пор, пока система не войдёт в режим перегрузки. С переходом в режим перегрузки увеличенный объём очереди создаёт для пакетов, попавших в очередь, значительные задержки, при этом не давая более никаких положительных эффектов. Увеличенная очередь в режиме перегрузки может быть полезна только в том случае, если нахождение в режиме перегрузки будет непродолжительным.

Приоритизация обслуживания создаёт дополнительную задержку для неприоритетных пакетов с вязи с тем, что каждый приоритетный пакет будет вставать перед неприоритетным пакетом на обслуживание. Относительное значение этой величины рассчитывается из соотношения интенсивности приоритетного потока пакетов к максимальной интенсивности обслуживания.

В случае прогнозирования появления продолжительной по времени перегрузки на определённом участке сети рекомендуется уменьшить доступный объём очереди, чтобы обслуженный трафик имел меньшие задержки. Потери пакетов при продолжительной перегрузке как при большом, так и при малом максимального объёме очереди быстро выравниваются и перестают зависеть от максимального объёма очереди. Эта рекомендация является общей для всех типов оборудования.

Наличие приоритизации обслуживания в этой ситуации перегрузки будет усугублять показатели обслуживания. В зависимости от обязательств оператора по передаче приоритетного трафика, рекомендуется временно приостанавливать работу механизмов QoS вследствие того, что значительный сброс и задержка неприоритетных пакетов, скорее всего, приведёт к значительному росту перепосылки пакетов протоколами с гарантированной доставкой и дальнейшему распространению «бутылочного» горлышка по сети.

Изменение позиции правила, на котором происходит выход из системы (в том числе это можно считать и увеличением самой базы правил) показало, что даже незначительное увеличение позиции этого правила в значительной степени влияют на показатели производительности МЭ. МЭ в значительной степени подвержены эффекту «бутылочного горлышка.

Изначально использованное оборудование МЭ при росте базы правил и отсутствии её оптимизации приведёт к ухудшениям параметров обслуживания. Под оптимизацией базы правил понимается процесс, при котором часто совпадающие правила необходимо пытаться поставить как можно раньше и при этом не нарушить логику работы базы правил. Для снижения эффекта рекомендуется выполнять оптимизацию базы правил после её изменения или изменения количества или типов информации, передаваемой от/к субъектам информационного обмена.

Ряд представленных выводов опубликован автором в работах [142, 143].

Выводы раздела

1. Результаты имитационного моделирования процессов функционирования межсетевого экрана при условии принятия времени обслуживания за постоянную и случайную величины (экспоненциальный закон) показали, что соотношения задержек пакетов обоих приоритетов в очереди, обоснованные в теории массового обслуживания, соблюдаются при функционировании межсетевого экрана в режиме без перегрузок.

При функционировании межсетевого экрана с приоритизацией обслуживания трафика при времени обслуживания, распределённом по

экспоненциальному закону, среднее время нахождения неприоритетных пакетов в очереди становится меньше, чем при постоянном времени обслуживания, что обуславливается повышенной вероятностью частичного освобождения очереди.

2. Результаты моделирования показателей производительности межсетевого экрана при применении механизма приоритизации обслуживания трафика показали, что увеличение объема очереди до наступления перегрузки повышает устойчивость модели к резким броскам трафика, снижая потери, а после наступления перегрузки приводит лишь к дополнительным задержкам в очереди неприоритетных пакетов. Так, при увеличении объема очереди от 30 до 960 пакетов при загрузке системы на 90% показали снижение потерь суммарного потока трафика от 2,5% до нулевого уровня при повышение средней задержки неприоритетных пакетов в очереди на 17% (с 42,4 до 49,9 мкс). В режиме перегрузки изменения процента потерь не наблюдалось, а величина средняя задержка неприоритетных пакетов увеличилась в 36 раз (с 125,3 до 4684,3 мкс).

3. Результаты моделирования показателей производительности межсетевого экрана при изменении среднего количества правил от 29 до 928, которое необходимо пройти пакету при обслуживании, демонстрирует увеличение времени обслуживания пакетов почти в 12 раз. Продолжительная эксплуатация базы правил фильтрации без её оптимизации приводит к значительному ухудшению показателей качества обслуживания пакетов, обслуживаемых МЭ.

4. Введение приоритизации обслуживания чувствительного к задержкам трафика во входных очередях межсетевых экранов позволяет избежать значительных потерь приоритетных пакетов при функционировании в режиме перегрузки и существенно (в разы) снизить задержки приоритетных пакетов в очереди в нормальном режиме функционирования и в режиме перегрузки. Так, в условиях 30% перегрузки при доле входящего потока приоритетных пакетов в 15% от общего потока, включение механизмов приоритизации обслуживания снижает:

– потери пакетов приоритетного потока с 23% до нулевого уровня при повышении потерь пакетов неприоритетного потока всего на 4 %;

– среднее время нахождения приоритетного пакета в очереди более, чем в 20 раз (с 105 до 5 мкс) при увеличении этой величины для неприоритетных пакетов только на 19%.

5. Выявлены зависимости показателей качества обслуживания трафика межсетевыми экранами, функционирующими в условиях приоритизации обслуживания критичного к задержкам трафика, от длины очереди, длительности обслуживания и правил фильтрации при функционировании в режиме без перегрузки и с перегрузкой. При эксплуатации межсетевого экрана рекомендуется учитывать следующие зависимости:

– при функционировании межсетевого экрана в режиме без перегрузок:

• увеличение объёма очереди приводит к сглаживанию флуктуаций входящего трафика как при включенной, так и выключенной приоритизации обслуживания;

• активация приоритизации обслуживания при отсутствии перегрузок не даёт значительного снижения потерь пакетов приоритетного трафика, особенно при увеличении объёма очереди;

• активация приоритизации обслуживания пакетов при отсутствии перегрузок обеспечивает значительное снижение задержки в очереди для приоритетных пакетов в диапазоне 50-100% загрузки межсетевого экрана.

– при функционировании межсетевого экрана в режиме перегрузки:

• увеличение объёма очереди приводит только к увеличению задержек обслуженных пакетов, но не обеспечивает снижения потерь, как при случае с функционированием в режиме без перегрузок;

• применение приоритизации обслуживания во входной очереди межсетевого экрана позволяет эффективно поддерживать значения потерь и задержки приоритетных пакетов, что особенно актуально для протоколов сигнализации и управления, не имеющих приоритизации обслуживания по умолчанию.

– использование алгоритмов с «жёсткой» приоритизацией, таких как PRIOR (Linux), LLQ (Cisco) и др. способствует «выдавливанию» пакетов с низким приоритетом из очереди пакетами с высоким приоритетом. При использовании подобных алгоритмов необходимо обоснованно выбирать типы трафика, которые будут иметь высший приоритет, так как рост интенсивности этих типов трафика может привести к отказу обслуживания неприоритетных типов трафика;

– периодическое перестроение базы (списка) правил фильтрации с целью снижения порядкового номера наиболее часто срабатывающего правила позволяет снизить среднюю задержку обслуживания пакетов.

ЗАКЛЮЧЕНИЕ

Диссертационная работа посвящена разработке метода оценки влияния приоритизации обслуживания трафика во входных очередях межсетевых экранов на показатели их производительности. Разработанный метод в отличие от существующих методов позволил получить численные оценки показателей производительности межсетевых экранов, функционирующих в условиях приоритизации обслуживания трафика на их входах.

В процессе диссертационных исследований были получены следующие основные результаты:

1. Разработка метод оценки показателей производительности межсетевых экранов типа NetFilter/IPTables при функционировании в условиях приоритизации обслуживания трафика на их входах. В основу метода положены математическая и имитационная модели.
2. Для обеспечения требований к показателям качества обслуживания трафика рекомендуется учитывать следующие зависимости:
   - при функционировании межсетевого экрана в режиме без перегрузок:
     - увеличение объёма очереди приводит к сглаживанию флуктуаций входящего трафика как при включенной, так и выключенной приоритизации обслуживания;
     - активация приоритизации обслуживания при отсутствии перегрузок не даёт значительного снижения потерь пакетов приоритетного трафика, особенно при увеличении объёма очереди;
     - активация приоритизации обслуживания пакетов при отсутствии перегрузок обеспечивает значительное снижение задержки в очереди для приоритетных пакетов в диапазоне 50-100% загрузки межсетевого экрана.

– при функционировании межсетевого экрана в режиме перегрузки:

• увеличение объёма очереди приводит только к увеличению задержек обслуженных пакетов, но не обеспечивает снижения потерь, как при случае с функционированием в режиме без перегрузок;

• применение приоритизации обслуживания во входной очереди межсетевого экрана позволяет эффективно поддерживать значения потерь и задержки приоритетных пакетов, что особенно актуально для протоколов сигнализации и управления, не имеющих приоритизации обслуживания по умолчанию.

– использование алгоритмов с «жёсткой» приоритизацией, таких как PRIOR (Linux), LLQ (Cisco) и др. способствует «выдавливанию» пакетов с низким приоритетом из очереди пакетами с высоким приоритетом. При использовании подобных алгоритмов необходимо обоснованно выбирать типы трафика, которые будут иметь высший приоритет, так как рост интенсивности этих типов трафика может привести к отказу обслуживания неприоритетных типов трафика;

– периодическое перестроение базы (списка) правил фильтрации с целью снижения порядкового номера наиболее часто срабатывающего правила позволяет снизить среднюю задержку обслуживания пакетов.

3. Результаты диссертации использованы в ЗАО «Научно-производственное предприятие «Безопасные информационные технологии», а также в учебном процессе на кафедре Сетей связи и систем коммутации МТУСИ, что подтверждено соответствующими актами (приложение Ж).

Список сокращений и условных обозначений

Сокращение		Расшифровка
ACL	–	access list (рус. лист контроля доступа)
ASIC	–	application-specific integrated circuit (рус. Интегральная схема специального назначения,
CAM	–	context addressable memory (рус. ассоциативное запоминающее устройство)
DSCP	–	differentiated services code point (рус. поле кода дифференцированных услуг)
FIFO	–	first-in, first-out (рус. первый пришёл первый обслужен)
GPSS	–	general purpose simulation system (рус. система моделирования общего назначения)
LLQ	–	low latency queue (рус. очередь с малой задержкой)
QoS	–	quality of service (рус. качество обслуживания)
RAM	–	random access memory (рус. запоминающее устройство с произвольной выборкой)
RED	–	random early detection
TCAM	–	ternary context addressable memory (рус. троичное ассоциативное запоминающее устройство)
ToS	–	type of service (рус. поле тип сервиса)
UNI	–	user network interface (рус. сетевой интерфейс пользователя)
ГСЧ	–	генератор случайных чисел
ЛВС	–	локальная вычислительная сеть (англ. local area network, LAN)
МСЭ	–	международный союз электросвязи (англ. International telecommunication union, ITU)
МЭ	–	межсетевой экран (англ. firewall)
ПО	–	программное обеспечение
СМО	–	система массового обслуживания
СУР	–	система уравнений равновесия
УЕВ	–	условная единица времени

Список литературы

1. Cisco Visual Networking Index (VNI) Forecast 2016-2021 [Электронный ресурс] // Cisco Systems. Inc. [сайт]. – 2017. – Режим доступа: https://www.cisco.com/c/en/us/solutions/service-provider/vni-network-traffic-forecast/infographic.html (дата обращения 15.03.2018).
2. Описание оперативной группы IMT-2020 [Электронный ресурс] // МСЭ [сайт]. – 2015. – Режим доступа: http://www.itu.int/ru/ITU-T/focusgroups/imt-2020/Pages/default.aspx (дата обращения 15.03.2018).
3. Report on Standards Gap Analysis [Text] / FG IMT-2020 // ITU-T, 2015. 172 p.
4. IMT Vision – Framework and overall objectives of the future development of IMT for 2020 and beyond [Text] / ITU-R M.2083-0 // ITU-R, 2015. 19 p.
5. Росляков, А.В. Будущие сети (Future networks) [Текст] / А.В. Росляков, С.В. Ваняшин. – Самара: ПГУТИ, 2015. – 274 с.
6. Росляков, А.В. Первые рекомендации МСЭ-Т о будущий сетях [Текст] / А.В. Росляков // Вестник связи. – 2014. – №10. – С.29-34.
7. Росляков, А.В. Future Networks. Версия МСЭ-Т. Часть 1 [Текст] / А.В. Росляков. ИнформКурьер-Связь. – 2014. – №12. – С.68-70.
8. Росляков, А.В. Future Networks. Версия МСЭ-Т. Часть 12 [Текст] / А.В. Росляков. – ИнформКурьер-Связь. – 2015. – №1-2. – С.62-63.
9. Информационное сообщение ФСТЭК России от 28 апреля 2016 № 240/24/1986.
10. End-user multimedia QoS categories [Text] / ITU-T G.1010 // ITU-R, 2001. – 10 p.
11. Acharya, S. Simulation study of firewalls to aid improved performance [Text] / S. Acharya, J. Wang, Z. Ge, T. Znati, and. A. Greeberg // ANSS '06 Proceedings of the 39th annual Symposium on Simulation, 2006. – pp. 18-26.
12. Al-Shaer, E. Conflict classification and analysis of distributed firewall policies [Text] / E. Al-Shaer, H. Hamed, R. Boutaba, M. Hasan // IEEE J. Sel. Areas Commun. – 2005. – vol. 23, № 10. – pp. 2069–2084.
13. Al-Shaer, E. Modeling and management of firewall policies [Text] / E. Al-Shaer, H. Hamed // IEEE Trans. Network Service Management. – 2004. – vol. 1, № 1. – pp. 2-10.

14. Gusev, M. Architecture of a Identity Based Firewall System [Text] / M. Gusev, N. Stojanovski // International Journal of Network Security & Its Applications (IJNSA). – 2011. – Vol. 3, № 4. – pp. 23-31.
15. Kaur, H. Implementation of Portion Approach in Distributed Firewall Application for Network Security Framework [Text] / H. Kaur, E. Omid Mahdi Ebadati, M. Afshar Alam // IJCSI International Journal of Computer Science Issues. – 2011. – Vol. 8, Issue 6, № 2. – pp. 207-217.
16. Liu, A.X. Diverse firewall design [Text] / A.X. Liu, M.G. Gouda // IEEE Trans. Parallel Distrib. Syst. – 2008. – Vol. 19, № 9. – pp. 1237-1251.
17. Liu, A.X. Structured firewall design [Text] / A.X. Liu, M.G. Gouda // Computer Networks: The Int'l J. Computer and Telecommun. Networking. – 2007 – № 51. – pp. 1106–1120.
18. Meiners, C.R. Topological transformation approaches to optimizing tcam-based packet classification systems [Text] / C.R. Meiners, A. X. Liu, E. Torng // Proc. 2009 International Joint Conference on Measurement and Modeling of Computer Systems. – Michigan, 2009. – pp. 73–84.
19. Salah, K. Queueing analysis of network firewalls [Text] / K. Salah // Global Telecommunications Conference (GLOBECOM 2010). – Miami, 2010. – pp. 1–5.
20. Salah, K. Performance Modeling and Analysis of Network Firewalls [Text] / K. Salah, K. Elbadawi, R. Boutaba // IEEE Translations on network and service. – 2012. – Vol. 9, № 1. – pp. 12-21.
21. Salah, K. Implementation and experimental performance evaluation of a hybrid interrupt-handling scheme [Text] / K. Salah, A. Qahtan // Int'l J. Computer Commun. – 2009. – Vol. 32, № 1. – pp. 179–188.
22. Барабанов, В.Ф. Построение «Периметровой» стратегии защиты информационных систем персональных данных на основе настроек межсетевого экрана netfilter [Текст] / В.Ф. Барабанов, Е.С. Пашковская, М.Е. Пашковский // Вестник ВГТУ. – 2012. – №4. – С.39-43.

23. Богораз, А.Г. Методика тестирования и оценки межсетевых экранов [Текст] / А.Г. Богораз, О.Ю. Пескова // Известия ЮФУ: Технические науки. – 2013. – №12. – С.148-156.
24. Гайдамака, Ю.В. Об одной системе массового обслуживания с активным управлением очередью [Текст] / Ю.В. Гайдамака, А.Г. Масленников // Вестник РУДН. Серия: математика. информатика. физика. – 2013. – №4. – С. 56-64.
25. Коломойцев, В.С. Выбор варианта построения многоуровневого защищенного доступа к внешней сети [Текст] / В.С. Коломойцев // Научно-технический вестник информационных технологий. Механики и оптики. – 2016. – №1. – С.115-121.
26. Леваков, А.К. Задачи оценки показателей, определяющих качество функционирования телекоммуникационных сетей [Текст] / А.К. Леваков, А.В. Федоров, Н.А. Соколов // Электросвязь. – 2015. – № 6. – С. 24-27.
27. Леваков А.К. Влияние характера входящего потока IP-пакетов на допустимую загрузку узла коммутации [Текст] / А.К. Леваков, А.В. Федоров, Н.А. Соколов // Труды ЦНИИС. Санкт-Петербургский филиал. – 2016. – Том 1, № 2 (3). – С. 21-25.
28. Назаров, А. Н. Модели и методы исследования процессов функционирования и оптимизации построения сетей связи следующего поколения при произвольных распределениях поступления и обслуживания пакетов различных классов качества [Текст] / А.Н. Назаров, К.И. Сычев // T-Comm: Телекоммуникации и транспорт. – 2011. – №7 – С. 112-116.
29. Назаров, А.Н. Модели и методы исследования процессов функционирования узлов коммутации сетей связи следующего поколения при произвольных распределениях поступления и обслуживания заявок различных классов качества [Текст] / А.Н. Назаров, К.И. Сычев // T-Comm: Телекоммуникации и транспорт. – 2012. – №7. – С.135-140.
30. Самуйлов, К.Е. Оценка времени установления сессии между пользователями при наличии межсетевого экрана [Текст] / К.Е. Самуйлов, А.Ю. Ботвинко,

Э.Р. Зарипова // Вестник РУДН. Серия: Математика. информатика. физика. – 2016. – №1. – С.59-66.

31. Самуйлов, К.Е. Математическая модель работы межсетевого экрана для мультимедийного трафика [Текст] / К.Е. Самуйлов, А.Ю. Ботвинко // T-Comm: Телекоммуникации и транспорт. – 2015. – №12. – С.56-60.
32. Шабуров, А.С. Реализация отказоустойчивого распределенного межсетевого экрана [Текст] / А.С. Шабуров, Р.Б. Рашевский // Вестник ПНИПУ. Электротехника. Информационные технологии. Системы управления. – 2014. – №11. – С.129-136.
33. Шелухин, О.И. Моделирование информационных систем. Учебное пособие для вузов. – 2-е изд., перераб. и доп. [Текст] / О.И. Шелухин. – М.: Горячая линия – Телеком, 2012. – 516 с.
34. Ingham, K. A History and Survey of Network Firewalls [Электронный ресурс] / K. Ingham, S. Forrest // University of New Mexico [сайт]. – 2012. – Режим доступа: http://www.cs.unm.edu/~treport/tr/02-12/firewall.pdf (дата обращения 15.03.2018).
35. Schimmel, J. A historical look at firewall technologies [Text] / J. Schimmel // Login. – 1997. – Vol. 22, № 7. – pp. 1-42.

Раздел 1.

36. Guidelines on Firewalls and Firewall Policy [Text] / NIST SP 800-41 Rev. 1 // USA NIST, 2009. – 48 p.
37. Национальный стандарт Российской Федерации. Информационная технология. Методы и средства обеспечения безопасности. Безопасность сетей. Часть 1. Обзор и концепции [Текст] / ГОСТ Р ИСО/МЭК 27033-1-2011 // РОССТАНДАРД России, 2011. – 66 с.
38. Page, B. A report of the internet worm [Электронный ресурс] / B. Page // University of Lowell, 1998 [сайт]. – Режим доступа: http://www.ee.ryerson.ca/~elf/hack/iworm.html (дата обращения 15.03.2018).
39. Fisher, D. Report of computer virus incident at Ames [Электронный ресурс] / D. Fisher, H. Finger, W. Kramer, J. Stanley // NASA Technical Report, 1998

[сайт]. – Режим доступа: http://foofus.com/amuse/public/Morris_Worm_Incident_Report_1. pdf (дата обращения 15.03.2018).

