Министерство науки и высшего образования Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
Институт прикладной математики и механики

Работа допущена к защите
Директор высшей школы
прикладной математики и
вычислительной физики
_____ Л.В. Уткин
«___»_____2020 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

РАЗРАБОТКА МЕТОДОВ АУТЕНТИФИКАЦИИ И ЗАЩИТЫ ДАННЫХ МОБИЛЬНЫХ АБОНЕНТОВ ДЛЯ ДОСТУПА К ОБЛАЧНЫМ СЕРВИСАМ

по направлению подготовки 02.04.01 – Математика и компьютерные науки

Направленность (профиль) 02.04.01_03 – Высокопроизводительные облачные вычисления и программное обеспечение роботов

Выполнил
студент гр. 3640201/80301 Д.А. Рыжов

Руководитель
доцент ВШПМиВФ ИПММ, к.т.н. Л.М. Курочкин

Консультант
доцент ВШПМиВФ ИПММ, к.т.н. В.В. Глазунов

Санкт-Петербург
2020

РЕФЕРАТ

На 109 с., 51 рисунок, 5 таблиц, 7 приложений.

КЛЮЧЕВЫЕ СЛОВА: IT-ТЕХНОЛОГИИ, АУТЕНТИФИКАЦИЯ, WEB-СЕРВЕР, ЗАЩИТА ИНФОРМАЦИИ, SSL/TLS, МОБИЛЬНЫЙ АБОНЕНТ, ПЕРЕДАЧА ДАННЫХ, IPSEC, VPN, ШИФРОВАНИЕ

Тема выпускной квалификационной работы: «Разработка методов аутентификации и защиты данных мобильных абонентов для доступа к облачным сервисам».

Данная работа посвящена исследованию особенностей процесса обмена данными в глобальной сети между мобильным абонентом и веб-сервисом по защищенному каналу связи и разработке методов аутентификации, позволяющих добиться высоких показателей производительности и безопасности в данном процессе при различных условиях.

Задачи, которые решались в ходе исследования:

1. Изучение актуальных алгоритмов шифрования, криптографических хеш-функций, сервисов защищенного канала, обеспечивающих высокий уровень безопасности информации при обмене данными между абонентами в глобальной сети.
2. Разработка классификаций мобильных абонентов — пользователей веб-сервисов и передаваемых данных.
3. Разработка модели оценки качества защищенного канала передачи данных.
4. Разработка сценариев — наборов характеристик мобильных абонентов и передаваемых данных, определяющих условия обмена данными.
5. Анализ актуальных данных о криптостойкости алгоритмов шифрования, криптографических хеш-функций, сервисов защищенного канала обмена данными и определение набора методов аутентификации, обеспечивающих максимальный уровень защищенности информации при умеренных требованиях к производительности системы абонента.

6. Разработка имитационной модели защищенного канала обмена данными между мобильным абонентом и облачным сервисом.

7. Реализация на модели набора методов аутентификации.

8. Исследование эффективности применения методов аутентификации в различных сценариях подключения мобильных абонентов к облачным сервисам.

Имитационная модель разработана на основе реальной сети с системой маршрутизаторов MikroTik, работающих под управлением операционной системы RouterOS. С помощью экспериментов на данной модели были проведены измерения характеристик защищенного канала передачи данных и по их результатам рассчитаны значения параметров, определяющих его качество.

В результате проведенной работы получен набор методов аутентификации, актуальных для решения проблем безопасности информации и качества ее обмена в глобальной сети при условии подключения мобильного абонента к облачному сервису. Разработана таблицы решений, определяющие целесообразность использования конкретного метода аутентификации при определенных условиях обмена данными.

THE ABSTRACT

109 pages, 51 pictures, 5 tables, 7 applications

KEYWORDS: IT TECHNOLOGIES, AUTHENTICATION, WEB SERVER, INFORMATION SECURITY, SSL/TLS, MOBILE USER, DATA TRANSFER, IPSEC, VPN, ENCRYPTION

The Theme of the final graduation research: “The Development of authentication methods and protection of data of mobile subscribers for access to cloud services”.

This research is devoted to study the features of the data exchange process in a global network between a mobile subscriber and a web service through a SVTC and the development of authentication methods which are able to achieve high productivity and security in this process under different conditions.