40. Report of IAB Workshop on Security in the Internet Architecture [Text] / RFC 1636 // IETF. 8-10.02, 1994. – 52 p.
41. Мусатов, В.К. Анализ стандартов и рекомендаций по обеспечению информационной безопасности сетей передачи данных [Текст] / В.К. Мусатов // Международной научно-технической конференций «INTERMATIC – 2012» Фундаментальные проблемы радиоэлектронного приборостроения: матер. конф. часть 6. – М.: Энергоатомиздат, 2012. – С. 93-98.
42. Мусатов, В.К. Анализ информационной безопасности в сетях связи [Текст] / В.К. Мусатов, С.А. Васильев // Телекоммуникационные и вычислительные системы: труды конференции – М.: ООО «Информпресс-94», 2011. – С. 37-38.
43. История МСЭ [Электронный ресурс] // МСЭ [сайт]. – Режим доступа: http://www.itu.int/ru/about/Pages/history.aspx (дата обращения 15.03.2018).
44. Коротко о МСЭ-Т [Электронный ресурс] // МСЭ-Т [сайт]. – Режим доступа: http://www.itu.int/ru/ITU-T/about/Pages/default.aspx (дата обращения 15.03.2018).
45. Описания и ссылки на страницы исследовательских групп МСЭ-Т [Электронный ресурс] // МСЭ-Т [сайт]. – Режим доступа: http://www.itu.int/en/ITU-T/studygroups/2017-2020/Pages/default.aspx (дата обращения 15.03.2018).
46. Мусатов, В.К. Анализ тенденций развития рекомендаций МСЭ-Т по информационной безопасности [Текст] / В.К. Мусатов // T-Comm: Телекоммуникации и транспорт. – 2013. – №7. – С. 93-96.
47. Об ИСО [Электронный ресурс] // ИСО [сайт]. – Режим доступа: https://www.iso.org/ru/about-us.html (дата обращения 15.03.2018).
48. About the IETF [Электронный ресурс] // IETF [сайт]. – Режим доступа: http://www.ietf.org/about/ (дата обращения 15.03.2018).
49. About IEEE [Электронный ресурс] // IEEE [сайт]. – Режим доступа: http://www.ieee.org/about/index.html (дата обращения 15.03.2018).

50. Государственные функции и услуги, предоставляемые ФСТЭК России [Электронный ресурс] // ФСТЭК России [сайт]. – Режим доступа: http://fstec.ru/deyatelnost/gosudarstvennye-funktsii-i-uslugi (дата обращения 15.03.2018).
51. Общая информация о ЦЛСЗ ФСБ России [Электронный ресурс] // ФСБ России [сайт]. – Режим доступа: http://clsz.fsb.ru/ (дата обращения 15.02.2018).
52. О РОССТАНДАРТЕ [Электронный ресурс] // РОССТАНДАРТ [сайт]. – Режим доступа: https://www.gost.ru/portal/gost//home/about (дата обращения 15.03.2018).
53. About NIST [Электронный ресурс] // NIST [сайт]. – Режим доступа: http://www.nist.gov/public_affairs/nandyou.cfm (дата обращения 15.03.2018).
54. Мусатов, В.К. Исследование вопросов фильтрации DDoS атак в контексте облачных вычислений [Текст] / В.К. Мусатов // Международная конференция «Радиоэлектронные устройства и системы для инфокоммуникационных технологий» RES-2013: доклады. – М.: ООО «Информпресс-94». – С. 209-212.
55. Мусатов, В.К. Моделирование производительности средств межсетевого экранирования [Текст] / В.К. Мусатов // Телекоммуникационные и вычислительные системы: труды конференции – М.: ООО «Информпресс-94», 2012. – С. 42-43.
56. Мусатов, В.К. Анализ возможности применения динамических списков фильтрации в средствах межсетевого экранирования [Текст] / В.К. Мусатов // Международной научно-технической конференций «INTERMATIC – 2012» Фундаментальные проблемы радиоэлектронного приборостроения: матер. конф. часть 4 – М.: Энергоатомиздат, 2013. – С. 190-195.
57. Мусатов, В.К. Обоснование эффективности применения автокоррекции баз правил фильтрации в средствах межсетевого экранирования [Текст] / В.К. Мусатов // T-Comm: Телекоммуникации и транспорт. – 2014. – №8. – С. 68-72.
58. JUNOS TRIO Programmable Silicon Optimized for the Universal Edge // Juniper Networks. Inc, 2009. – 12 p.

59. The Cisco Flow Processor: Cisco's Next Generation Network Processor Solution Overview [Электронный ресурс] // Cisco Systems. Inc. [сайт]. – Режим доступа: https://www.cisco.com/c/en/us/products/collateral/routers/asr-1000-series-aggregation-services-routers/solution_overview_c22-448936.html (дата обращения 15.03.2018).
60. Pagiamtzis, K. Content-Addressable Memory (CAM) Circuits and Architectures: A Tutorial and Survey [Text] / K. Pagiamtzis, Ali. Sheikholeslami // IEEE Journal of solid-state circuits. – 2006 – Vol. 41, №. 3. – pp. 712-727.
61. Noda, H. A 143 MHz 1.1W4.5 Mb dynamic TCAM with hierarchical searching and shift redundancy architecture [Text] / H. Noda, K. Inoue, M. Kuroiwa and etc. // IEEE Int. Solid-State Circuits Conf. (ISSCC). – San Francisco, 2004. – pp. 208-209.
62. CAM (Content Addressable Memory) VS TCAM (Ternary Content Addressable Memory) [Электронный ресурс] // Cisco Systems. Inc. [сайт]. – Режим доступа: https://supportforums.cisco.com/t5/network-infrastructure-documents/cam-content-addressable-memory-vs-tcam-ternary-content/ta-p/3107938 (дата обращения (15.03.2018).
63. Taylor, D.E. Survey and taxonomy of packet classification techniques [Text] / D.E. Taylor // ACM Computing Surveys. – 2005. – Vol. 37, № 3. – pp. 238-275.
64. Масленников, А.Г. Разработка метода обработки трафика в очередях маршрутизаторов мультисервисной сети на основе нечёткой логики. Диссертация на соискание учёной степени кандидата технических наук. [Текст]: дис… канд. тех. наук: 05.12.13: / А.Г. Масленников. – Самара, 2016.
65. mHealth Wearables Boost Patient Healthcare Both Inside and Outside the Hospital [Электронный ресурс] // Веб-сайт ABI research. Inc. [сайт]. – Режим доступа: https://www.abiresearch.com/press/mhealth-wearables-boost-patient-healthcare-both-in/ (дата обращения 15.03.2018).
66. Соколов, А.Н. Однолинейные системы массового обслуживания: учебное пособие [Текст] / А.Н. Соколов, Н.А. Соколов. – СПБ: Изд-во «Теледом» ГОУВПО СПбГУТ, 2010. – 122 с.

67. Network performance objectives for IP-based services [Text] / ITU-T Y.1541 // ITU-T, 2011. – 66 p.
68. Labrecque, M. The case for hardware Transactional memory in software packet processing [Text] / M. Labrecque, J. Gregory Steffan // Architectures for Networking and Communications Systems (ANCS). – La Jolla, 2010. – 11 p.
69. Nakajima, Y. Software packet processing and Hardware packet processing: The advantages and disadvantages [Электронный ресурс] / Y. Nakajima // Japan Network Operators Group [сайт]. – 2016. – Режим доступа: https://www.janog.gr.jp/en/attach/janog37/janog37-nakajima.pdf (дата обращения 15.03.2018).
70. Ebisawa, K. Software packet processing and Hardware packet processing: Architecture [Электронный ресурс] / K. Ebisawa // Japan Network Operators' Group [сайт]. – 2016. – Режим доступа: https://www.janog.gr.jp/en/attach/janog37/janog37-ebisawa.pdf (дата обращения 15.03.2018).
71. Networkers Cisco Router Architecture Session 601 [Text]. – Cisco Systems, Inc., 1999. – 44 p.
72. Кучерявый, С.А. Управление трафиком и качество обслуживания в сети интернет [Текст] / С.А. Кучерявый. – М.: Наука и техника, 2004. – 336 с.
73. Шринивас, В. Качество обслуживания в сетях IP / В. Шринивас. – Изд-во Вильямс, 2003. – 368 с.
74. Bridges and Bridged Networks [Text] / IEEE Std 802.1Q // IEEE Computer Society, 2014. – 1367p.
75. New Terminology and Clarifications of Diffserv [Text] / RFC 3260 // IETF, 2002. – 10 p.
76. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers [Text] / RFC 2474 // IETF, 1998. – 20 p.
77. Бочаров, П.П. Теория массового обслуживания [Текст] / П.П. Бочаров, А.В. Печинкин – М: Изд-во РУДН. 1995. – 529 с.

78. TCP Slow Start. Congestion Avoidance. Fast Retransmit and Fast Recovery Algorithms [Text] / RFC 2001 // IETF, 1997. – 5 p.
79. The addition of Explicit Congestion Notification (ECN) to IP [Text] / RFC 3168 // IETF, 2001. – 63 p.
80. Floyd, S. Random Early Detection Gateways for Congestion Avoidance [Text] / S. Floyd, V. Jacobson // IEEE/ACM Transitions on Networking. – 1993. – Vol. 1, № 4. –pp. 387-413.
81. M.S.P. Moraes, A. Cisco Firewalls (Cisco Press Networking Technology) [Text] / A. M.S.P. Moraes. – Cisco Press, 2011. – 912 p.
82. Woodberg, B. Juniper SRX Series [Text] / Brad Woodberg, Rob Cameron. – O'Reilly Media, 2013. – 1020 p.
83. Колисниченко, Д.Н. Серверное применение Linux – 3-е изд., перераб и доп. [Текст] / Д.Н. Колисниченко. – СПб.: БХВ-Петербург, 2011. – 528 с.
84. Мусатов, В.К. Анализ механизмов фильтрации в межсетевых экранах [Текст] / В.К. Мусатов // Телекоммуникационные и вычислительные системы: труды конференции. – М.: ООО «Информпресс-94», 2013. – С. 35-36.
85. ASA 8.2: Packet Flow through an ASA Firewall. [Электронный ресурс] // Cisco Systems Inc. [сайт]. – Режим доступа: http://www.cisco.com/c/en/us/support/docs/security/asa-5500-x-series-next-generation-firewalls/113396-asa-packet-flow-00.html (дата обращения 15.03.2018).
86. ASA Order of operation [Электронный ресурс] // Cisco Systems Inc. [сайт]. – Режим доступа: https://learningnetwork.cisco.com/servlet/JiveServlet/showImage/2-145132-40964/ASAOperation031408.jpg (дата обращения 15.03.2018).
87. ASA Order of operation in Routing Mode [Электронный ресурс] // Cisco Systems Inc. [сайт]. – Режим доступа: http://deepakarora1984.blogspot.ru/2009/05/asa-order-of-operation.html (дата обращения 15.03.2018).
88. Cisco ASA New Features by Release [Электронный ресурс] // Cisco Systems Inc. [сайт]. – Режим доступа:

http://www.cisco.com/c/en/us/td/docs/security/asa/roadmap/asa_new_features.html (дата обращения 15.03.2018).

89. Santos, O. Cisco Next-Generation Security Solutions: All-in-one Cisco ASA Firepower Services. NGIPS and AMP (Networking Technology: Security) [Text] / O. Santos, P. Kampanakis, A. Woland. // Cisco Press, 2016. – 368 p.
90. Cisco ASA FirePOWER Module Quick Start Guide [Электронный ресурс] // Cisco Systems Inc. [сайт]. – Режим доступа: http://www.cisco.com/c/en/us/td/docs/security/asa/quick_start/sfr/firepower-qsg.html (дата обращения 15.03.2018).
91. QoS на примерах конфигурации Cisco ASA [Электронный ресурс] // Cisco Systems Inc. [сайт]. – Режим доступа: http://www.cisco.com/c/ru_ru/support/docs/security/asa-5500-x-series-next-generation-firewalls/82310-qos-voip-vpn.html (дата обращения 15.03.2018).
92. JNCIS-SEC Study Guide – Part 1 [Text] // Juniper Networks. Inc., 2012. – 211 p.
93. JNCIS-SEC Study Guide – Part 2 [Text] // Juniper Networks. Inc., 2012. – 62 p.
94. JunOS ALG Basics for Security Devices // Juniper Networks. Inc., 2012. – 38 p.
95. HUAWEI WSG6000&USG9500 Administrator Guide (V500R001C30) // Huawei Technologies Co., ltd., 2016. – 6480 p.
96. HUAWEI USG 6000 Series Next-Generation Firewall Technical White Paper [Text] // Huawei Technologies Co., ltd, 2014. – 59 p.
97. Sameer Seth, M. TCP/IP Architecture. Design and Implementation in Linux [Text] / M. Sameer Seth, V. Ajaykumar. – Wiley-IEEE Computer Society Pr, 2009. – 772p.
98. Bovet, D. Understanding the Linux Kernel, 3rd edition [Text] / D. Bovet, M. Cesati. – O'Reily, 2005. – 944 p.
99. Herbert, T.F. The Linux TCP/IP Stack: Networking for Embedded Systems [Text] T.F. Herbert. – Charles River Media, 2004. – 600 p.
100. Advanced Routing и QoS [Электронный ресурс] // OpenNET [сайт]. – Режим доступа: http://www.opennet.ru/docs/RUS/adv_route_qos/ (дата обращения 15.03.2018).

101. Networking Concepts HOWTO [Электронный ресурс] // netfilter [сайт]. – Режим доступа: http://www.netfilter.org/documentation/HOWTO//networking-concepts-HOWTO.a4.ps (дата обращения 15.03.2018).
102. Packet Filtering HOWTO [Электронный ресурс] // netfilter [сайт]. – Режим доступа: https://netfilter.org/documentation/HOWTO/pl/packet-filtering-HOWTO.txt (дата обращения 15.03.2018).
103. NAT HOWTO [Электронный ресурс] // netfilter [сайт]. – Режим доступа: http://www.netfilter.org/documentation/HOWTO//NAT-HOWTO.a4.ps (дата обращения 02.02.2014).
104. Netfilter Extensions HOWTO [Электронный ресурс] // netfilter [сайт]. – Режим доступа: https://netfilter.org/documentation/HOWTO//NAT-HOWTO.txt (дата обращения 15.03.2018).
105. Netfilter Hacking HOWTO [Электронный ресурс] // netfilter [сайт]. – Режим доступа: https://netfilter.org/documentation/HOWTO//netfilter-hacking-HOWTO.txt (дата обращения 15.03.2018).
106. Анатомия сетевого стека в Linux [Электронный ресурс] // IBM [сайт]. – Режим доступа: https://www.ibm.com/developerworks/ru/library/l-linux-networking-stack/ (дата обращения 15.03.2018).
107. A One-way Delay Metric for IPPM [Электронный ресурс] / RFC 2679 // IETF [сайт]. – 1999. – Режим доступа: http://www.ietf.org/rfc/rfc2679.txt (дата обращения 15.03.2018).
108. A Round-trip Delay Metric for IPPM [Электронный ресурс] / RFC 2681 // IETF [сайт]. – 1999. – Режим доступа: http://www.ietf.org/rfc/rfc2681.txt (дата обращения 15.03.2018).
109. IP Packet Delay Variation Metric for IP Performance Metrics (IPPM) [Электронный ресурс] / RFC 3393 // IETF [сайт]. – 2002. – Режим доступа: http://www.ietf.org/rfc/rfc3393.txt (дата обращения 15.03.2018).
110. Назаров, А.Н. Модели и метод расчёта показателей качества функционирования узлового оборудования структурно-сетевых параметров

сетей связи следующего поколения [Текст] / А.Н. Назаров, К.И. Сычёв. – Изд-во ООО «Поликом»., 2011. – 491 с.

111. Столлингс, В. Современные компьютерные сети [Текст] / В. Столлингс. – СПб.: Питер, 2003. – 783с.
112. Аджемов, А.С. Общая теория связи [Текст] / А.С. Аджемов, В.Г. Санников. – М.: Горячая линия – Телеком, 2018. – 624 с.
113. Башарин, Г.П. Теория сетей массового обслуживания и её приложения к анализу информационно-вычислительных систем [Текст] / Г.П. Башарин, А.А. Толмачев // Итоги наука и техника. Теория вероятностей. Математическая статистика. Теоретическая кибернетика. – 1983. – С.3-119.
114. Жожикашвили, В.А. Сети массового обслуживания. Теория и применение к сетям ЭВМ [Текст] / В.А. Жожикашвили, В.Н. Вишневский. – М.: Радио и связь, 1988. – 192 с.
115. Ивницкий, В.А. Теория сетей массового обслуживания [Текст] / В.А. Ивницкий. – М.: Изд-во физико-математической литературы, 2005. – 772 с.
116. Наумов, В.А. Мультипликативные решения конечных цепей Маркова: монография [Текст] / В.А. Наумов, К.Е. Самуйлов, Ю.В. Гайдамака – М: РУДН, 2015. – 159 с.
117. Мусатов, В.К. Математическое моделирование средств межсетевого экранирования в условиях приоритизации трафика [Текст] / В.К. Мусатов, А.А. Щербанская // T-Comm: Телекоммуникации и транспорт. – 2015. – Том 9, №8. – 2015. – С.47-57.
118. Обзор продукта Wolfram Mathematica [Электронный ресурс] // Wolfram [сайт]. – Режим доступа: http://www.wolfram.com/mathematica/?source=nav (дата обращения 15.03.2018).
119. Акопов, А.С. Имитационное моделирование: учебник и практикум для академического бакалавриата [Текст] / А.С. Акопов. – М.: Изд-во Юрайт, 2016. – 389 с.
120. Вьюненко, Л.Ф. Имитационное моделирование: учебник и практикум для академического бакалавриата [Текст] / Л.Ф. Вьюненко, М.В. Михайлов,

Т.Н. Первозванская; под ред. Л. Ф. Вьюненко. – М.: Изд-во Юрайт, 2018. – 283 с.

121. What Is Managed Code? [Электронный ресурс] // Microsoft MSDN [сайт]. – Режим доступа: https://msdn.microsoft.com/ru-ru/library/windows/desktop/bb318664(v=vs.85).aspx (дата обращения 15.03.2018).
122. Garbage Collection // msdn.microsoft.com [Электронный ресурс] // Microsoft MSDN [сайт]. – Режим доступа: https://msdn.microsoft.com/ru-ru/library/0xy59wtx(v=vs.110).aspx (дата обращения 15.03.2018).
123. Бусленко, Н.П. Моделирование сложных систем [Текст] / Н.П. Бусленко – М.: Изд-во «НАУКА», 1968. – 355 с.
124. Гамма, Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования [Текст] / Э. Гамма, Р. Халм, Р. Джонсон, Д. Влиссидес. – Изд-во «Питер», 2016. – 366 с.
125. Welcome to NLog [Электронный ресурс] // NLog [сайт]. – Режим доступа: http://nlog-project.org/ (дата обращения 15.03.2018).
126. Math.Net Numerics [Электронный ресурс] // Math.NET Numerics [сайт]. – Режим доступа: https://numerics.mathdotnet.com/ (дата обращения 15.03.2018).
127. Bool (справочник по C#) [Электронный ресурс] // Microsoft MSDN [сайт]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/bool (дата обращения 15.03.2018).
128. Integer (справочник по C#) [Электронный ресурс] // Microsoft MSDN [сайт]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/int (дата обращения 15.03.2018).
129. ENUM (Справочник по C#) [Электронный ресурс] // Microsoft MSDN [сайт]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/enum (дата обращения 15.03.2018).

130. Double (справочник по C#) [Электронный ресурс] // Microsoft MSDN [сайт]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/double (дата обращения 15.03.2018).
131. Long (справочник по C#) [Электронный ресурс] // Microsoft MSDN [сайт]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/long (дата обращения 15.03.2018).
132. String (справочник по C#) [Электронный ресурс] // Microsoft MSDN [сайт]. – Режим доступа: https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/string (дата обращения 15.03.2018).
133. Matsumoto, M. Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator [Text] / M. Matsumoto, T. Nishimura // ACM Transactions on Modeling and Computer Simulation (TOMACS). – 1998. – Vol. 8, № 1. – pp. 3-30.
134. Random – класс [Электронный ресурс] // Microsoft MSDN [сайт]. – Режим доступа: https://msdn.microsoft.com/ru-ru/library/system.random(v=vs.110).aspx (дата обращения 15.03.2018).
135. Knuth, D.E. The Art of Computer Programming. Volume 2: Seminumerical Algorithms. Third edition [Text] / D.E. Knuth. – Addison-Wesley. Reading. MA, 1997. – 762p.
136. Seemann, M. Dependency Injection in .NET [Text] / M. Seemann. – WILEY, 2011. – 584 p.
137. Мусатов, В.К. Имитационное моделирование средств межсетевого экранирования в условиях приоритизации трафика [Текст] / В.К. Мусатов, А.П. Пшеничников, А.А. Щербанская // T-Comm: Телекоммуникации и транспорт. – 2016. – Том 10, №12. – С. 10-17.
138. Уэйкерли, Д.Ф. Проектирование цифровых устройств. Том I. [Текст] / Д.Ф. Уэйкерли. – Постмаркет, 2002. – 544с.
139. Smith, M. Application-Specific Integrated Circuits 1st Edition [Text] / M. Smith. – Addison-Wesley Professionally, 1997. – 1040 с.

140. Giladi, R. Network Processors: Architecture. Programming. and Implementation (Systems on Silicon) 1st Edition [Text] / R. Giladi.– Morgan Kaufmann, 2008. – 736 p.
141. Клейнрок, Л. Вычислительные системы с очередями. Пер. с англ. под ред. Б.С. Цыбакова [Текст] / Л. Клейнрок. – М.: Мир, 1979. – 600 с.
142. Мусатов, В.К. Моделирование влияния приоритизации трафика на входном интерфейсе межсетевого экрана на его показатели производительности [Текст] / В.К. Мусатов, А.П. Пшеничников. // 12-ая международной научно-технической конференции «Перспективные технологии в средствах передачи информации»: матер. конф. том II – Владимир: Изд-во ВлГУ, 2017. – С.84-86.
143. Мусатов, В.К. Влияние применения приоритизации обслуживания пакетов на входе межсетевого экрана на показатели производительности [Текст] / В.К. Мусатов // Телекоммуникационные и вычислительные системы: труды конференции – М.: Горячая линия - Телеком, 2017. – С. 51-55.

Список иллюстративного материала

Номер страницы **Наименование рисунка, таблицы**

Введение

Раздел 1. Анализ методов повышения производительности межсетевых экранов

Раздел 2. Анализ принципов обработки пакетов в межсетевых экранах

Номер страницы **Наименование рисунка, таблицы**

Раздел 3. Разработка математической модели функционирования межсетевого экрана в условиях приоритизации обслуживания трафика

Раздел 4. Разработка Имитационной модели функционирования межсетевого экрана в условиях приоритизации обслуживания трафика

Номер страницы **Наименование рисунка, таблицы**

Приложение А.

Полное описание схем обслуживания рассматриваемых межсетевых экранов

Компания Cisco Systems. В работе рассматривалось МЭ линейки Cisco ASA 5500 серии предыдущего поколения (т.е. не Cisco ASA FirePower). Схема обработки пакетов представлена на рисунке А.1.

Схема сформирована для старого поколения Cisco ASA, ввиду наличия большего количества информации о принципах функционирования. В схеме обслуживания не рассматриваются процессы построения виртуальных частных сетей, шифрования и кодирования трафика.

Cisco ASA with Firepower от ASA предыдущего поколения отличается изменённой аппаратной архитектурой и составом модулей поддержки, рядом новых функций и некоторыми изменениями процесса обслуживания. В ASA нового поколения доступен модуль расширения функционала – FirePOWER (сервисная плата). Такой модуль добавляется в штатное шасси ASA и предоставляет дополнительную прослойку функций безопасности, таких как. – Режим доступа-фильтрацию, продвинутую защиту от вредоносного ПО, систему предотвращения вторжений [79]. В ASA предыдущего поколения часть функционала модуля FirePOWER предоставляли CSC и IPS модули, которые также монтировались в штатное шасси ASA. Модуль FirePOWER один и он находится в очереди обслуживания на позиции IPS-модуля ASA предыдущего поколения. Функционал фильтрации, предоставляемый модулем CSC, выполняется в качестве базовая функция ASA нового поколения.

Также уточним, что изменилась очерёдность операций проверки правил трансляции сетевых адресов (NAT/PAT) и фильтрации по ACL в первичном пути проверки пакета. Эти операции являются соседними.

Необходимо учитывать, что механизмы качества обслуживания в Cisco ASA реализуются только в *режиме маршрутизатора* (router mode) и не функционируют в *прозрачном режиме* (transparent mode). В остальном принципы обслуживания в режиме маршрутизатора и прозрачном режиме почти не

отличаются. Сама Cisco ASA позиционируется чаще как шлюз доступа (пограничный межсетевой экран), работая в режиме маршрутизатора.

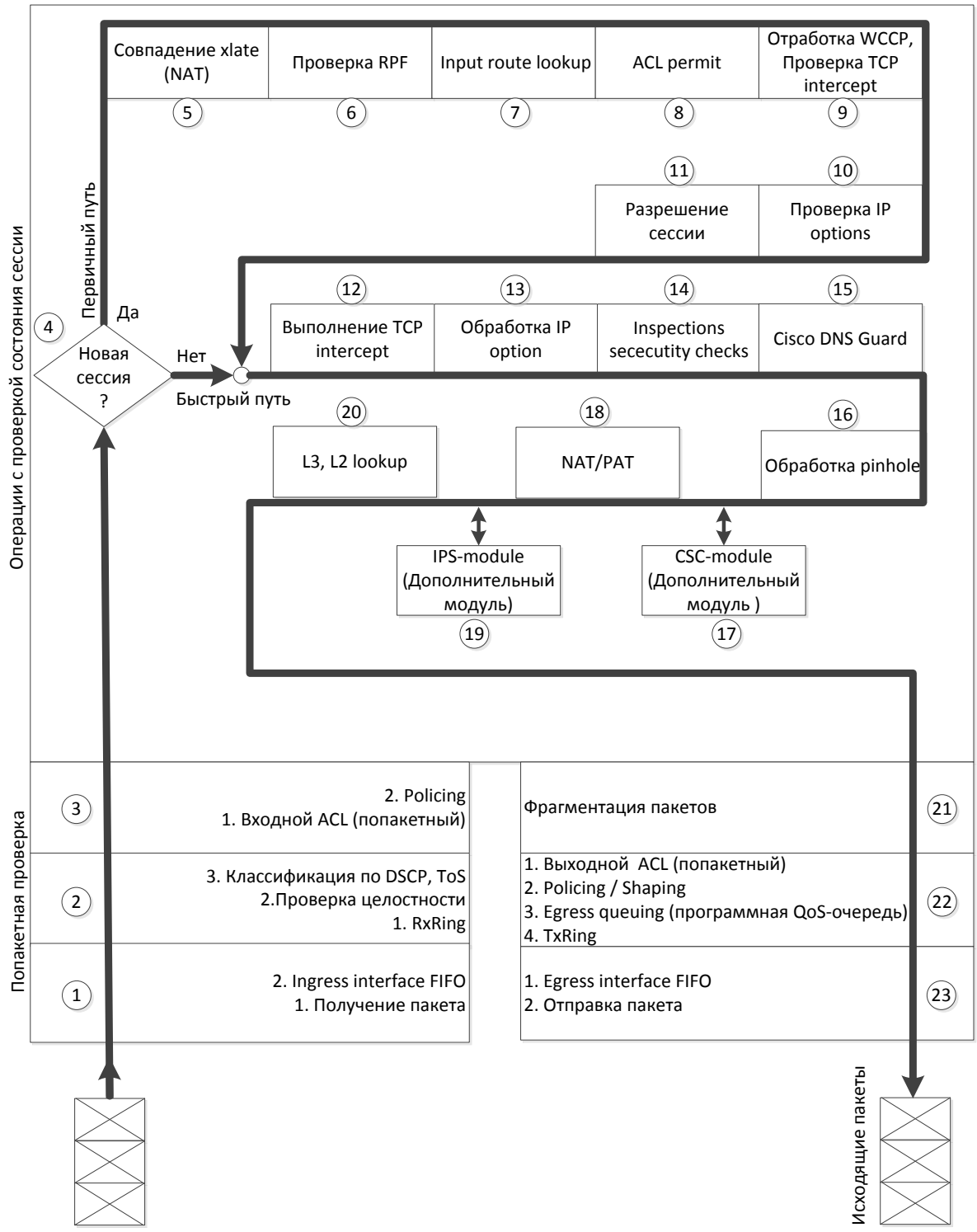


Рисунок А.1 – Схема обработки пакетов Cisco ASA предыдущего поколения

Условно разделим процесс обработки пакетов на два этапа. Первый этап – попакетная проверка, в течение которого механизмы МЭ проверяют каждый пакет полностью. Второй этап – операции с проверкой

состояния сессии, в течение которых механизмы МЭ проверяют пакеты с учётом состояния соединения, в рамках которого они передаются. Такое разделение будет характерно и для следующих двух рассмотренных МЭ.

Пошаговое описание схемы обслуживания (рисунок А.1):

1. Пакет приходит по физическому кабелю и попадает в буфер сетевой карты. Пакет может быть сброшен при переполнении памяти сетевой карты. Алгоритм сброса не учитывает приоритет пакета.
2. Используется прямой доступ к памяти (direct memory access, DMA) для переноса пакета в кольцевую структуру данных RxRing, находящуюся в общей памяти. Перенос осуществляется в порядке поступления (FIFO). Затем производит распаковку и проверку целостности по контрольным суммам. Далее производится сборка фрагментированного трафика, а также классификация по DSCP, ToS, 802.1q.
3. Производится попакетная проверка по ACL. Пакет может быть сброшен или промаркирован (классифицирован) по результатам работы ACL. Затем применяется ограничение полосы пропускания, по результатам выполнения операции пакет может быть сброшен или промаркирован. Если пакет был классифицирован как приоритетный, то он минует стадию ограничения потока трафик и сразу направляется на обслуживание.
4. Выполняется поиск по таблице разрешённых соединений, если соединение новое, то идём к шагу 5, этот путь называется «first path», если соединение зафиксировано в таблице как разрешённое, то идём к шагу 12, этот путь называется «fast path».
5. Производится проверка правил трансляции сетевых адресов. При неудовлетворении правил пакет может быть сброшен.
6. Производится проверка Reverse Path Forwarding на передачу широковещательного трафика без петель. При обнаружении петли пакет сбрасывается.
7. Производится проверка наличия дальнейшего маршрута (input route lookup). При отсутствии маршрута пакет будет сброшен.

8. Производится проверка трафика по L3-L4 ACL. Пакет может быть сброшен по результатам работы ACL.
9. Выполняется переадресация Web Cache Communication Protocol (выполняется опционально). Выполняется TCP Intercept (механизмы защиты от SYN-flood). Пакет может быть сброшен по результатам выполнения операции.
10. Производится проверка дополнительных параметров IP-пакета (IP options; пакет может быть сброшен).
11. Производится запись о сессии в таблицу разрешённых соединений.
12. Вторично выполняется TCP Intercept (в частности механизм отвечает за проверку TCP sequence).
13. Выполняются операции, связанные с дополнительными параметрами IP-пакета (IP options).
14. Производится проверка приложений и протоколов, реализует базовый функционал шлюза прикладного уровня (Application Layer Gateway, ALG). Функционал ALG расширяется CSC-модулем или модулем FirePOWER в ASA следующего поколения. Пакеты могут быть сброшены по результатам работы механизма.
15. Производится проверка пакетов на наличие атак с использованием уязвимостей в DNS – Cisco DNS Guard. Пакеты могут быть сброшены по результатам работы механизма, проверке подлежат только пакеты протокола DNS.
16. Выполняется операции pinhole (позволяет пропускать трафик без дальнейших проверок безопасности).
17. Производится проверка трафика CSC-модулем (выполняется при наличии соответствующего модуля).
18. Выполняется трансляция сетевых адресов (NAT/PAT).
19. Производится проверка трафика модулем предотвращения вторжений (Intrusion prevention system, IPS; выполняется при наличии соответствующего модуля).

20. Выполняется поиск маршрута сначала на сетевом, а затем на канальном уровне. На этом этапе пакет может быть сброшен, если маршрут не найден, нет маршрута по умолчанию, и заранее не выполнялся пункт 7, который обнаружил бы отсутствие маршрута.

21. Фрагментация пакетов, если это требуется. После фрагментации пакет переносится в выходную очередь в общей памяти (на схеме не указано).

22. Производится попакетная проверка по ACL, затем ограничение и/или выравнивание полосы пропускания, *постановка в программную очередь на отправку* (egress queuing). Программная очередь поддерживает приоритетное обслуживание. Пакет может быть сброшен или промаркирован по результатам выполнения ACL или при выполнении ограничения и выравнивания полосы пропускания. Выходная очередь может быть с приоритетом обслуживания или без него. Пакет может быть сброшен из очереди ввиду её переполнения. Операция сброс пакетов из очереди (active queue management) может выполняться с учётом приоритета или без учёта приоритета, алгоритмы сброса tail drop, RED, WRED. Пакет переносится в TxRing (всё ещё в основной памяти).

23. Использованием DMA драйвер переносит пакет из TxRing в аппаратную очередь сетевой карты. Отправка из очереди осуществляется в соответствии с дисциплиной FIFO.

Общая концепция обслуживания пакетов и архитектура МЭ при работе в режиме маршрутизатора совпадают с той, что рассматривалась в пункте 2.2.3. С точки зрения рассматриваемых аспектов функционирования МЭ в прозрачном режиме отличия от общей архитектуры в основном заключаются в отсутствии поддержки функции качества обслуживания.

Операции 1 и 23 выполняются на сетевых картах (в том числе встроенных) или интерфейсных платах. Операции 2-16, 18, 20-22 выполняются ресурсами МЭ. Операции 17 и 19 выполняются на специальных платах расширения функционала CSC-модуле и IPS-модуле.

Платы расширения функционала получают копию пакета из общей памяти МЭ. Они не сбрасывают трафик самостоятельно, а только помечают его к сбросу. Помеченный трафик будет сброшен МЭ самостоятельно.

Пакет, попавший на первичный путь, проходит ряд проверок, которым не подвергаются пакеты в быстром пути. После прохождения первичного пути, пакеты попадут в быстрый путь и будут там дополнительно проверены. Проверки в быстром пути частично повторяют функционал, заложенный в проверках первичного пути, а также содержат дополнительные проверки.

Приоритет обслуживания трафика может быть учтён в группах операций 3 и 22. Приоритет обслуживания в группе 3 учитывается при выполнении ограничения полосы трафика, точнее, если пакет приоритетный, то он пропускает эту операцию. В старших моделях ASA можно выделять приоритетным пакетам отдельную полосу пропускания. В группе операций 22 приоритизация обслуживания учитывается в операциях выравнивание потока трафика и отправки пакетов в сеть из очереди (egress queuing), а также при сбросе пакетов при переполнении очереди. Также если в группе операций 22 используется не выравнивание, а ограничение трафика, то приоритет может учитываться аналогично операции 3.

Выходные программные очереди (операция 22) могут работать в соответствии с дисциплинами FIFO и *очередь с малой задержкой* (low latency queue; LLQ). Дисциплины сброса пакетов из очереди реализованы алгоритмами tail drop, *Weighted random early detection* (WRED).

Условно можно считать, что операции вносят задержку, составляющую постоянную или динамическую величину. Постоянную задержку вносят операции, реализованные аппаратным способом, например, с использованием ASIC. В остальных случаях принятие факта того, что операция вносит постоянную задержку, стоит считать допущением (упрощением модели).

Вносящими динамические задержки стоит считать операции 8, 14, 17, 19, 22, выполняемые с низкой степенью аппаратной поддержки или имеющие большой разброс по возможной внутренней задержке, что объясняется случайным

характером соответствия пакета какому-либо правилу. При формировании допущений задержки, вносимые другими операциями, можно принять за постоянные величины. При этом операции с динамической задержкой при малом количестве правил вполне можно считать вносящими постоянные задержки.

Компания Juniper Networks. В работе рассматривалось МЭ линейки Juniper SRX. Схема обработки пакетов представлена на рисунке А.2.

Juniper SRX может функционировать в двух режимах фильтрации: на *основе сессий* (session based) и на *основе пакетов* (packet based). На основе сессий доступно два режима работы в *прозрачном режиме* (transparent mode) и более классическом режиме, условно назовём его режимом шлюза доступа (в документации производителя режим никак не назван). В пакетном режиме становится доступен *режим маршрутизатора* (router mode). Режим маршрутизатора у Cisco и Juniper полностью различаются по функционалу. Режим маршрутизатора превращает SRX в маршрутизатор, а у Cisco, режим маршрутизатора подразумевает роль и функциональность шлюза доступа. В пакетном режиме SRX теряет возможность фильтрации трафика с учётом состояния соединения, далее этот режим рассматриваться не будет.

Набор функций при обслуживании пакетов в режиме шлюза и прозрачном режиме почти не различается. Различия касаются того, что в прозрачном режиме существуют некоторые ограничения по функционалу, например, не доступна трансляция сетевых адресов. Различий по функционалу не так много и они описаны в [19]. Отсутствующий функционал представляет собой ряд аппаратно поддерживаемых операций, и с точки зрения будущей модели их можно считать постоянными задержками.

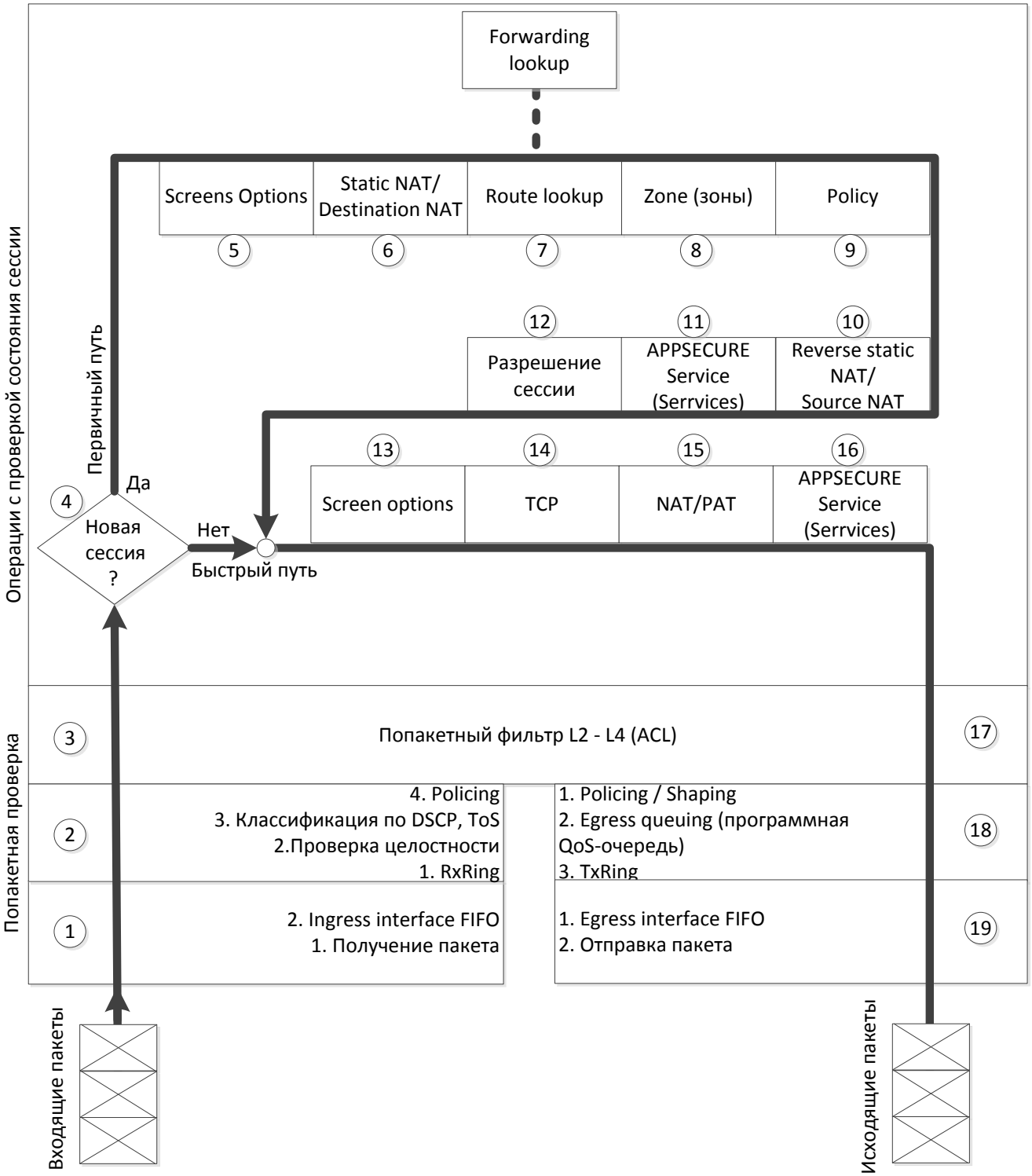


Рисунок А.2 – Схема задержки при обработке пакетов Juniper SRX предыдущего поколения

Пошаговое описание схемы обслуживания (рисунок А.2):

1. Пакет приходит по физическому кабелю и попадает в буфер сетевой карты. Пакет может быть сброшен при переполнении памяти сетевой карты. Алгоритм сброса не учитывает приоритет пакета.

2. Используется прямой доступ к памяти (DMA) для переноса пакета в кольцевую структуру данных RxRing, находящуюся в общей памяти. Перенос осуществляется в порядке поступления (FIFO), выполняет распаковку и проверку целостности. Как правило, пакеты забираются из RxRing сразу на обслуживание, в некоторых случаях они могут помещаться отдельные структуры памяти. Выполняется ограничение полосы пропускания, по результатам работы которого пакет может быть сброшен или промаркирован.