The Tasks that were solved during the research:

1. The research of current encryption algorithms, cryptographic hash functions, the services of secure channel providing a high level of information security during data exchange between subscribers in the global network.
2. Development of classifications of mobile subscribers - users of web services and transmitted data.
3. Development of a model to estimate the quality of a secure data channel.
4. Development of scenarios - sets of characteristics of mobile subscribers and transmitted data determination the conditions for data exchange.
5. Analysis of current data on the cryptographic strength of encryption algorithms, cryptographic hash functions, services of a secure data exchange channel and determination of a set of authentication methods providing the highest level of information security with moderate requirements for subscriber system performance.
6. Development of a simulation model of a secure data exchange channel between a mobile subscriber and a cloud service.
7. Implementation on a model of a set of authentication methods.
8. A research of the effectiveness of implementing authentication methods in various scenarios of connected mobile subscribers to cloud services.

The imitation model is developed on the basis of a real network with a system of MikroTik routers working under the RouterOS operating system. Using experiments on this model, measures of the characteristics of a secure data transmission channel were carried out and the values of parameters determining its quality were calculated based on their results.

As a result of the work, a set of authentication methods was obtained that are relevant to solve the problems of information security and the quality of its exchange in the global network, provided that the mobile subscriber is connected to the cloud service. A decision table has been developed that determines the usefulness of using a specific authentication method under certain conditions of data exchange.

СОДЕРЖАНИЕ

Введение

Актуальность исследования. На сегодняшний день повсеместное распространение беспроводных технологий передачи данных привело к стремительному росту числа мобильных абонентов - пользователей веб-сервисов в сети Интернет. Такие абоненты по всему миру генерируют большое количество веб-трафика, больше, чем все прочие устройства суммарно. Однако, существуют уязвимости, которые могут представлять угрозу для данных таких пользователей. К таким уязвимостям относится регулярный процесс смены IP-адреса абонента, вызванный его переходом в другую сеть. Такой переход может происходить как при смене активного физического интерфейса подключения к сети, так и в рамках одного интерфейса. Также поведение мобильного абонента в сети Интернет характеризуется повышенным количеством разрывов сети в условиях недостаточного уровня сигнала, в которых некоторые абоненты находятся довольно часто.

Данные проблемы являются следствием перемещений абонента в пространстве. Абонент, успешно прошедший аутентификацию, «перемещается» из одной сети в другую, при этом время жизни следующей сети, в которой он будет находиться может быть сильно ограничено и представлять собой значение, меньшее, чем время повторной аутентификации на сервере. Это оказывает негативное влияние на качество передачи данных. При этом также большое внимание следует уделять вопросам защиты данных.

Для того чтобы обеспечить пользователям полную защиту данных при использовании веб-ресурсов необходимо гарантировать, что каналы доступа защищены должным образом. Необходимо достигнуть максимальной конфиденциальности обмена «пользователь — информационный ресурс». Главной задачей является защита сетевого трафика от злоумышленника, мотивами которого являются перехват данных или их подмена. Единственным решением является использование надежных протоколов, обеспечивающих защиту данных.

Если участниками сетевого взаимодействия являются веб-сервер и мобильный абонент, то между ними реализовать устойчивый защищенный канал передачи данных является сложной задачей. Необходимо выяснить, как нестабильное поведение мобильного абонента в глобальной сети будет влиять на качество передачи данных и насколько высоки риски эти данные утратить.

Используя различные методы аутентификации и проведя соответствующие эксперименты, можно прийти к выводам, при каких сценариях тот или иной метод будет показывать себя лучше. Результат, к которому необходимо стремиться — минимизировать время повторной аутентификации абонента, обеспечить надежную защиту данных, при этом поддерживать стабильно высокую скорость обмена данными при минимальном потреблении ресурсов системы.

Целью данной работы является разработка методов аутентификации, позволяющих уменьшить потери сетевых пакетов и обеспечивающих высокую степень защищенности данных мобильных абонентов при обращении к облачному сервису в условиях динамической смены сетей.

Для достижения поставленной цели должны быть решены следующие **задачи**:

1. Изучение актуальных алгоритмов шифрования, криптографических хеш-функций, сервисов защищенного канала, обеспечивающих высокий уровень безопасности информации при обмене данными между абонентами в глобальной сети.
2. Разработка классификаций мобильных абонентов — пользователей веб-сервисов и передаваемых данных.
3. Разработка модели оценки качества защищенного канала передачи данных.
4. Разработка сценариев — наборов характеристик мобильных абонентов и передаваемых данных, определяющих условия обмена данными.
5. Анализ актуальных данных о криптостойкости алгоритмов шифрования, криптографических хеш-функций, сервисов защищенного канала

обмена данными и определение набора методов аутентификации, обеспечивающих максимальный уровень защищенности информации при умеренных требованиях к производительности системы абонента.

6. Разработка имитационной модели защищенного канала обмена данными между мобильным абонентом и облачным сервисом.