3. Выполняется попакетная фильтрация трафика L2-L4. Пакеты могут быть сброшены по результатам фильтрации.

4. Выполняется поиск по таблице существующих соединений, если соединение разрешено, то идём к шагу 13, этот путь называется «fast path», если соединение не разрешено, то идём к шагу 5, этот путь называется «first path».

5. Производится проверка пакета комплексом операций screen options в first path. Screen options является системой обнаружения атак, именуется разработчиками, как attack detection system. При обнаружении атак screen options может сбросить пакеты, а на этапе 14 заблокировать сессию.

6. Производится проверка правил трансляции сетевых адресов назначения. Сначала выполняется статическая трансляция адреса назначения, если она не требуется, то выполняется динамическая трансляция. При неудовлетворении правил пакет может быть сброшен.

7. Выполняется поиск маршрута (route lookup), если маршрута нет, то пакет сбрасывается, если маршрут есть, то выполняется прямой поиск маршрута (forward lookup), определяются интерфейсы/зоны источника и назначения.

8. Производится определение интерфейса/зоны, из которой пришёл пакет, и в которую он направляется. В соответствии с определённым направлением движения трафика применяется «грубая» политика фильтрация между зонами, затем пакет передаётся на этап 9 для более детальной проверки. Пакет может быть сброшен в процессе проверок безопасности.

9. Производится детальная проверка пакетам на соответствие политикам фильтрации. При выборе политики фильтрации учитывается направление движения трафика, определённое на этапе 8. Такой подход позволяет специализировать фильтры под конкретные задачи, разделить одну большую базу правил на несколько меньших, тем самым ускорив обслуживание. Пакет может быть сброшен в процессе фильтрации.

10. Проверка правил трансляции сетевых адресов источника. Сначала выполняется обратная *статическая трансляция адреса источника* (reverse static NAT), если она не требуется, то выполняется динамическая трансляция.

11. Производится проверки пакета модулем APPSECURE Service. В рамках модуля работают такие сервисы как ALG, App FW, AppTrack, AppQoS, AppDos, IDP (IPS) и др. В результате работы модуля пакет может быть сброшен, также может быть заблокирована вся сессия (сеанс) связи.

12. Производится запись о сессии в таблицу разрешённых соединений.

13. Производится проверка пакета комплексом операций screen options в fast path. При обнаружении атак screen options может сбросить пакеты или заблокировать сессию.

14. Производится проверки состояния TCP-сессии, в частности TCP sequence.

15. Выполнение трансляции сетевых адресов NAT/PAT.

16. Производится проверки пакета модулем APPSECURE Service. В результате работы сервисов пакет может быть сброшен, также может быть заблокирована вся сессия (сеанс) связи.

17. Производится попакетная фильтрация трафика. Пакеты могут быть сброшены по результатам фильтрации.

18. Выполняется ограничение и/или выравнивание полосы пропускания, постановка в выходную программную очередь на отправку. Программная очередь поддерживает приоритетное обслуживание. Пакет может быть сброшен по результатам выполнения ACL, сброшен или перемаркирован

при выполнении ограничения полосы пропускания. Затем пакет попадает во входную очередь, которая может быть с приоритетом обслуживания или без него, пакет может быть сброшен из очереди ввиду её переполнения. Операция сброс пакетов из очереди (active queue management) может выполняться с учётом приоритета или без учёта приоритета. Пакет попадает в TxRing (всё ещё в основной памяти).

19. С использованием DMA пакет из TxRing переносится в аппаратную очередь сетевой карты. Отправка из очереди осуществляется в соответствии с дисциплиной FIFO.

Общая концепция обслуживания пакетов и рассматриваемые элементы архитектуры SRX в целом совпадают представленной в пунктах 2.1.3-2.1.5 информацией.

Операции 1 и 19 выполняются на сетевых картах (в том числе встроенных) или интерфейсных платах. Операции 2-18 выполняются ресурсами МЭ.

Пакет, попавший на первичный путь, проходит ряд проверок, которым не подергаются пакеты в быстром пути. После прохождения первичного пути пакеты попадут в быстрый путь и будут там дополнительно проверены. Проверки в быстром пути частично повторяют функционал, заложенный в проверках первичного пути, а также содержат дополнительные проверки.

Приоритет обслуживания трафика может быть учтён в группах операций 2 и 18. В группе операций 2 приоритет может быть учтён при выполнении операции ограничения полосы пропускания. Зачастую приоритет не учитывают, если не доверяют меткам качества обслуживания. В группе операций 18 приоритет учитывается в операциях выравнивания полосы пропускания и отправки пакетов в сеть из очереди, а также при сбросе пакетов при переполнении очереди. Если в группе операций 22 используется не выравнивание, а ограничение трафика, то приоритет может учитываться аналогично операции 2.

Дисциплины обслуживания пакетов в выходной QoS-очереди другие. Вместо LLQ поддерживаются две дисциплины: stick-priority queue и low-low priority queue. По-прежнему доступна дисциплина FIFO. Дисциплина

strict-priority queue аналогична по своему принципу работы с дисциплиной LLQ. Не удалось подтвердить алгоритм сброса пакетов при переполнении аппаратной очереди. Некоторые источники утверждают, что используется алгоритм RED, однако, размер аппаратной очереди обычно так мал, что применение алгоритма RED нецелесообразно. Для программных очередей используются алгоритмы tail drop и RED.

Вносящими динамические задержки стоит считать операции 3, 5, 9, 11, 13, 16, 17, 18. Можно допустить, что остальные операции вносят постоянную задержку, а также что при малом количестве правил в операциях с динамической задержкой можно считать вносимую задержку постоянной величиной.

Компания Huawei Technologies. В работе рассматривалось МЭ линейки Huawei USG 6000 серии, кратко затрагивается особенность обработки пакетов в USG 9000 серии. Схема обработки пакетов представлена на рисунке А.3.

Серия межсетевых экранов Huawei USG6000 не поддерживает ряд механизмов качества обслуживания, а именно, выравнивание полосы пропускания и приоритетное обслуживание, эти механизмы доступны только в старшей линейке USG 9500. Механизм, аналогичный по функционалу ограничению полосы пропускания, относится к группе «bandwidth policy», однако, эта групп имеет целый ряд дополнительных механизмов управления полосой пропускания. В линейке USG 9500 помимо прочего доступно формирование приоритетных очередей на входе МЭ.

Пошаговое описание схемы обслуживания.

1. Пакет приходит по физическому кабелю и попадает в аппаратную очередь (буфер памяти) сетевой картыю. Пакет может быть сброшен при переполнении очереди. Алгоритм сброса не учитывает приоритет пакета.
2. Используется DMA для переноса пакета из буфера сетевой карты в RxRing в порядке поступления (FIFO). RxRing находится в основной памяти устройства. В дальнейшем пакет будет забран на обслуживание и при необходимости перенесён в другие структуры выделенные области основной памяти. Выполняется фильтрация по L2, L3, которая может быть

настроена вручную или с использованием функции IP/MAC binging. Пакет может быть сброшен или промаркирован по результатам выполнения операций.

3. Выполняется группа операций, управляющих полосой пропускания. Проверяется необходимость ограничения полосы пропускания, после чего пакет сбрасывает или продолжает обслуживание. Наличие механизмов очередей и приоритизации обслуживания на входе в USG 6000 в отличие от USG9000 не предусмотрено. Пакет может быть сброшен при выполнении операций.

4. Производится проверка пакетов на обнаружение DDoS-атаки, по результатам работы которой, пакет может быть сброшен.

5. Производится проверка пакетов механизмов single-packet attack defense. Пакет может быть сброшен по результатам выполнения операций.

6. Выполняется поиск по таблице существующих соединений, если соединение разрешено, то идём к шагу 19, этот путь называется «fast path», если соединение не разрешено, то идём к шагу 7, этот путь называется «first path».

7. Производится проверка механизмом «Stateful inspection mechanism», которая определяет, что только первый TCP или ICMP пакет может открыть сессию (создать запись в таблице разрешённых сессий).

8. Производится фильтрация по чёрному списку (blacklist), чья работа основана на IP-адресах и информации о пользователе, содержащейся в пакете. При выполнении операций пакет может быть сброшен.

9. Производится проверка на соответствие механизму «Server-map», который помогает обеспечить работу протоколов с выделенными каналами управления, такими как FTP, из внешней сети.

10. Выполняется запись в таблицу активных пользователей (online user list), содержащую сочетание информации: пользователь, IP-адрес, время открытие сессии, время активности.

11. Выполняется запись информации о приложении, выславшем пакет, для её ассоциации с конкретной сессией.
12. Выполняется поиск маршрута. Пакет может быть потерян, если не удалось найти маршрут, и нет маршрута по умолчанию.
13. Выполняется выбор способа аутентификации для потока и получение пользовательской информации на основе IP-адреса и зоны безопасности.
14. Выполняется переадресация пользователя на страницу аутентификации (если это требуется). При вводе неверных учётных данных пакет может быть сброшен. Межсетевой экран будет выполнять фильтрацию других пакетов, пока пользователь будет аутентифицироваться на нём. После успешной аутентификации продолжится обработку.
15. Производится применение политик безопасности (фильтрация трафика). Этот этап необходимо разделить на два: проверку по состоянию и по профилю. Фильтрация по состоянию выполняется по служебной информации пакета, а также по данным полученных при выполнении предыдущих операций. Первое сработавшее правило применяет действие к пакету, будь то пропуск или сброс пакета. Если пакет пропущен, то его можно направить на проверку по профилю. Проверка по профилю производится на основании полезной нагрузки пакета. Профили работают по принципу все, что не запрещено разрешено. На этапе проверки по профилю пакет также может быть сброшен. Если проверка по всем профилям успешна, то пакет продвигается дальше.
16. Выполняется проверка правил трансляции сетевых адресов источника. Пакет может быть сброшен при выполнении операции.
17. Выполняется ограничение количества соединений, основываясь на политике управления полосой пропускания (bandwidth policy). Пакет может быть сброшен при выполнении операции.
18. Выполняется запись о сессии в таблицу разрешённых соединений.
19. Обновляется информация об активности пользователя, за которым закреплена сессия. Сессия закрепляется на этапе 10.

20. Производится проверка пакетов механизмом защиты от атак на основе потоков (flow-based attack defense). Механизм анализирует множество пакетов в сессии и может фильтровать или ограничивать полосу пропускания для потока трафика в рамках этой сессии. Пакет может быть сброшен при выполнении операции.

21. Производится проверка пакетов механизмом проверки состояния сессии (stateful inspection). Пакет может быть сброшен при выполнении операции.

22. Выполняется проверка наличия изменений в параметрах сессии. Если параметры изменились, то пакет отправляется на дополнительный комплекс проверок этап 23, если нет, то он отправляется на этап 27.

23. Выполняется проверка маршрутной информации, допускается смена маршрута, если она разрешена. Операция аналогична той, что проводиться на этапе 12. Пакет может быть сброшен при выполнении операции.

24. Производится вторичная проверка пакета на соответствие политикам безопасности. Операция аналогична той, что проводится на этапе 15. Пакет может быть сброшен при выполнении операции.

25. Производится фильтрация по чёрному списку (blacklist). Операция аналогична той, что проводится на этапе 8. Пакет может быть сброшен при выполнении операции.

26. Выполняется переадресация пользователя на страницу аутентификации (если это требуется). Операция аналогична той, что выполняется на этапе 13. При вводе неверных учётных данных пакет может быть сброшен.

27. Выполняется проверка выполнения политик полосы пропускания, корректируется профиль трафика (при необходимости).

28. Производится проверка пакета на соответствие политик безопасности прикладного уровня. На этом этапе функционируют система предотвращения вторжений и потоковый антивирус. Пакет может быть сброшен при выполнении операции.

29. Выполняется трансляция сетевых адресов, на основе ранее полученной в процессе обработки пакета информации.

30. Выполняется построения VPN-туннеля, пакет шифруется (опционально).

31. Выполняется ограничение полосы пропускания. Проверяется необходимость ограничения полосы пропускания, после чего пакет сбрасывает или отправляется в сеть. Наличие механизмов очередей, формирования потока трафика и приоритизации отправки на выходном интерфейса в USG 6000 в отличие от USG9000 не предусмотрено.

32. Пакет попадает в TxRing (всё ещё в основной памяти) и затем с использованием DMA переносится в аппаратную очередь сетевой карты. Отправка из очереди осуществляется в соответствии с дисциплиной FIFO.

Общая концепция обслуживания пакетов и архитектура МЭ совпадают с той, что рассматривалась в пункте 2.2.3, однако, существует несколько нюансов.

Операции 1 и 32 выполняются на сетевых картах (в том числе встроенных) или интерфейсных платах. Операции 2-31 выполняются ресурсами МЭ.

Как и в рассмотренных ранее устройствах пакет, попавший на первичный путь, проходит ряд проверок, которым не подвергаются пакеты в быстром пути. После прохождения первичного пути пакеты попадут в быстрый путь и будут там дополнительно проверены. Проверки в быстром пути, частично повторяют функционал, заложенный в проверки первичного пути, частично состоят из дополнительных проверок. В отличие от рассмотренных ранее МЭ в быстром пути предусмотрено дополнительное ветвление алгоритма проверок. При обнаружении изменения ряда параметров сессии производятся дополнительные проверки, которым соответствуют операции 23-26.

При обслуживании приоритет пакетов в USG 6000 серии учитывается только при ограничении выделенной полосы пропускания. В USG 6000 серии присутствуют механизмы управления полосой пропускания, однако, специалисты Huawei не используют термин policer. При этом управление полосой пропускания подразумевает выделение определённым типам трафика, доступной и/или гарантированной им полосы пропускания в соответствии с их классом обслуживания. Трафик, выходящий за доступную полосу, сбрасывается. Классификация трафика выполняется на попакетном фильтре.

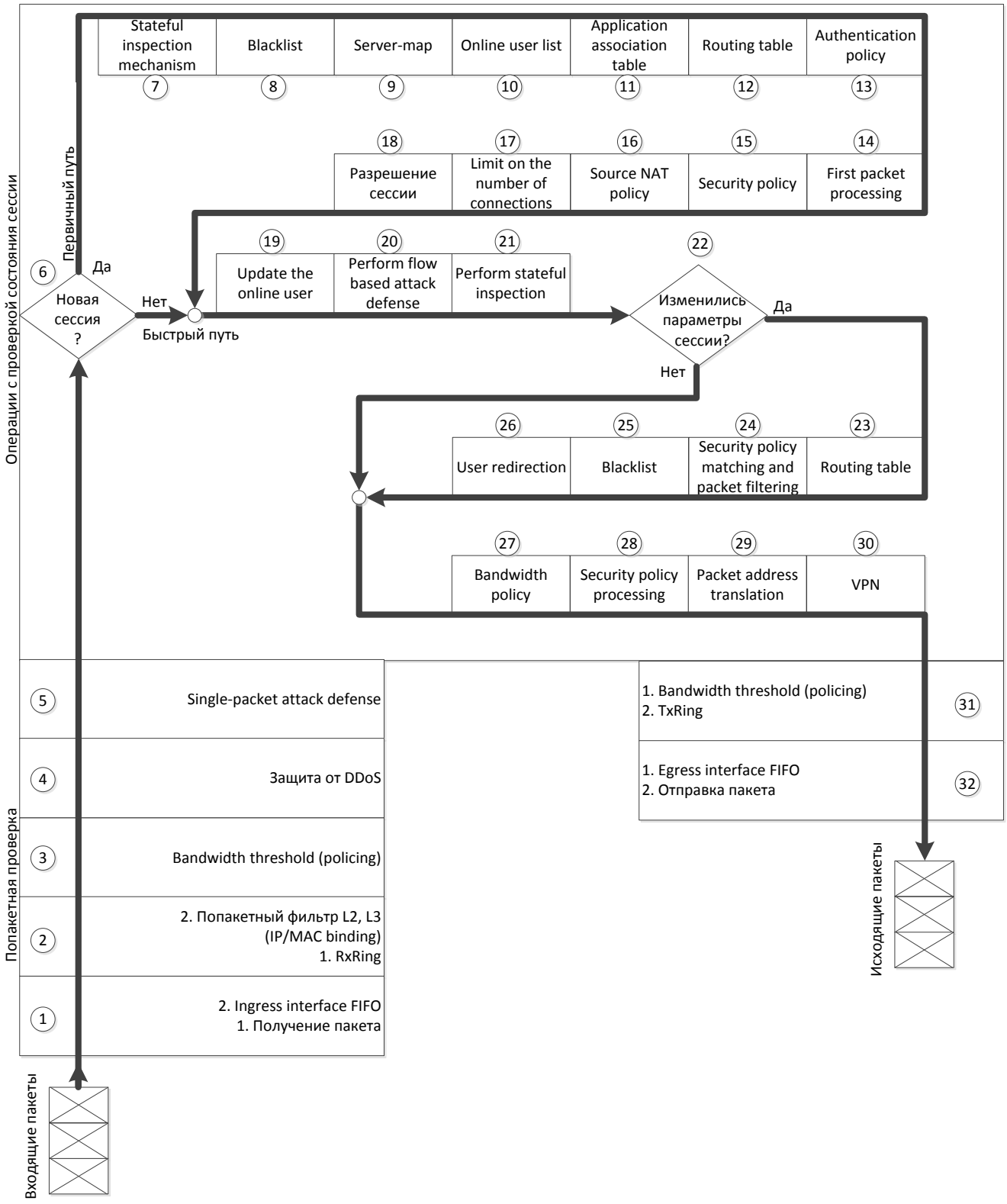


Рисунок А.3 – Схема задержке при обработке пакетов Huawei USG 6000

Документация не раскрывает подробностей работы физических очередей сетевых карт, вероятно, стоит считать принятой дисциплиной обслуживания FIFO, а принятым алгоритмом сброса пакетов tail drop.

Вносящими динамические задержки стоит считать операции 4, 5, 8, 13, 15, 20, 21, 24, 25, 26, 28. Можно допустить, что остальные операции вносят постоянную задержку, а также что при малом количестве правил в операциях с динамической задержкой можно считать вносимую задержку постоянной величиной.

Производители свободно распространяемого программного обеспечения. В работе рассматривалось МЭ, представленный комплексом ОС Linux, программного обеспечения NetFilter/IPTables, iproute2, tc. Схема обработки пакетов представлена на рисунке 2.9. Общее описание и выводы по работе оборудования представлены в пункте 2.2.3.

Пошаговое описание схемы обслуживания.

1. Пакет приходит по физическому кабелю и попадает в аппаратную очередь (буфер памяти) сетевой карты. Пакет может быть сброшен при переполнении очереди. Алгоритм сброса не учитывает приоритет пакета.
2. Используется DMA для переноса пакета из буфера сетевой карты в RxRing в порядке поступления (FIFO). RxRing находится в основной памяти устройства. В дальнейшем пакет будет забран на обслуживание и при необходимости перенесён в другие структуры выделенные области основной памяти. Выполняется классификация пакета. Пакет помещается во входную очередь, реализуемую компонентом qdisc. Очередь может предоставлять множество дополнительных возможностей: более сложные алгоритмы управления очередью (AQM), такие как RED, ограничение полосы пропускания, выравнивание полосы пропускания, а также приоритизацию обслуживания трафика. В отличие от рассмотренных ранее МЭ присутствует возможность использовать эти функции в произвольном порядке, хотя это не приветствуется и может давать отрицательный эффект на качество обслуживания трафика. Пакет может быть сброшен при переполнении очереди. Дополнительно в очереди доступна возможность по сборке фрагментированного трафика.
3. Выполняется распаковка и проверка целостности.

4. Пакет попадает в цепочку (chain) PREROUTING. Внутри цепочки содержится 3 базовые таблицы: raw, mangle, NAT, а также выполняется работа механизма отслеживания соединений (connection tracker). Через цепочку PREROUTING проходят все пакеты, попавшие на узел размещения МЭ.

Внутри базовых таблиц могут быть созданы дополнительные «вложенные» таблицы, переход в которые осуществляется на основе правила в родительской таблице. Возврат из вложенной таблицы всегда происходит в родительскую таблицу. Переход из таблицы в таблицу доступен только в рамках цепочки.

Выполняется проверка правил базовой таблицы raw цепочки PREROUTING. Эта таблица в основном используется для пометки пакетов с целью их пропуска через механизм отслеживания соединений (connection tracker) без обработки.

5. Выполняется работа механизма отслеживания состояний (connection tracker) в цепочке PREROUTING. Механизм отслеживает состояние сеансов связи различных протоколов. В базовой версии умеет работать с проколами TCP, UDP, ICMP. Полученная информация используется в других таблицах для фильтрации и обеспечения корректной работы различных сервисов.

6. Производится проверка правил базовой таблицы mangle в цепочке PREROUTING. Данная таблица предназначена для изменений полей в заголовках пакета. Имеется возможность помечать пакеты специальными метками, которые могут быть критериями проведения операций над пакетами. Метки действуют внутри узла, и при отправке пакета метка не отправляется. Пакет может быть сброшен при выполнении операции. Пакет может быть сброшен в результате выполнения операции.

7. Производится проверка правил базовой таблицы NAT в цепочке PREROUTING. Данная таблица предназначена для изменения IP-адреса назначения. Через таблицу проходят только пакеты открывающие соединение или пакеты, которые не может опознать механизм

отслеживания соединений. К пакетам в рамках установленного соединения будет автоматически применяться правило трансляции сетевого адреса, выбранное для пакета открывшего эту сессию. Пакет может быть сброшен в результате выполнения операции.

8. Производится определение назначения пакета. Если пакет предназначен узлу, на котором установлено МЭ, то переходим к операции № 9, если пакет предназначен другому узлу, то переходим к операции № 18. Пакет может быть сброшен в результате выполнения операции.

9. Пакет попадает в цепочку INPUT. Внутри цепочки содержится 2 базовые таблицы: mangle, filter, а также выполняется работа механизма отслеживания соединений. Через цепочку INPUT проходят пакеты, направляющиеся к узлу размещения МЭ (к его локальным процессам).

Выполняется проверка правил базовой таблицы mangle в цепочке INPUT. Эта таблица, как и в операции №6, предназначена для изменений полей в заголовках пакета, а также для маркировки пакетов. Пакет может быть сброшен в результате выполнения операции.

10. Производится проверка правил базовой таблицы filter в цепочке INPUT. Эта таблица предназначена для фильтрации трафика. Пакет может быть сброшен в результате выполнения операции.

11. Выполняется работа механизма отслеживания состояний в цепочке INPUT. Механизм отслеживания состояний выполняет те же функции, что и в операции 4.

12. Пакеты передаются на обработку локальным процессам (программам).

13. Пакет попадает в цепочку OUTPUT. Внутри цепочки содержится 4 базовые таблицы: raw, mangle, NAT, filter, а также выполняется работа механизма отслеживания соединений. Через цепочку OUTPUT проходят пакеты, созданные на узле размещения МЭ.

Выполняется проверка правил базовой таблицы raw в цепочке OUTPUT. Эта таблица, как и в операции №4, используется для пометки пакетов с

целью их пропуска через механизм отслеживания соединений (connection tracker) без обработки.

14. Выполняется работа механизма отслеживания состояний в цепочке OUTPUT. Механизм отслеживания состояний выполняет те же функции, что и в операции № 4 и 11.

15. Выполняется проверка правил базовой таблицы mangle в цепочке OUTPUT. Эта таблица, как и в операции № 6, 9, предназначена для изменений полей в заголовках пакета, а также для маркировки пакетов. Пакет может быть сброшен в результате выполнения операции.

16. Производится проверка правил базовой таблицы NAT в цепочке OUTPUT. Данная таблица предназначена для изменения IP-адреса источника. Функционирование таблицы аналогично тому, что описано в операции № 7. Пакет может быть сброшен в результате выполнения операции.

17. Производится проверка правил базовой таблицы filter в цепочке OUTPUT. Эта таблица предназначена для фильтрации трафика. Пакет может быть сброшен в результате выполнения операции.

18. Пакет попадает в цепочку FORWARD. Внутри цепочки содержится 2 базовые таблицы: mangle, filter. Через цепочку FORWARD проходят пакеты, направляющиеся к другим узлам сети, через (сквозь) узел размещения МЭ.
Выполняется проверка правил базовой таблицы mangle в цепочке FORWARD. Эта таблица, как и в операции № 6, 9, 15, предназначена для изменений полей в заголовках пакета, а также для маркировки пакетов. Пакет может быть сброшен в результате выполнения операции.

19. Производится проверка правил базовой таблицы filter в цепочке FORWARD. Эта таблица предназначена для фильтрации трафика. Пакет может быть сброшен в результате выполнения операции.

20. Выполняется маршрутизация пакета. Главным образом необходимо проводить маршрутизацию повторно потому, что адрес назначения мог

быть изменён в результате выполнения предыдущих операций. Пакет может быть сброшен в результате выполнения операции.

21. Пакет попадает в цепочку POSTROUTING. Внутри цепочки содержится 2 базовые таблицы: mangle, NAT, а также выполняется работа механизма отслеживания соединений. Через цепочку POSTROUTING проходят все пакеты покидающие узел размещения МЭ. Пакет может быть сброшен в результате выполнения операции.

Выполняется проверка правил базовой таблицы mangle в цепочке POSTROUTING. Эта таблица, как и в операции № 6, 9, 15, 18 предназначена для изменений полей в заголовках пакета, а также для маркировки пакетов.

22. Производится проверка правил базовой таблицы NAT в цепочке POSTROUTING. Данная таблица предназначена для изменения IP-адреса источника. Изменение IP-адреса назначения не допускается, так как уже принято решение о маршрутизации. Через данную таблицу проходят только пакеты открывающие соединение или пакеты, которые не может опознать механизм отслеживания состояний. К пакетам в рамках установленного соединения будет автоматически применяться правило трансляции сетевого адреса, выбранное для пакета открывшего это соединение. Пакет может быть сброшен в результате выполнения операции.

23. Выполняется работа механизма отслеживания состояний в цепочке INPUT. Механизм отслеживания состояний выполняет те же функции, что и в операциях № 4, 11, 14.

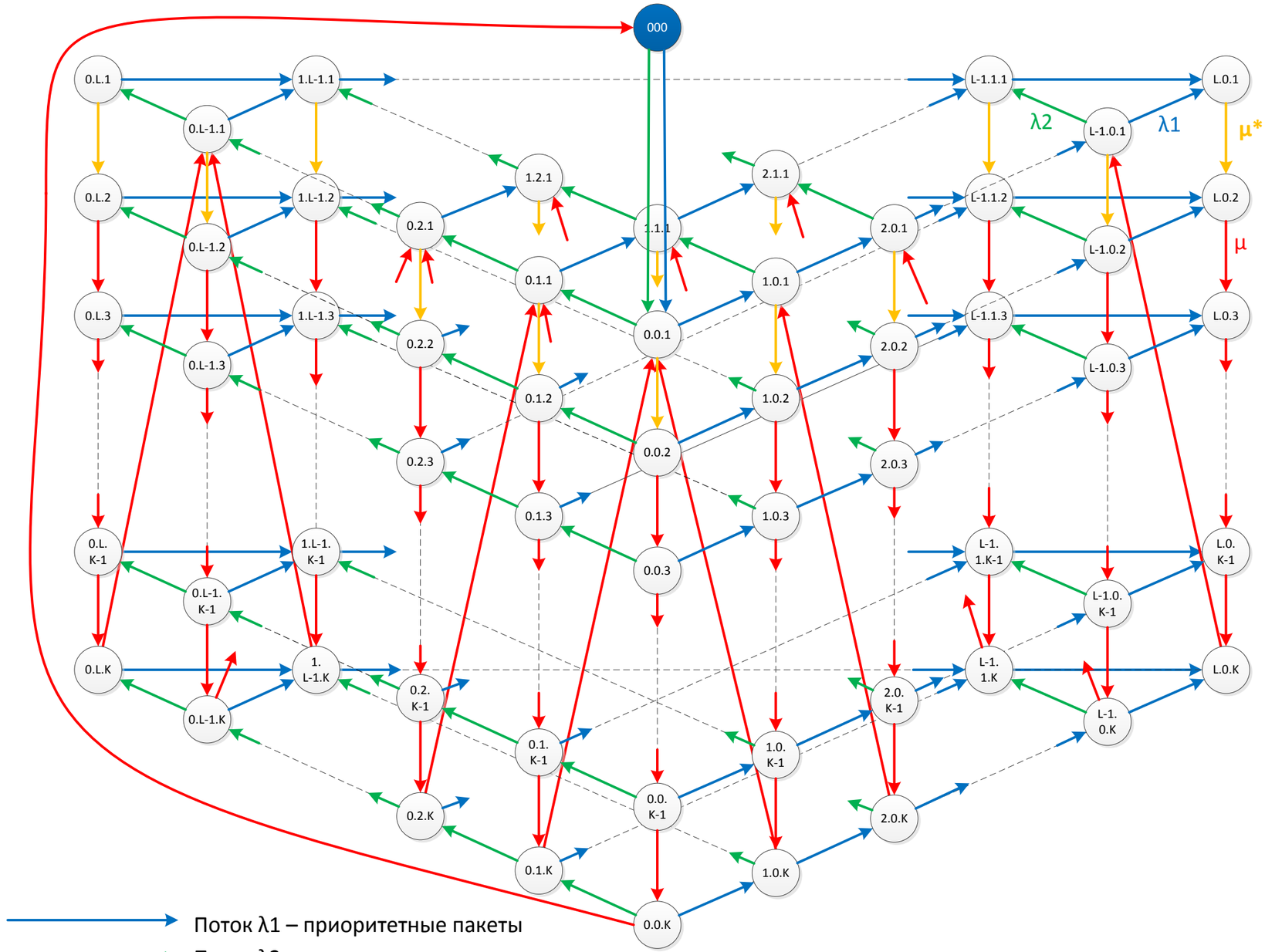
24. Выполняется фрагментация пакетов, если это необходимо.

25. Пакет помещается в выходную очередь, реализуемую компонентом qdisc. Возможности выходной очереди аналогичны тем, что предоставляет входная очередь (см. операцию №3). Пакет может быть сброшен при переполнении очереди или в результате ограничения или выравнивания потока трафика.

26. Пакет попадает в TxRing (всё ещё в основной памяти) и затем с использованием DMA переносится в аппаратную очередь сетевой карты. Отправка из очереди осуществляется в соответствии с дисциплиной FIFO.

Приложение Б.

Диаграмма состояний и переходов



Поток λ1 – приоритетные пакеты

Поток λ2 – неприоритетные пакеты

Поток с параметром μ* для первой фазы обслуживания

Поток с параметром μ для остальных (К-1) фаз обслуживания

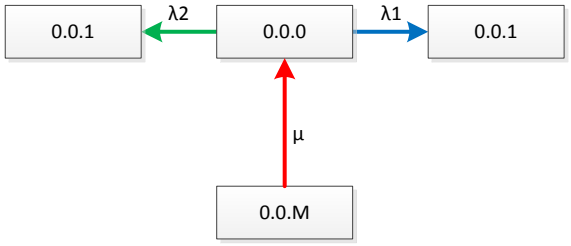
Рисунок Б.1 – Диаграмма состояний и переходов

Приложение В.

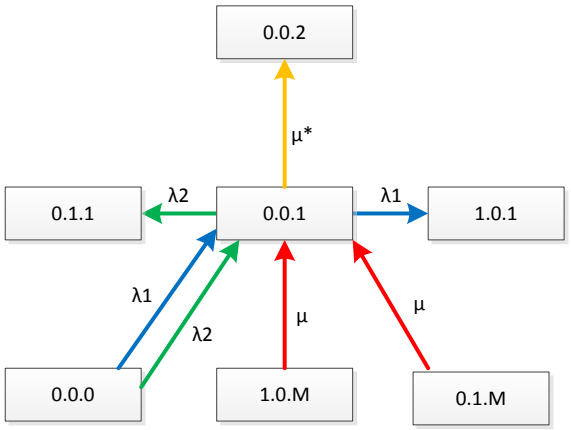
Формирование диаграммы состояний и переходов и системы уравнений равновесия

Для формирования диаграммы состояний и переходов, представленной на рисунке Б.1, а также СУР, необходимо было сформировать упрощённый набор диаграмм, по которым проще написать отдельные уравнения из состава СУР.

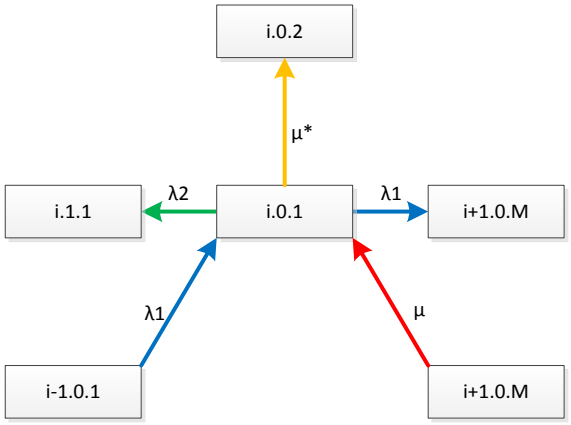
$$(\lambda_1 + \lambda_2) p_{0.0.0} = \mu p_{0.0.M}, \quad (1)$$



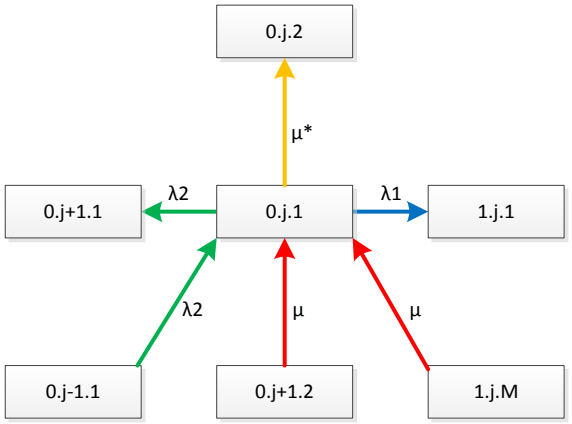
$$(\lambda_1 + \lambda_2 + \mu^*) p_{0.0.1} = (\lambda_1 + \lambda_2) p_{0.0.0} + \mu p_{1.0.M} + \mu p_{0.1.M}, \quad (2)$$



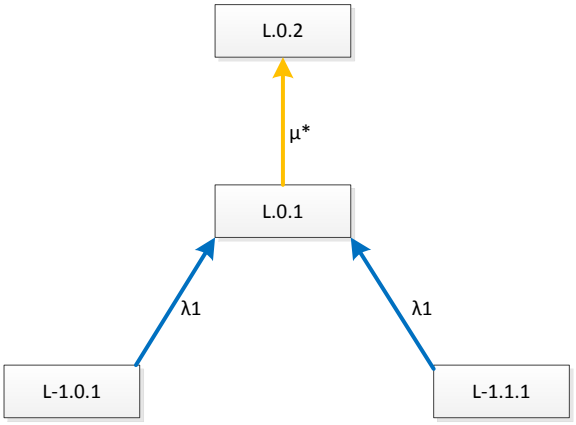
$$(\lambda_1 + \lambda_2 + \mu^*) p_{i.0.1} = \lambda_1 p_{i-1.0.1} + \mu p_{i+1.0.M}, \; 0 < i < L, \quad (3)$$



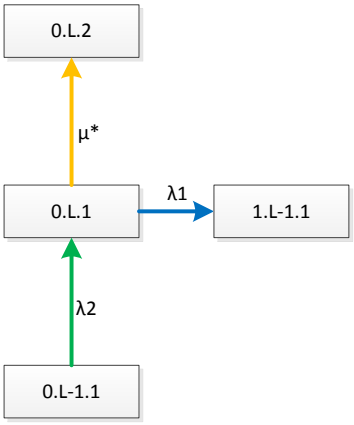
$$(\lambda_1 + \lambda_2 + \mu^*) p_{0.j.1} = \lambda_2 p_{0.j-1.1} + \mu p_{0.j+1.M} + \mu p_{1.j.M} \quad ,0 < j < L, \tag{4}$$



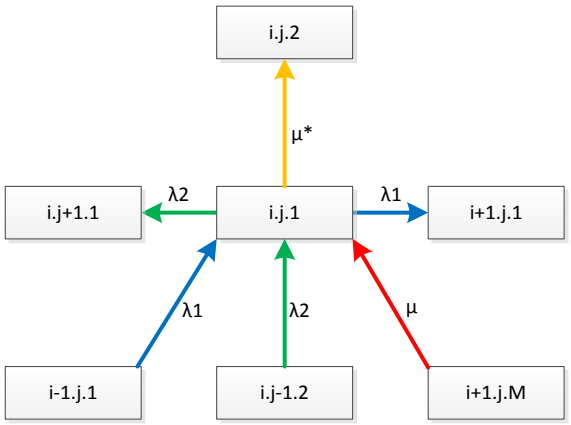
$$\mu^* p_{L.0.1} = \lambda_1 p_{L-1.0.1} + \lambda_1 p_{L-1.1.1}, \tag{5}$$



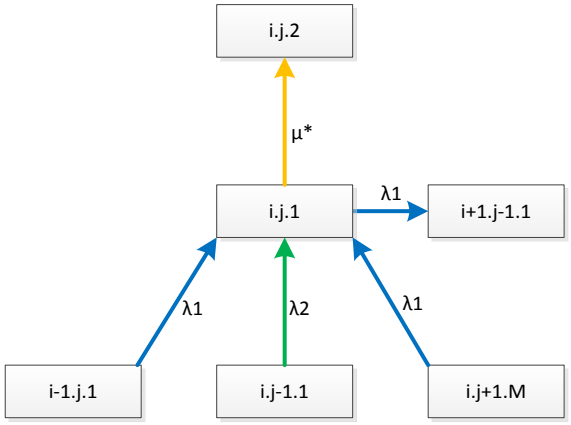
$$(\lambda_1 + \mu^*) p_{0.L.1} = \lambda_2 p_{0.L-1.1}, \tag{6}$$



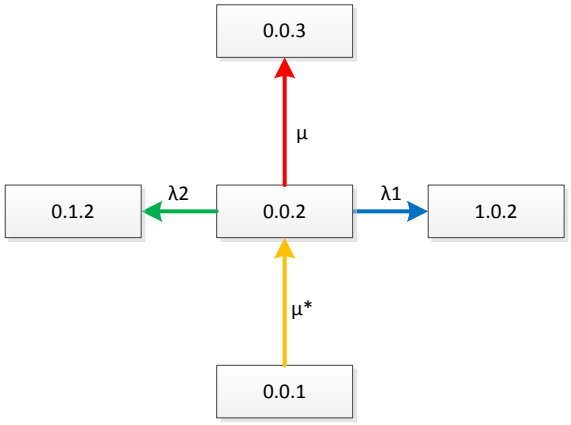
$$(\lambda_1 + \lambda_2 + \mu^*) p_{i.j.1} = \lambda_1 p_{i-1.j.1} + \lambda_2 p_{i.j-1.1} + \mu p_{i+1.j.M},\ i + j < L,\ i > 0,\ j > 0, \quad (7)$$



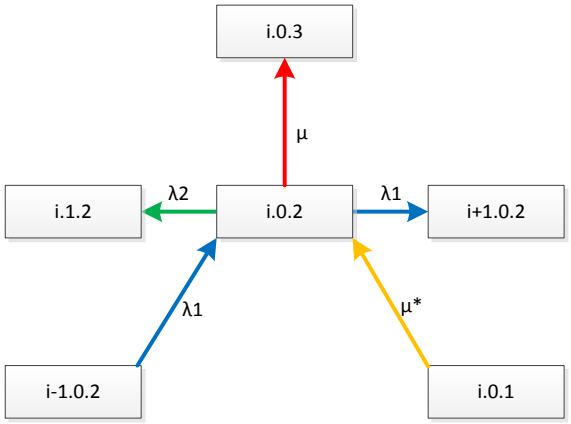
$$(\lambda_1 + \mu^*) p_{i.j.1} = \lambda_1 p_{i-1.j.1} + \lambda_2 p_{i.j-1.1} + \lambda_1 p_{i-1.j+1.1},\ i + j = L,\ i > 0,\ j > 0, \quad (8)$$



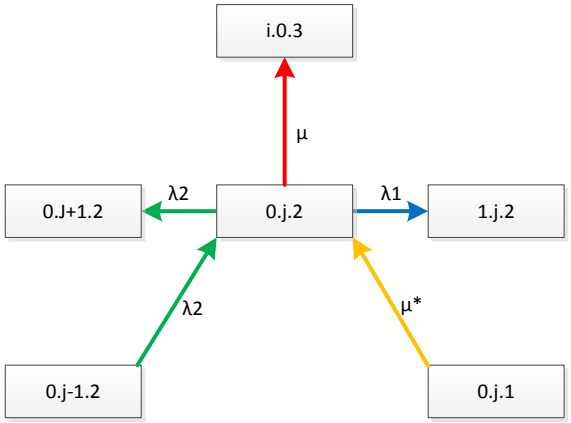
$$(\lambda_1 + \lambda_2 + \mu) p_{0.0.2} = \mu^* p_{0.0.1}, \quad (9)$$