7. Реализация на модели набора методов аутентификации.

8. Исследование эффективности применения методов аутентификации в различных сценариях подключения мобильных абонентов к облачным сервисам.

Объектом исследования являются вычислительные системы и сетевое программное обеспечение, позволяющее организовать защищенный канал связи между мобильным абонентом и облачным сервисом.

Предметом исследования являются свойства алгоритмов и протоколов, обеспечивающих конфиденциальность, целостность и аутентичность данных в защищенном канале связи между мобильным абонентом и облачным сервисом.

Методы исследования. В ходе исследования использовались теоретические и экспериментальные методы. К теоретическим методам относится изучение актуальных данных о криптосистемах и их последующий анализ, к экспериментальным — метод имитационного моделирования системы обмена данными, метод модельных экспериментов.

Глава 1. Обзор методов аутентификации в защищенных соединениях

1.1 Обзор актуальных алгоритмов шифрования

При взаимодействии участников процесса обмена данными в глобальной сети между собой, весь трафик от источника до пункта назначения проходит через множество сетевых устройств, которые администрируют посторонние организации. Данный факт может вызывать опасения, так как существует вероятность, что на каком-либо из хостов, содержимое пакетов будет скомпрометировано или подвержено изменениям [2]. Это создает серьезные проблемы, тем более существенные, чем выше уровень значимости или конфиденциальности передаваемых данных. Для решения таких проблем и применяются защищенные каналы связи. Такие каналы можно представить как туннель, в котором информация помещается с одной стороны и распознать ее можно только с другой [4]. Информация модифицируется таким образом, чтобы ее невозможно было изменить или просмотреть на пути их следования. Такая модификация называется шифрованием.

Шифрование — это обратимое преобразование данных с целью их защиты от несанкционированного доступа. Данное преобразование обеспечивает три важных состояния защищенности информации:

– Конфиденциальность. Шифрование необходимо для сокрытия информации при обмене данными;

– Целостность. Шифрование необходимо для исключения угрозы изменения информации в процессе ее передачи или хранения;

– Аутентичность. Шифрование необходимо для проверки подлинности источника информации.

В криптографии с помощью алгоритмов шифрования происходит процесс генерации ключей в виде последовательности символов, которая используется для кодирования фрагментов информации и дальнейшего обратного преобразования. Способ использования данных ключей указывает на различия симметричного и асимметричного методов шифрования.

В то время как симметричные криптосистемы используют один ключ для выполнения такой функции, асимметричные используют два — один из ключей для шифрования данных, а другой для обратного преобразования [5]. В асимметричных криптосистемах ключ, использующийся для шифрования, также известный как публичный, может открыто передаваться другим пользователям. Ключ, используемый для дешифрования, является приватным и обязательно должен храниться в секрете.

У данных видов шифрования существуют как преимущества, так и недостатки. Симметричные криптосистемы работают намного быстрее и не требуют высокой вычислительной мощности, но их главным недостатком является проблема распределения ключей. Вследствие того факта, что один и тот же ключ должен использоваться для шифрования и расшифровки данных, данный ключ необходимо передать всем пользователям, которым потребуется доступ к этим данным. Это создает определенные риски.

Асимметричные криптосистемы решают эту проблему, используя разные ключи для шифрования и дешифрования. Компромисс решений заключается в том, что асимметричные криптосистемы менее производительны по сравнению с симметричными и требуют большей вычислительной мощности из-за своей длины ключа.

Во многих приложениях используется гибридная система, включающая в себя симметричное и асимметричное шифрование. Примером таких систем являются криптографические протоколы семейства SSL/TLS, разработанные для обеспечения безопасной передачи данных в глобальной сети.

AES (Advanced Encryption Standard) — симметричный итеративный блоковый алгоритм. Данный алгоритм может поддерживать любую комбинацию данных (128 бит) и длину ключа 128, 192 и 256 бит. В процессе обратного преобразования система проходит 10 раундов для 128-битных ключей, 12 раундов для 192-битных ключей и 14 раундов для 256-битных ключей для получения окончательного зашифрованного текста или получения исходного текста [6]. Алгоритм представляет каждый блок информации в виде двумерного

байтного массива размерностью 4×4, 4×6 или 4×8 в зависимости от установленной длины. Затем на соответствующих этапах проводится процесс преобразования над независимыми столбцами, строками или над отдельными байтами. Структура алгоритма AES показана на рисунке 1.1.