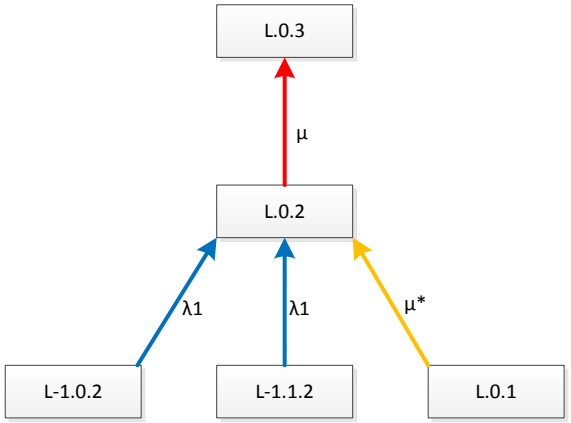
$$(\lambda_1 + \lambda_2 + \mu) p_{i.0.2} = \lambda_1 p_{i-1.0.2} + \mu^* p_{i.0.1}, \ 0 < i < L, \tag{10}$$



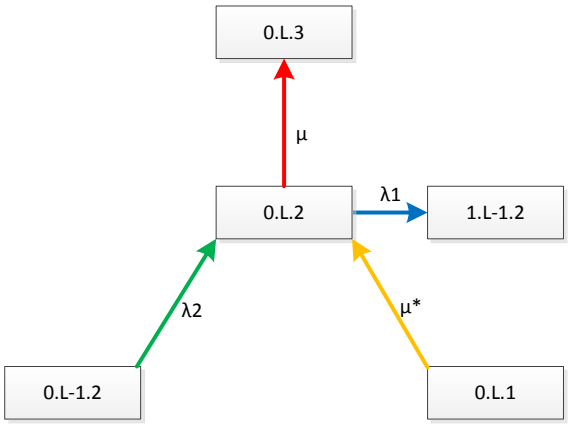
$$(\lambda_1 + \lambda_2 + \mu) p_{0.j.2} = \lambda_2 p_{0.j-1.2} + \mu^* p_{0.j.1}, \ 0 < j < L, \tag{11}$$



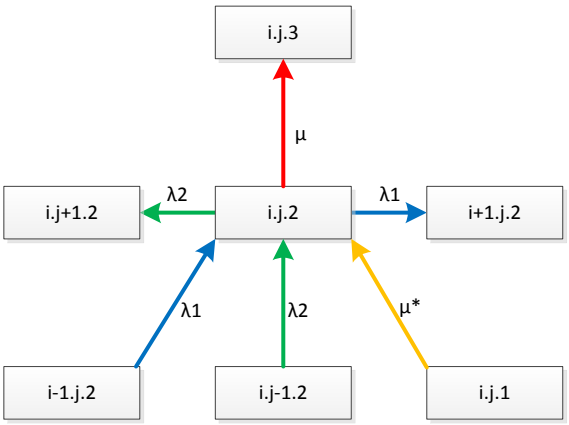
$$\mu p_{L.0.2} = \lambda_1 p_{L-1.0.2} + \lambda_1 p_{L-1.1.2} + \mu^* p_{L.0.1}, \tag{12}$$



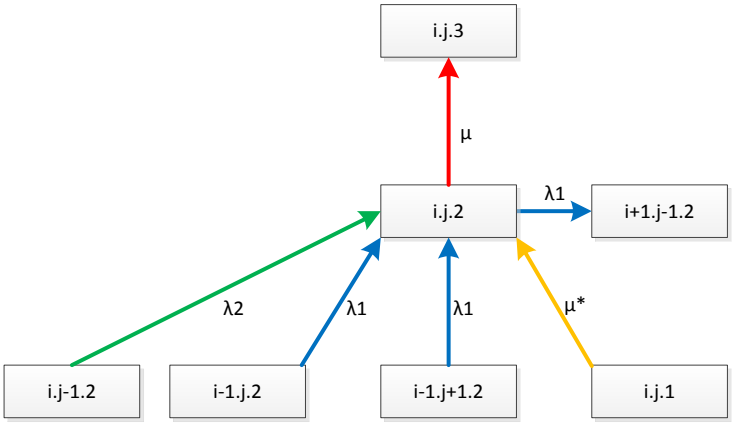
$$(\lambda_1 + \mu) p_{0.L.2} = \lambda_2 p_{0.L-1.2} + \mu^* p_{0.L.1}, \tag{13}$$



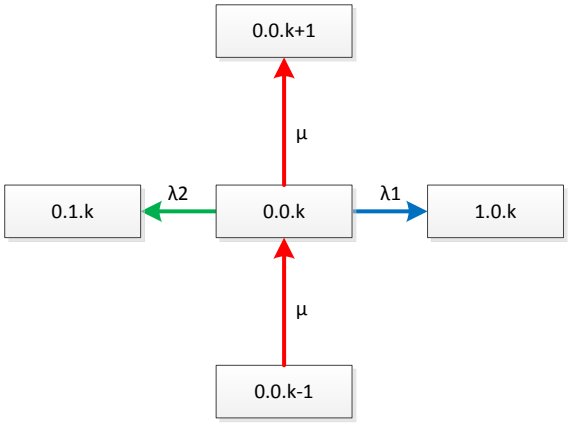
$$(\lambda_1+\lambda_2+\mu)p_{i.j.2}=\lambda_1 p_{i-1.j.2}+\lambda_2 p_{i.j-1.2}+\mu^* p_{i.j.1}, i+j<L,\ i>0,\ j>0, \quad (14)$$



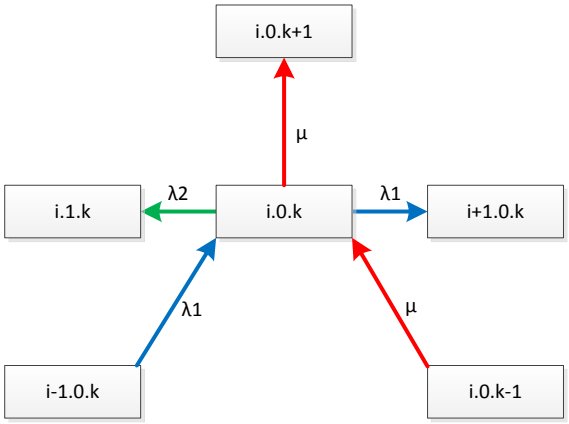
$$(\lambda_1+\mu)p_{i.j.2}=\lambda_1 p_{i-1.j.2}+\lambda_2 p_{i.j-1.2}+\lambda_1 p_{i-1.j+1.2}+\mu^* p_{i.j.1}, i+j=L,\ i>0,\ j>0, \quad (15)$$



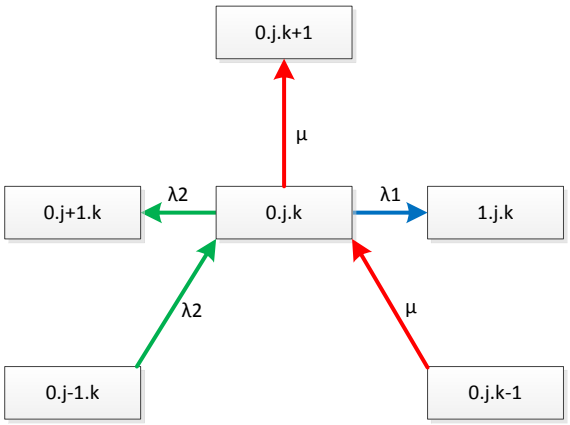
$$(\lambda_1+\lambda_2+\mu)p_{0.0.k}=\mu p_{0.0.k-1},\ 2<k\le M, \quad (16)$$



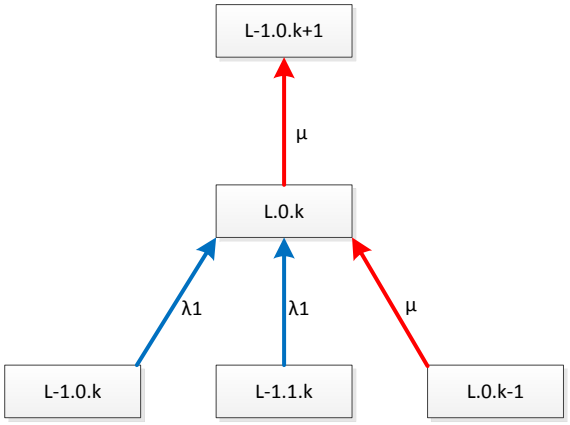
$$(\lambda_1+\lambda_2+\mu)p_{i.0.k}=\lambda_1 p_{i-1.0.k}+\mu p_{i.0.k-1},\ 0<i<L,\ 2<k\le M, \quad (17)$$



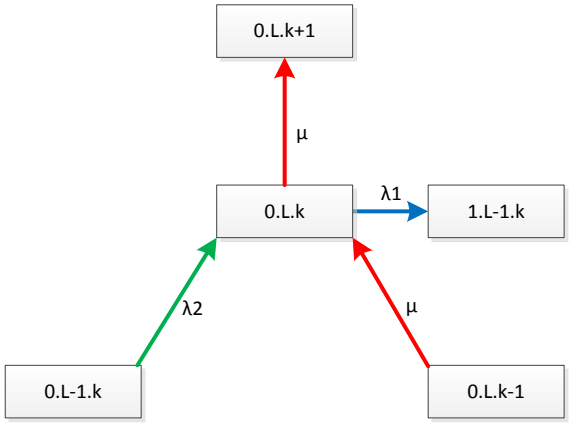
$$(\lambda_1+\lambda_2+\mu)p_{0.j.k}=\lambda_2 p_{0.j-1.k}+\mu p_{0.j.k-1},\ 0<j<L,\ 2<k\le M, \tag{18}$$



$$\mu p_{L.0.k}=\lambda_1 p_{L-1.0.k}+\lambda_1 p_{L-1.1.k}+\mu p_{L.0.k-1},\ 2<k\le M, \tag{19}$$

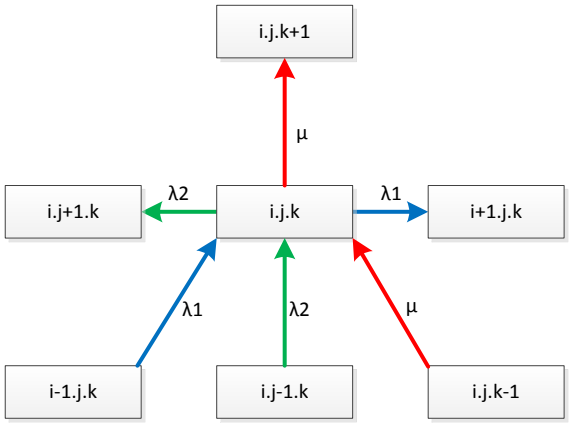


$$(\lambda_1+\mu)p_{0.L.k}=\lambda_2 p_{0.L-1.k}+\mu p_{0.L.k-1},\ 2<k\le K, \tag{20}$$



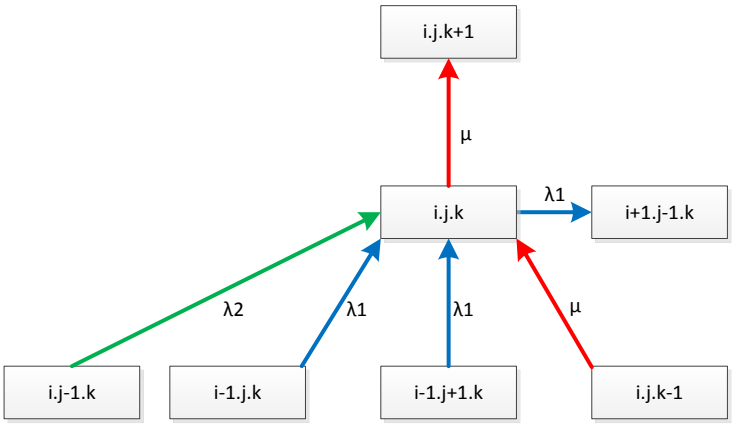
$$(\lambda_1+\lambda_2+\mu)p_{i.j.k}=\lambda_1 p_{i-1.j.k}+\lambda_2 p_{i.j-1.k}+\mu p_{i.j.k-1}, \quad (21)$$

$$i+j<L,\ i>0,\ j>0, 2<k\le M,$$



$$(\lambda_1+\mu)p_{i.j.k}=\lambda_1 p_{i-1.j.k}+\lambda_2 p_{i.j-1.k}+\lambda_1 p_{i-1.j+1.k}+\mu p_{i.j.k-1}, \quad (22)$$

$$i+j=L,\ i>0,\ j>0,\ 2<k\le M.$$



Приложение Г.

Код математической модели на языке Wolfram Mathematica

Обнуление внутренних переменных (необходимо при повторном расчёте)

Clear[*i*, *j*, *k*, *x*, *g*, $\lambda 1$, $\lambda 2$, μ, $\mu 1$, *L*, *M*, *a*, *b*, *c*, Φ, R, *p*, *comfac*, *res*, *MatYVar*, *P*1*loss*, *P*2*loss*, *PlossUnpr*, *PlossPrior*, *PlossDenial*, *Qprior*, *Qunprior*];

Ввод исходных данных

$\lambda 1, \lambda 2, \mu, \mu 1, L, M$;

Создание векторов, хранящих субматрицы

$a = Array[0, L + 1]$;

$b = Array[0, L + 2]$;

$c = Array[0, L + 1]$;

Заполнение матриц типа "A" (индекс матрицы совпадает с индексом хранения в векторе т.е. A_1 = a[[1]])

Заполнение матриц типа "A_1"

```
a[[1]] = ConstantArray[0, {1, M}];
a[[1]][[1, 1]] = λ1 + λ2;
```

Заполнение матриц типа "$A_{2...L+1}$"

```
For[i = 2, i ≤ L + 1, i++,
 a[[i]] = ConstantArray[0, {(i − 1)·M, i·M}];
 For[j = 1, j ≤ (i − 1)·M, j++,
  a[[i]][[j, j]] = λ2;
  a[[i]][[j, j + M]] = λ1]];
Clear[i, j];
```

Заполнение матриц типа "B" (индекс матрицы меньше на еденицу индекса хранения в векторе т.е. B_0 = b[[1]])

Заполнение матриц типа "B_0"

```
b[[1]] = ConstantArray[−λ1 − λ2, {1, 1}];
```

Заполнение матриц типа "B_1"

```
b[[2]] = ConstantArray[0, {M, M}];
b[[2]][[1, 1]] = -λ1 - λ2 - μ1;
b[[2]][[1, 2]] = μ1;
For[i = 2, i ≤ M, i++,
 b[[2]][[i, i]] = -λ1 - λ2 - μ];
For[i = 2, i ≤ M - 1, i++, b[[2]][[i, i + 1]] = μ];
Clear[i]
```

Заполнение матриц типа "$B_{2...L}$"

```
For[i = 2, i ≤ L, i++,
 b[[(i + 1)]] = ConstantArray[0, {i·M, i·M}];
 For[j = 1, j ≤ i·M, j += M,
  b[[(i + 1)]][[j, j]] = -λ1 - λ2 - μ1;
  b[[(i + 1)]][[j, j + 1]] = μ1;
  For[x = j + 1, x ≤ i·M, x++,
   b[[i + 1]][[x, x]] = -λ1 - λ2 - μ];
   For[x = j + 1, x ≤ i·M - 1, x++,
    b[[i + 1]][[x, x + 1]] = μ];
   For[k = M + 1, k ≤ i·M, k += M,
    b[[i + 1]][[k - 1, k]] = 0]]];
Clear[i, j, k, x]
```

Заполнение матриц типа "B_{L+1}"

```
b[[(L + 2)]] = ConstantArray[0, {(L + 1)·M, (L + 1)·M}];
For[j = 1, j ≤ L·M, j += M,
 b[[(L + 2)]][[j, j]] = -λ1 - μ1;
 b[[(L + 2)]][[j, j + 1]] = μ1;
 For[x = j + 1, x ≤ L·M, x++,
  b[[L + 2]][[x, x]] = -λ1 - μ];
 For[x = j + 1, x ≤ L·M - 1, x++,
  b[[L + 2]][[x, x + 1]] = μ];
```

```
For[k = M + 1, k ≤ L·M, k += M,
 b[[L + 2]][[k − 1, k]] = 0]]
b[[(L + 2)]][[L·M + 1, L·M + 1]] = −μ1;
b[[(L + 2)]][[L·M + 1, L·M + 2]] = μ1;
For[x = L·M + 2, x ≤ (L + 1)·M, x++,
 b[[L + 2]][[x, x]] = −μ];
For[x = L·M + 2, x ≤ (L + 1)·M − 1, x++,
 b[[L + 2]][[x, x + 1]] = μ];
For[g = 1, g ≤ L·M, g++,
 b[[L + 2]][[g, g + M]] = λ1];
Clear[i, j, k, x, g]
```

Заполнение матриц типа "C" (индекс матрицы меньше на еденицу индекса хранения в векторе т.е. C_0 = c[[1]])

Заполнение матриц типа "C_0"

```
c[[1]] = ConstantArray[0, {M, 1}];
c[[1]][[M, 1]] = μ;
```

Заполнение матриц типа "C_1"

```
c[[2]] = ConstantArray[0, {2·M, M}];
c[[2]][[M, 1]] = μ;
c[[2]][[2·M, 1]] = μ;
```

Заполнение матриц типа "$C_{2...L}$"

```
For[i = 2, i ≤ L, i++,
 c[[i + 1]] = ConstantArray[0, {(i + 1)·M, i·M}];
 c[[i + 1]][[M, 1]] = μ;
 c[[i + 1]][[2·M, 1]] = μ;
 For[j = 3, j ≤ i + 1, j++,
  c[[i + 1]][[j·M, (j - 2)·M + 1]] = μ]];
Clear[i, j]
```

Создаём матрицы "Ф, R_r" (индекс матрицы меньше на еденицу индекса хранения в векторе т.е. $\Phi_0 = \Phi[[1]], R_0 = R[[1]]$)

```
Ф = Array[0, {L+ 2}]
R = Array[0, {L+ 2}]
```

Используем рекуррентные формулы расчётов

```
Ф[[L + 2]] = b[[L + 2]];
For[i = L + 1, i ≥ 0, i − −,
  Ф[[i + 1]] = Inverse[Ф[[i + 1]]];
  R[[i]] = −1 · a[[i]].Ф[[i+ 1]];
  Ф[[i]] = b[[i]] + R[[i]].c[[i]]
Clear[i]
```

Расчёт матрицы результирующих векторов (res)

```
p = ConstantArray[0, {L + 2, 1}];
p[[1, 1]] = Constant Array[1, {1, 1}]
For[i = 1, i <= L + 1, i + +,
  p[[i + 1, 1]] = p[[i, 1]].r[[i]]];
comfac = Total[p, L + 2]
Clear[i]
For[i = 1, i <= L + 2, i + +,
 p[[i, 1]] = p[[i, 1]] / comfac]
res = p[[1, 1]]
For[i = 2, i <= L + 2, i + +,
 res = Join[res, p[[i, 1]], 2]]
```

Проверка вектора, сумма вероятностей должна быть равна 1:

```
Total[res, 2]
Clear[i]
```

Функция формирования матрицы

Clear[*i*, *j*, *l*, *k*]

MatYVar[*i* _, *j* _, *k* _, *M* _] =

If[*i* == 0 && *j* == 0 && *k* == 0, var1 = 1,

If[*i* == 0 && *j* == 0, var1 = *k* + 1,

If[*i* == 0 && *j* > 0, var1 = 1 + *M* × $\sum_{x=0}^{j} x + k$,

If[*i* > 0 && *j* == 0, var1 = 1 + *M* · *i* + *M* · $\sum_{x=0}^{j} x + k$,

If[*i* > 0 && *j* > 0, var1 = *MatYVar*[0, (*i* + *j*), *M*, *M*] + *M* · (*i* - 1) + *k*]]]]]

Вторая проверки СУР и корректности работы функции формирования матриц:

$\sum_{i=0}^{L} \sum_{j=0}^{L-i} \sum_{k=1}^{M}$ Abs[res[[1, MatYVar[i, j, k, M]]]] + Abs[res[[1, MatYVar[0, 0, 0, M]]]]

Рассчитываем вероятности потерь пакетов

Расчитываем вероятность сброса приоритетного пакета при переполненнии очереди

$$P_{loss\,\mathrm{Prior}} = \frac{\lambda 1}{\lambda 1 + \lambda 2} \cdot \sum_{k=1}^{M} \mathrm{res}[[1, \mathrm{MatYVar}[L, 0, k, M]]]$$

Расчитываем вероятность вытеснения из переполненной очереди неприоритетного пакета при приходе приоритетного пакета

$$P_{1loss} = \frac{\lambda 1}{\lambda 1 + \lambda 2} \cdot \sum_{j=0}^{L-i} \sum_{k=1}^{M} \mathrm{res}[[1, \mathrm{MatYVar}[i, L - i, k, M]]]$$

Расчитываем вероятность сброса неприоритетного пакета при переполненнии очереди

$$P_{2loss} = \frac{\lambda 1}{\lambda 1 + \lambda 2} \cdot \sum_{j=0}^{L} \sum_{k=1}^{M} \mathrm{res}[[1, \mathrm{MatYVar}[i, L - i, k, M]]]$$

Рассчитываем среднее количество заявок в очереди

Рассчитываем среднее количество приоритетных заявок в очереди

$$Q_{prior} = \sum_{i=0}^{L} \sum_{j=1}^{L-i} \sum_{k=1}^{M} (i \cdot \mathrm{res}[[1, \mathrm{MatYVar}[i, j, k, M]]])$$

Рассчитываем среднее количество неприоритетных заявок в очереди

$$Q_{unptior} = \sum_{i=0}^{L} \sum_{j=1}^{L-i} \sum_{k=1}^{M} (j \cdot \mathrm{res}[[1, \mathrm{MatYVar}[i, j, k, M]]])$$

Приложение Д.

Применение паттернов проектирования в имитационной модели

При разработке применялось большее число паттернов проектирования, чем те, что перечислены ниже, но описанные ниже следует считать основополагающими для реализации имитационной модели.

Abstract factory

Паттерн abstract factory применяется в фабриках:

– RandomFacktory;

– ExponentialBasicRadnomFactory;

– ExponentialNumericRandomFactory;

– ExponentialNumericRandomMersenneTwisterFactory;

– PacketFactory.

Применение паттерна abstract factory продиктовано тем, что переменная «случайное число» и объект «пакет» могут быть сгенерированы различными методами и с использованием различных классов, в зависимости от исходных данных, введённых пользователем. Решение подобных задач с использованием паттерна abstract factory является эффективным и удобным.

Command

Паттерн command применяется при работе кнопок графического интерфейса пользователя. Паттерн инкапсулирует запросы, создаваемые нажатием кнопок, в качестве объектов, позволяя передавать их в качестве параметров, ставить их в очередь на исполнение, записи действий в журнал и обеспечивать отмену операций.

Facade

Паттерн facade применяется в ProcessingComposer (компоновщик обработчиков), который описывает комплекс операций с пакетами, доступными в имитационной модели. Паттерн facade предоставляет единый интерфейс доступа к этой группе операций, при этом определяя, какую из операций необходимо

выполнить в тот или иной момент вызова на основе передаваемого в facade контекста.

Flyweight

Паттерн flyweight применяется для реализации пакетов (легковесных объектов). Паттерн позволяет выполнять эффективную совместную работу с малыми объектами (пакетами). В случае имитационной модели паттерн flyweight работает в паре с паттерном object pool.

Null object

Паттерн null object предоставляет доступ к пустому объекту, который не выполняет никаких действий и доступен по умолчанию. Например, если на консоль не нужно ничего выводить, то в вывод отправляется null object, что позволяет не создавать другие нулевые объекты.

Object pool

Паттерн Object pool применяется в пакетных фабриках. Паттерн обеспечивает эффективное повторное использование объектов типа пакет. После того как пакет уже использован в системе, он не создаётся заново, а получает новые свойства (характеристики) и снова попадает в СМО.

Singleton

Паттерн singleton применяется в объектах в составе ContainerControlled. Паттерн гарантирует, что алгоритм, сам по себе не имеющий состояния, упакованный в Processor не будет запущен более чем в одном экземпляре.

Приложение Е.

Руководство пользователя программы имитационного моделирования

1. Информация общего характера

1.1. Назначение имитационной модели

Имитационная модель предназначена для воспроизведения поведения межсетевого экрана типа Linux NetFilter/IPTables, функционирующего в условиях приоритизации трафика, и получения его показателей производительности (обслуживания трафика).

1.2. Способ реализации имитационной модели

Для реализации модели выбран подход, именуемый дискретно-событийным моделированием. Имитационная модель реализована на языке C# в среде Microsoft Visual Studio на основе Microsoft.NET 4.0.

2. Системные требования

Системные требования имитационной модели представлены в таблице Е.1.

Таблица Е.1 – Системные требования имитационной модели

Компонент	Требования
Компьютер и процессор	1 гигагерц (ГГц) и выше, 32- или 64-разрядный процессор
Память (ОЗУ)	от 1 гигабайт (ГБ) ОЗУ, необходимый объём ОЗУ определяется сложностью проводимый расчётов
Жесткий диск	10 мегабайт (МБ) свободного места на жёстком диске + свободное место для хранения выводимых результатов моделирования (1000 расчётов по 30 реализаций занимают около 1,2 мегабайта (МБ) свободного места на жёстком диске)
Операционная система	Работа поддерживается следующими операционными системами Windows: – Windows 10 (32- или 64-разрядная версия); – Windows 8.1 (32- или 64-разрядная версия); – Windows 8 (32-разрядная или 64-разрядная версия); – Windows 7 (32-разрядная или 64-разрядная версия); – Windows Server 2012 R2 (64-разрядная версия); – Windows Server 2012 (64-разрядная версия); – Windows Server 2008 R2 (64-разрядная версия).

Компонент	Требования
Сопутствующее ПО	Для работы с отчётами, выводимыми имитационной моделью, подойдёт любой табличный редактор, воспринимающий файлы формата *.CSV (функционирование проверено для Microsoft Excel, входящего в комплект Microsoft Office 2007, 2010, 2013 и LibreOffice 5).
Версия .NET	4.0 или выше

2.1. Принятая модель функционирования и допущения

2.1.1. Описание принятой модели

В имитационной модели воспроизводит не полный процесс обслуживания пакетов в Linux NetFilter/IPtables. Подобное ограничение было принято для упрощения модели и обусловлено описанными далее допущениями.

На рисунке Е.1 отражена модель функционирования межсетевого экрана.

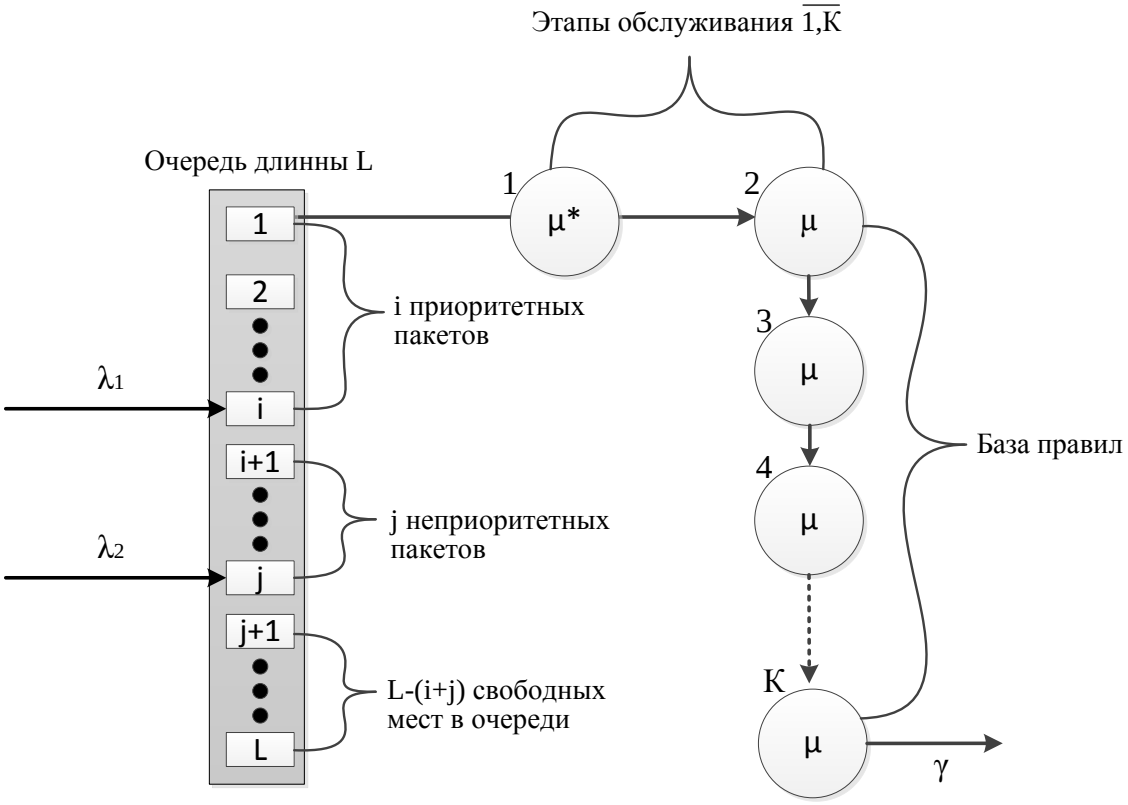


Рисунок Е.1 – Модель функционирования межсетевого экрана

Краткое описание модели обслуживания. В систему поступают два независимых потока пакетов с интенсивностями λ_1 и λ_2. Обслуживание состоит из K этапов, время обслуживания заявок на которых является постоянным или

случайным, с равномерным или экспоненциальным распределением с параметрами μ_1 – для первого этапа и μ – для остальных $(K-1)$ этапов. Пакеты попадают в очередь длины L, в которой может вместиться i-приоритетных пакетов и j-неприоритетных пакетов, при условии, что $i + j \le L$.

Входящие потоки пакетов (вызовов). На вход имитационной модели поступает два потока пакетов: приоритетный и неприоритетный с интенсивностями λ_1 и λ_2. Каждый поток состоит из пакетов (заявок), время между приходом которых может быть постоянной величиной или случайной величиной, распределённой по экспоненциальному или равномерному законам. Потоки можно лишать свойства приоритетности, в этом случае один из них не будет иметь приоритета над вторым.

Постановка в очередь (пакетный буфер). Пакеты, поступающие на вход модели, попадают в очередь способную вместить не более L пакетов обоих приоритетов.

В режим с приоритетом обслуживания приоритетные пакеты способны выдавливать неприоритетные пакеты из очереди, в случае если очередь переполнена.

В режиме без приоритетов обслуживания пакеты застав систему переполненной теряются.

Постановка на обслуживание. Пакеты попадают на обслуживание из очереди в соответствие с выбранной дисциплиной обслуживания. Прервать уже начатое обслуживание пакета нельзя.

В режиме с приоритетом обслуживания приоритетные пакеты всегда будут выбираться из очереди на обслуживание первыми. Неприоритетные пакеты способны попасть на обслуживание только в том случае, если приоритетных пакетов в очереди нет. В рамках каждого из приоритетов постановка на обслуживание осуществляется по дисциплине FIFO.

В режиме без приоритета обслуживания все пакеты попадают на обслуживание из очереди в соответствии с дисциплиной FIFO.

Окончание обслуживания. Когда пакет оканчивает своё обслуживание, он покидает систему. Учитывая допущения модели, нет разницы, покидает пакет систему по запрещающему или разрешающему правилам.

2.1.2. Допущения

Допущение 1. Межсетевой экран подключен к сети связи, функционирующей на основе технологии Ethernet на канальном уровне и технологии IP на сетевом уровне.

Допущение 2. Межсетевой экран расположен на границе между сетями клиента и оператора доступа или на границе между сетями двух операторов доступа.

Допущение 3. Процесс функционирования сетевой карты, процесс переноса пакета из памяти сетевой карты в память межсетевого экрана и процесс переноса пакета между программами очередями межсетевого экрана не рассматривается, задержка этого процесса не учитывается. Считать, что сетевая карта имеет пропускную способность, заметно превышающую возможности межсетевого экрана.

Допущение 4. Рассматривается не полный цикл приёма-передачи пакета, а комплекс операций от получения пакета до фильтрации включительно. Не имеет значение, будет ли мы рассматриваться фильтрация до таблицы filtering в цепочке INPUT или до таблицы filtering в цепочке forwarding. Будем считать, что и в первой и во второй цепочке одинаковое количество правил, или считать, что весь трафик проходит сквозь одну из цепочек, не затрагивая вторую. По этой причине формируемая модель будет линейной, без ветвления, демонстрирующего цепочки INPUT и FORWARD.

Допущение 5. В модели пренебрегается различием в задержке, порождаемой при классификации пакетов различными методами. Это допущение влияет на статическую задержку, вносимую в процесс обслуживания трафика устройством. Проверки меток на канальном и сетевом уровнях будут производиться с различной задержкой, по причине необходимости распаковки пакета, однако, разница незначительна.

Допущение 6. В модели не рассматриваются механизмы управления полосой пропускания: ограничение и выравнивание полосы пропускания. Условно можно считать, что входящие потоки уже являются результатом ограничения полосы пропускания.

Допущение 7. В модели рассматриваются две дисциплины обслуживания FIFO и дисциплина обслуживания с приоритетом, реализуемая алгоритмами PRIOR (IPTables/NetFilter). Аналогичные алгоритмы у других рассмотренных разработчиков МЭ именуются LLQ (Cisco), stick-priority queuing (Juniper).

Допущение 8. В модели рассматривается фильтрация без учёта состояния соединения.

Допущение 9. Входящих потоков всего два: с приоритетом обслуживания и без приоритета. Допущение было взято для уменьшения количества состояний системы и, как следствие, уменьшения ресурсов, необходимых для машинных расчётов.

Допущение 10. В модели не рассматривается возможность того, что пакет был повреждён при передаче или для него не был найден маршрут для его передачи.

2.2. Особенности работы с моделью с точки зрения пользователя

Относительность модельного времени. Все величины, зависящие от времени, в имитационной модели имеют размерность на абстрактную единицу времени. Это подразумевает, что пользователь имитационной модели должен вводить значения в одинаковой размерности, например все временные характеристики в наносекундах или в микросекундах и т.п. Выходные величины будут иметь ту же размерность, что и входные величины. Такой подход усложняет работу пользователя, но позволяет не использовать внутреннего приведения величин к единой размерности, что положительно сказывается на точности вычислений.

Рекомендуется принимать за единицу времени, минимально значащую для расчёта величину. Например, если в моделируемой системе все операции производятся в пределах, например, наносекунды, то за узловую единицу времени

необходимо её и принять. Выбор пикосекунды будет неоправданно усложнять расчёт и анализ результатов, не повышая точности расчёта. Выбор микросекунды не обеспечит необходимую точность.

Вывод результатов. Сокращённый вывод результатов осуществляется в главное окно программы. Расширенный вывод результатов осуществляется в отчуждаемый файл формата results.csv, с которым способны работать большинство табличных редакторов.

ВНИМАНИЕ! Файл results.csv не должен быть открыт в любой сторонней программе на момент окончания процесса моделирования иначе ПО моделирования не получит к нему доступ и не будет иметь возможность записать результаты моделирования.

3. Использование имитационной моделью

3.1. Запуск имитационной модели

Для запуска ПО имитационного моделирования необходимо зайти в папку, содержащую имитационную модель, и запустить ярлык «Musatov.Modelling.App» (расширение *.exe).

По завершению запуска пользователю на экране отобразиться главное окно имитационной модели рисунок Е.2.

3.2. Описание главного окна

В рабочем окне программы нет красных цифр, отражённых на рисунке Е-2.

Поле 1 – Входные параметры. При нажатии на всю верхнюю плоскость окна на уровне пункта «Входные параметры» будут скрыты (поля 2-17). Поле 18 – поле вывода будет расширено, для облегчения чтения результатов. Поля 19, 20, 21 останутся на своих местах. Вторичное нажатие на пункт «Входные параметры» вернёт скрытые пункты. Результат выполнения пункта 1 смотри на рисунке Е.3. Для примера на рисунке осуществлён вывод тестовых результатов.

При появлении результатов в поле вывода (18) появиться вывод результатов. Поле вывода будет снабжено ползунками для прокручивания окна вывода. Если все результаты будут помещаться в окне, то ползунки исчезнут.

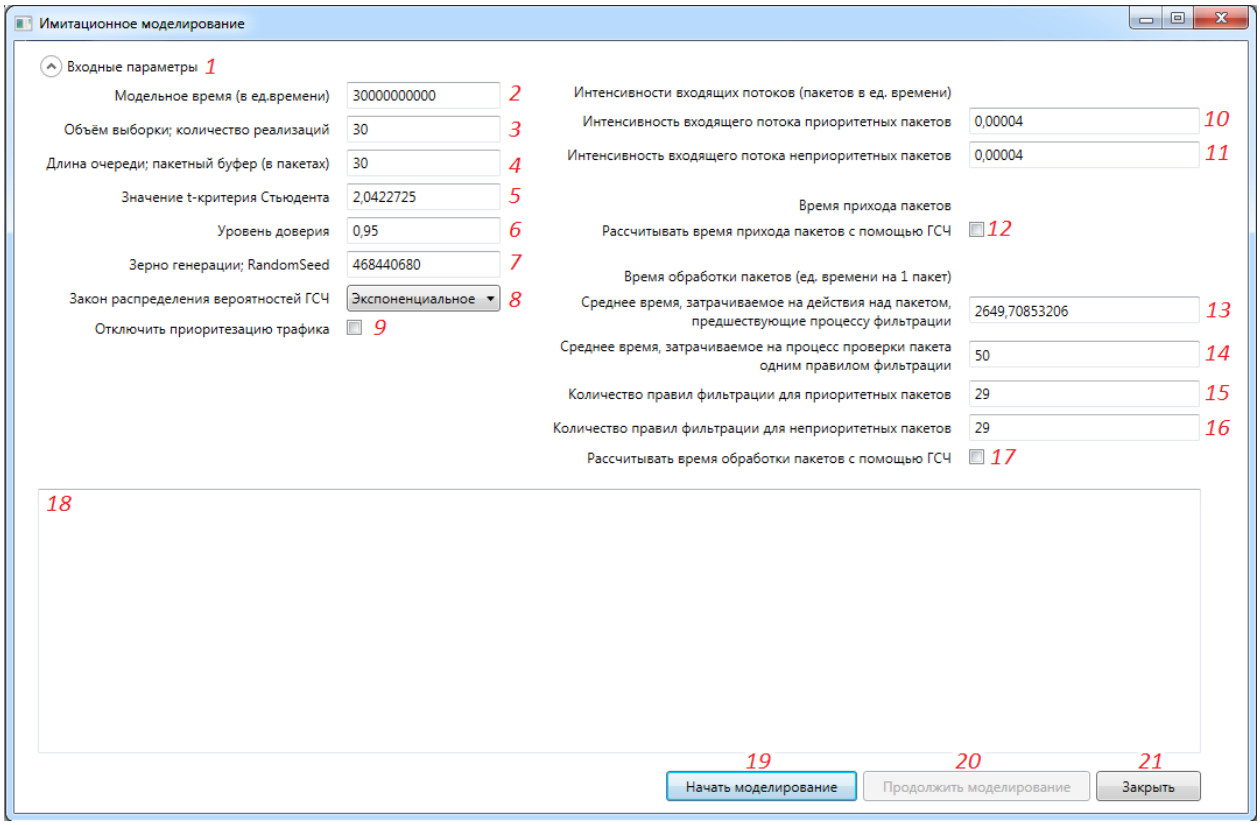


Рисунок Е.2 – Главное окно имитационной модели

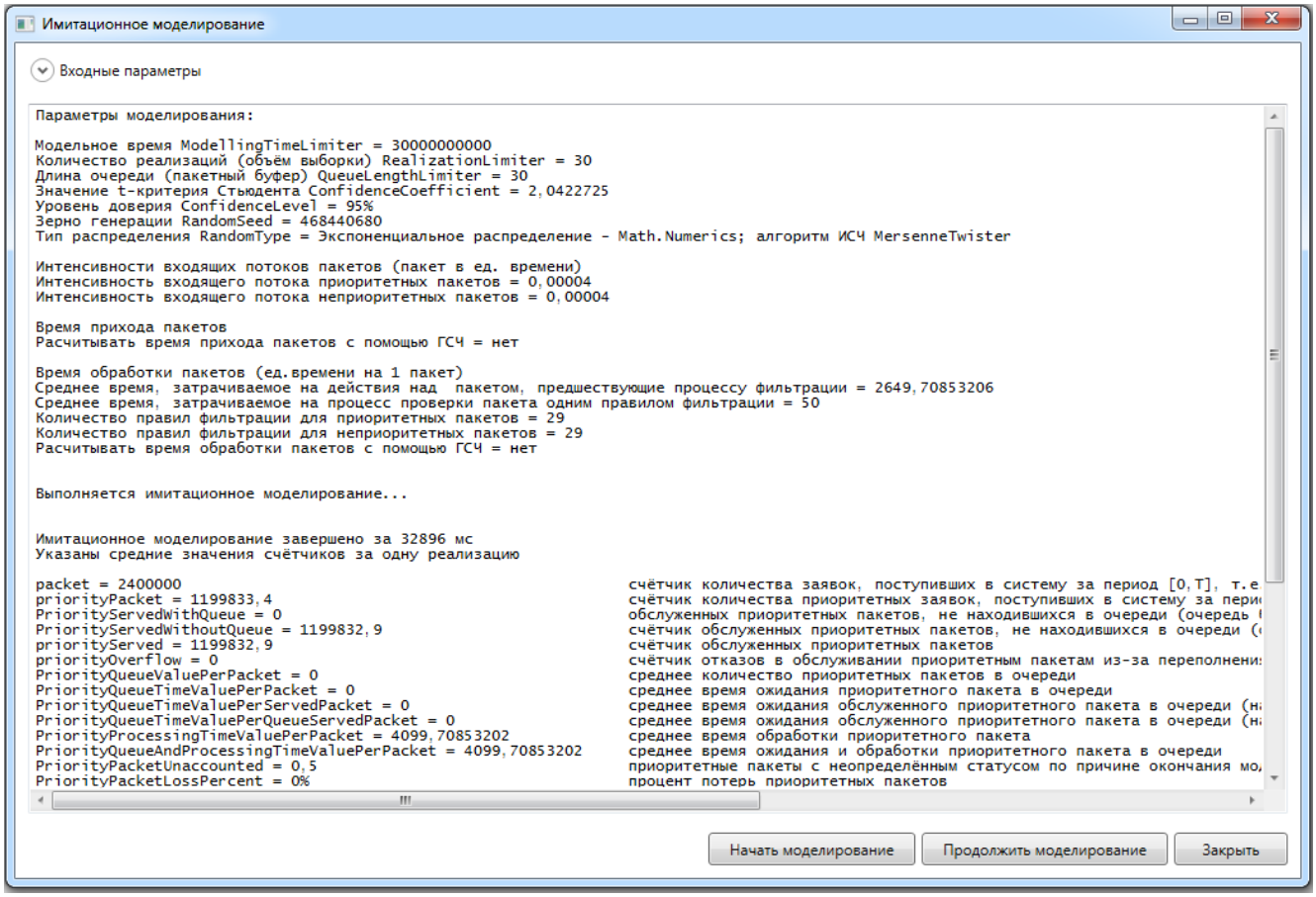


Рисунок Е.3 – Результат выполнения пункта 1.