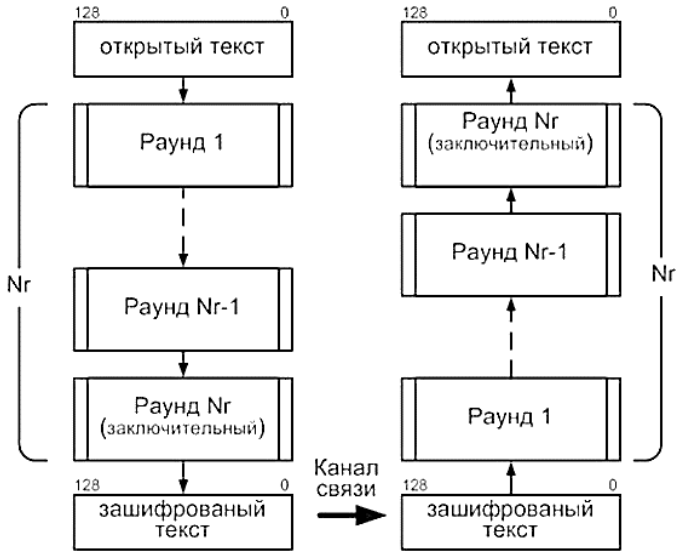

Рисунок 1.1 — Структура алгоритма AES

Blowfish — 64-битный блочный шифр с ключом переменной длины, от 32 до 448 бит. Данный алгоритм состоит из двух частей: расширения ключей и шифрования данных. Расширение ключа преобразует исходный ключ в некоторое количество суммированных массивов подключей [7]. Шифрование информации происходит в 16-итерационной сети Фейстеля. Каждая итерация состоит из зависимой перестановки, зависящей от ключа и данных замены. Основные используемые операции — XOR и сложение на 32-битных словах. Также присутствуют дополнительные операции — 4 поиска в индексированных массивах на одну итерацию. Структура алгоритма показана на рисунке 1.2.

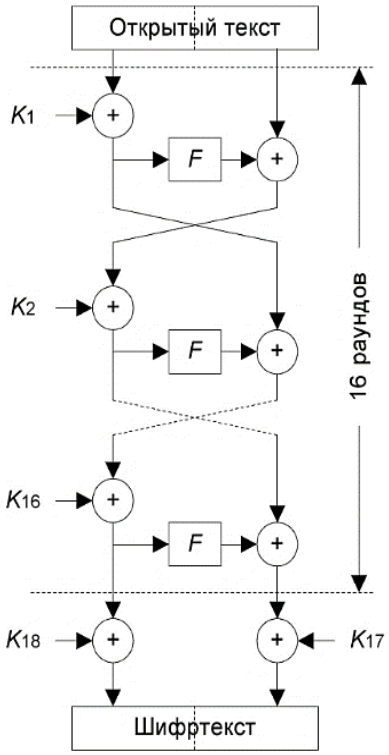

Рисунок 1.2 — Структура алгоритма Blowfish

Twofish — симметричная криптосистема с размером блока 128 бит и длиной ключа до 256 бит. Число раундов в такой системе — 16. Алгоритм является преемником алгоритма Blowfish.

Разработчики алгоритма Twofish оставили удачные решения, используемые в предыдущем проекте Blowfish, помимо этого, они провели тщательные исследования, посвященные перемешиванию данных в сети Фейстеля. Алгоритм Twofish представляет собой сеть Фейстеля смешанного типа: первые две ветви на нечетных раундах осуществляют модификацию третьей и четвертой, на четных раундах данная ситуация становится противоположной [8]. В алгоритме Twofish используется криптографическое преобразование Адамара — это обратимое арифметическое сложение первого потока со вторым, второго с первым. Структура алгоритма показана на рисунке 1.3.