Поле 2 – Модельное время (в ед. времени). В это поле осуществляется ввод модельного времени. Модельное время – это такое виртуальное время, в рамках которого происходят и обрабатываются все события моделирования, причем не обязательно пропорционально реальному времени, в котором развивается моделируемый процесс.

Поле 3 – Объём выборки; количество реализаций. В это поле осуществляется ввод количества реализаций, которые будут проведены при расчёте. В течении каждой реализации выполняется моделирование процесса функционирования системы в течении модельного времени (пункт 2).

Поле 4 – Длина очереди; пакетный буфер (в пакетах). В это поле осуществляется ввод количества пакетов, которые может вместить очередь (пакетный буфер модели).

Поле 5 – Значение t-критерия Стьюдента. В это поле осуществляется ввод табличного значения t-критерия Стьюдента, которое используется моделью, для расчёта доверительного интервала. По умолчанию значение соответствует 95% доверительному интервалу.

Поле 6 – Уровень доверия. В это поле осуществляется ввод уровень доверия, соответствующий выбранному и введённому ранее t-критерию Стьюдента. Данное поле имеет информационный характер, введённое значение попадает в вывод результатов и напоминает то, какой уровень доверия был выбран. Поле принимает цифровые значения без проверки, то есть возможно вводить уровень доверия в форматах 0,95 или 95 (как удобней).

Поле 7 – Зерно генерации; Random Seed. В это поле осуществляется ввод стартового числа для генераторов случайных чисел. Детерминированность используемых генераторов случайных чисел позволяет повторять результаты эксперимента, используя известное зерно генерации.

Поле 8 – Закон распределения вероятностей ГСЧ. Поле 8 представлено выпадающим меню с выбором из трёх пунктов:

1. Равномерное распределение – алгоритм закона распределения собственный; ИСЧ (источника случайных чисел) SystemRandom.

2. Экспоненциальное распределение – алгоритм закона распределения Math.Numerics; алгоритм ИСЧ MersenneTwister.

3. Экспоненциальное распределение – алгоритм закона распределения Math.Numerics; алгоритм ИСЧ SystemRandom.

Каждый пункт позволяет выбрать один из 3 способов генерации случайных чисел. Два типа законов распределения для генераторов случайных чисел: равномерный закон и экспоненциальный закон (показательный).

Реализация равновесного закона распределения выполнена автором диссертации самостоятельно. Реализация экспоненциального закона распределения заимствована в сторонней библиотеке Math.Numerics [131].

Для экспоненциального закона доступен выбор из двух источников случайных чисел, функционирующих по алгоритмам Вихря Мерсена (Mersenne Twister) [131] и по модифицированному алгоритму Donald. E Knuth’s, используемого в основе класса System.Random в C# с .Net Framework 1.1 [132, 133].

Поле 9 – Отключить приоритизацию трафика. Поле представлено «флажком». Если установить в нём галочку, то приоритетный поток пакетов потеряет свойство приоритетности над неприоритетным потоком, то есть они станут равноправными при обслуживании. Приоритетные пакеты не смогут выдавливать неприоритетные пакеты из очереди. Забор пакетов из очереди на обслуживание будет осуществляться в соответствии с дисциплиной FIFO.

При этом статистика по приходу и обслуживанию потоков пакетов по-прежнему будет собираться раздельно. ВНИМАНИЕ! Потоки пакетов не потеряют свои названия, «приоритетный» и «неприоритетный».

Поле 10 – Интенсивность входящего потока приоритетных пакетов. В это поле осуществляется ввод значения интенсивности поступающего потока приоритетных пакетов в единицу времени.

Поле 11 – Интенсивность входящего потока неприоритетных пакетов. В это поле осуществляется ввод значения интенсивности поступающего потока неприоритетных пакетов в единицу времени.

Поле 12 – Рассчитывать время прихода пакетов с помощью ГСЧ. Поле представлено «флажком». Если галочка в поле не установлена, то время межу приходом пакетов будет иметь постоянную величину, рассчитываемую на основе интенсивностей входящих потоков пакетов (поля 10, 11). Если установить в поле галочку, то время между приходом пакетов будет рассчитываться с использованием генератора случайных чисел, выбранного в поле 8. Математическое ожидание генератора случайных чисел рассчитывается на основе интенсивностей входящих потоков пакетов (поля 10, 11).

Поле 13 – Среднее время, затрачиваемое на действия над пакетом, предшествующее процессу фильтрации. В поле осуществляется ввод среднего времени, затрачиваемого на действия над пакетом, предшествующее процессу фильтрации. Время вводится в условных единицах времени.

Поле 14 – Среднее время, затрачиваемое на процесс проверки пакета одним правилом фильтрации. В поле осуществляется ввод среднего времени, затрачиваемого на процесс проверки пакета одним правилом фильтрации. Время вводится в условных единицах времени.

Поле 15 – Количество правил фильтрации для приоритетных пакетов. В поле осуществляется количества правил фильтрации для приоритетных пакетов. Количество правил будет умножено на среднее время, затрачиваемое на процесс проверки пакета одним правилом (поле 14) в результате чего будет получено среднее время, затрачиваемое на процесс фильтрации.

Если ГСЧ для расчёта времени обработки пакетов (поле 17) не включен, то это будет интерпретироваться как факт выхода пакета из базы правил именно с выбранного правила. Если ГСЧ для расчёта времени обработки пакетов (поле 17) включен, то это будет интерпретироваться, как математическое ожидание выбора правила, на котором происходит выход из базы правил.

Поле 16 – Количество правил фильтрации для неприоритетных пакетов. В поле осуществляется количества правил фильтрации для неприоритетных пакетов. Описание поля аналогично тому, что представлено для поля номер 15, но относится к неприоритетным пакетам.

Если количество правил для неприоритетных и приоритетных пакетов одинаковое, то математическое ожидание времени обработки для них будет одинаковым.

Поле 17 – Рассчитывать время обработки пакетов с помощью ГСЧ. Поле представлено «флажком». Если галочка в поле не установлена, то время обработки пакетов будет иметь постоянную величину, рассчитываемую на основе параметров обработки пакетов (поля 13-16). Если установить в поле галочку, то время обработки пакетов будет рассчитываться с использованием генератора случайных чисел, выбранного в поле 8. Математическое ожидание генератора случайных чисел рассчитывается на основе параметров обработки пакетов (поля 13-16).

Поле 18. В это поле осуществляется вывод результатов моделирования (пример на рисунке Е.3).

Поле 19 – начать моделирование. Поле представлено кнопкой. По нажатии кнопки производится отчистка внутренних переменных моделирования и запускается процесс моделирования в соответствии с введёнными сходными данными. Все поля ввода и кнопки за исключением кнопки 21 и кнопок окна windows в правом верхнем углу блокируются до окончания процесса моделирования.

Поле 20 – продолжить моделирование. Поле представлено кнопкой. При первом запуске модели кнопка неактивна и становится активной, после осуществления первого расчёта. При нажатии на эту кнопку конечное состояние очереди будет зафиксировано и перенесено в следующий расчёт, в том числе пакет, находящийся на обслуживании. То есть в новом расчёте, с новыми параметрами, очередь будет изначально заполнена пакетами, которые остались в системе с предыдущего расчёта со всеми своими параметрами. Пакет, который находился на обслуживании, будет продолжать своё обслуживание. В систему начнут приходить новые пакеты.

Кнопка становиться неактивной после старта моделирования.

Данная функция нужна для ввода системы в стационарный режим с начала модельного времени, либо для оценки временных характеристик МЭ при переходных процессах.

Поле 21 – закрыть. Поле представлено кнопкой. Действие кнопки аналогично стандартному крестику вверху окна Windows и выполняет закрытие программы, однако, если закрывать модель, с её использованием, то появиться дополнительное контекстное меню, которое потребует подтверждение закрытия программы.

В течение протекания процесса моделирования поле 21 изменяет свою функцию с закрытия имитационной модели на остановку процесса моделирования. При остановке модель произведёт краткий вывод результатов в поле 18, но не произведёт вывод в CSV-файл.

3.3. Проведение имитационного моделирования

Первый запуск. Для проведения имитационного моделирования изначально необходимо подготовить значения всех входных параметров (поля 2-17). Следующий шаг – определение размерности времени, которая будет использоваться в рамках расчёта (например, секунда, микросекунда и т.п). Затем необходимо привести все подготовленные значения к выбранной размерности времени и после этого ввести их в поля имитационной модели.

По окончании ввода входных параметров необходимо нажать кнопку 19 «Начать моделирование».

Все поля ввода и кнопки за исключением кнопки 21 и кнопок окна windows в правом верхнем углу блокируются до окончания процесса моделирования. В процессе моделирования в нижней части окна появится полоса прогресса, отражающая протекание процесса моделирования. Пример, представлен на рисунке Е.4

Рисунок Е.4 – Полоса прогресса процесса моделирования

Полоса делиться на количество реализаций, введённых пользователем в поле 3. Окончание каждой реализации отражается заполнением одного деления

полосы и в целом хорошо иллюстрирует протекание процесса моделирования. Время моделирования одной реализации почти постоянное, что позволяет пользователю легко ориентироваться в продолжительности расчёта.

По окончании расчёта имитационная модель произведёт вывод в поле 18 и в файл results.csv.

ВНИМАНИЕ! Файл results.csv не должен быть открыт в любой сторонней программе на момент окончания процесса моделирования иначе ПО моделирования не получит к нему доступ и не будет иметь возможность записать результаты моделирования.

После этого можно запустить моделирование заново с полным обнулением, что аналогично первому запуску или продолжить моделирование с сохранением состояния очереди.

Вторичный запуск процесса моделирования с обнулением состояния. При необходимости расчёта работы модели при других входных параметрах с обнулением состояния необходимо заменить входные параметры и запустить процесс моделирования кнопкой 19.

После запуска модель полностью отчистить промежуточную статистику, обнулит очередь и все записанные результаты, за исключением тех, что выведены в results.csv и начнёт процесс моделирование. Дальнейшее поведение системы аналогично первичному запуску.

Продолжение моделирования с сохранением состояния. При необходимости продолжения расчёта с сохранением состояния необходимо заменить входные параметры и продолжить процесс моделирования кнопкой 20.

После запуска модель создаст копию старой очереди, отчистит промежуточную статистику и все результаты, за исключением тех, что выведены в results.csv. Затем, используя информацию из скопированной очереди, обновит промежуточную статистику, восстановит состояние очереди и обслуживаемого пакета и продолжит процесс моделирования. Дальнейшее поведение системы аналогично первичному запуску.

Ввод системы в стационарное состояние. Для этого необходимо провести расчёт при любых исходных данных и оценить результаты. Если по результатам становится ясно, что модель вошла в стационарный режим, то необходимо не изменяя исходные данные продолжить процесс моделирования кнопкой 20

3.4. Получение результатов моделирования

Как упоминалось ранее, сокращённый вывод результатов осуществляется в главное окно программы. Расширенный вывод результатов осуществляется в отчуждаемый файл формата results.csv, с которым способны работать большинство табличных редакторов.

ВНИМАНИЕ! Файл results.csv не должен быть открыт в любой сторонней программе на момент окончания процесса моделирования иначе ПО моделирования не получит к нему доступ и не будет иметь возможность записать результаты моделирования.

Пример сокращённого вывода результатов приведён на рисунке Е.3.

Расширенный вывод результатов с использованием программы Microsoft Excel приведён на рисунках Е.5. Размер вывода достаточно большой, чтобы поместиться на лист формата А4, но в целом рисунок Е.5 даёт общее представление о выводе.

Для каждого расчёта в выводе предусмотрена демонстрация вводимых исходных данных и значения всех исследуемых показателей производительности, полученные при проведении каждой из реализаций. Замыкает расчёт средних значений, значений среднего отклонения и значений доверительного интервала по формулам Стьюдента.

ВНИМАНИЕ! Выводимые в файл results.csv результаты в качестве разделителя дробного и целого числа используют знак « , ». Будьте внимательный при работе с данными, при необходимости проводите замену знака на « . ».

3.5. Анализ результатов моделирования

Анализ результатов можно производить в табличных редакторах или с использованием другого ПО, которое предназначено для анализа подобных данных и обладает возможностями взаимодействия с данными в формате *.csv.

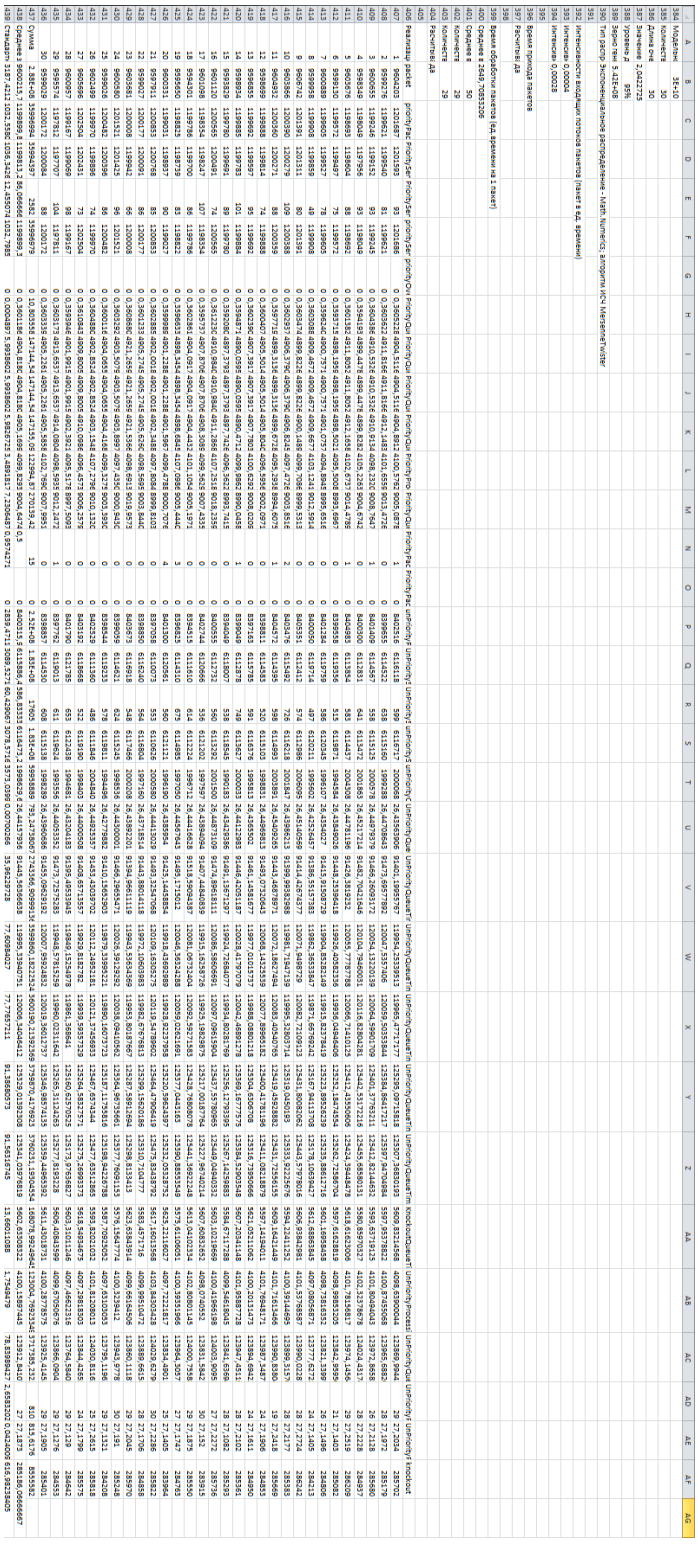


Рисунок Е.5 – Пример расширенного вывода результатов в файл result.csv

Приложение Ж.

Акты использования результатов диссертации

ЗАКРЫТОЕ АКЦИОНЕРНОЕ ОБЩЕСТВО
«НАУЧНО-ПРОИЗВОДСТВЕННОЕ ПРЕДПРИЯТИЕ
«БЕЗОПАСНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ»

105082, г. Москва, Большая Почтовая ул., 22
тел.: (495) 780-54-68
факс: (495) 783-55-01
e-mail: mail@npp-bit.ru
WEB: www.npp-bit.ru

Исх. № 1088 от «17» 12 2017 г.

На. № _____ от «___» _____ 201___ г.

АКТ

Об использовании результатов диссертационной работы Мусатова В.К. «Разработка метода оценки показателей производительности межсетевых экранов при функционировании в условиях приоритизации трафика» в проектной деятельности ЗАО «НПП «БИТ»

Результаты диссертационной работы Мусатова Владислава Константиновича (имитационная модель, инструкция по её использованию и теоретическое обоснование её функционирования) переданы в ЗАО «НПП «БИТ» по акту № 1 (вх. № 286 от 18.12.2017 г.)

ЗАО «НПП «БИТ», рассмотрев полученные результаты диссертации Мусатова В.К., определило потенциальную полезность представленных результатов в целях получения оценок производительности межсетевых экранов, используемых в рамках проектируемых ЗАО «НПП «БИТ» систем защиты информации.

Имитационной модель направлена в проектный отдел ЗАО «НПП «БИТ» для применения в процессе проектирования систем защиты информации.

Генеральный директор С.Н. Лапин

УТВЕРЖДАЮ
Проректор по учебной работе
МТУСИ, к.т.н., доцент
Е. В. Титов
«26» января 2018 г.

АКТ

об использовании результатов диссертационной работы В. К. Мусатова на тему: «Разработка метода оценки показателей производительности межсетевых экранов при функционировании в условиях приоритизации трафика» в учебном процессе кафедры СС и СК МТУСИ

Комиссия в составе директора Департамента организации и управления учебным процессом Н. Д. Карпушиной, заведующей Центром планирования и сопровождения учебного процесса Е. К. Патенченковой, заведующего кафедрой Сетей связи и систем коммутации д.т.н., проф. С. Н. Степанова, рассмотрев материалы диссертационной работы В. К. Мусатова, составила настоящий акт о том, что материалы диссертационной работы используются в учебном процессе кафедры СС и СК при проведении лекционных занятий по дисциплине «Сети связи».

Материалы для проведения учебного процесса представлены в виде лекции на тему «Анализ функционирования межсетевых экранов в сетях передачи данных».

Директор Департамента организации и управления учебным процессом **Н. Д. Карпушина**

Заведующая Центром планирования и сопровождения учебного процесса **Е. К. Патенченкова**

Заведующий кафедрой СС и СК, д. т. н, профессор **С. Н. Степанов**