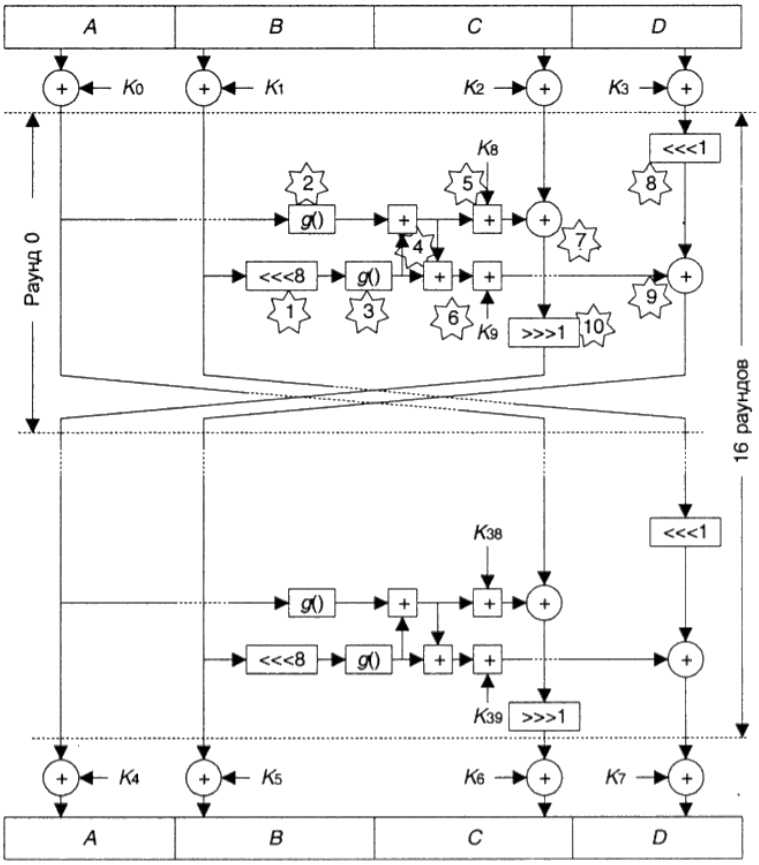

Рисунок 1.3 — Структура алгоритма Twofish

Camellia — это 128-битная блочная симметричная криптосистема, оперирующая ключами шифрования, которые составляют длину 128, 192 и 256 бит. В основе данной криптографической системы лежит сеть Фейстеля.

В зависимости от дины ключа в алгоритме предусмотрено различное количество раундов: 18 раундов для 128-битного ключа шифрования и 24 раунда для всех остальных вариантов длины ключа шифрования.

Перед первым раундом преобразования происходит входное отбеливание данных – на блок данных с помощью операции XOR накладывается фрагмент расширенного ключа, составляющий 128-бит [9]. На следующем этапе 128-битный блок данных разделяется на субблоки, составляющие 64 бита. Данные субблоки проходят раунды шифрования. В каждом шестом раунде левый субблок подвергается обработке функцией FL, правый – функцией FLI. После завершения раундов преобразования полученные субблоки меняются местами. Следующий этап — выходное отбеливание данных. На данном этапе

используется следующий 128-битный фрагмент расширенного ключа. При обратном преобразовании данных фрагменты расширенного ключа используются в обратном порядке. Полная структура алгоритма шифрования Camellia показана на рисунке 1.4.

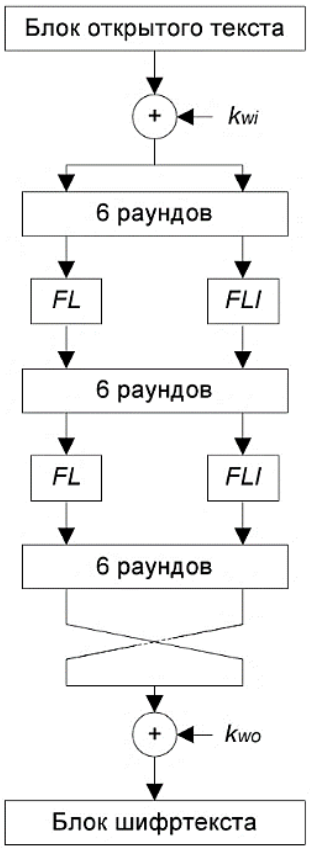

Рисунок 1.4 — Структура алгоритма Camellia

В основе криптосистемы **ChaCha20** лежит конструкция блочного шифра.

Операции, которые входят в шифр ChaCha20, можно разбить на три части. Первая часть — функция для "четверть-раунда" [10]. В раундах преобразования эта функция преобразует четыре переменные состояния из общих шестнадцати за один вызов, в каждом раунде такая функция используется восемь раз: четыре раза в каждой конфигурации.

Вторая часть — это функция преобразования состояния: она построена на последовательном изменении состояния шифра благодаря использованию QR-функции. Первое состояние формируется на основе ключа, вектора

инициализации и счётчика, результатом будет 64-байтный блок ключевого потока данных.

Третья часть — это функция шифрования: происходит вызов функции шифрования с заданным ключом и nonce с последовательным увеличением значения счетчика, вычисляемый ключевой поток суммируется с открытым текстом при помощи операции XOR. На вход поступают следующие данные: ключ длиной 256 бита, начальное значение счетчика, поток открытых данных. На выходе функции — зашифрованные данные. Вследствие того, что данный шифр представляет собой вариацию счетчика с гаммированием, обратное преобразование производится аналогичным образом. Структура алгоритма показана на рисунке 1.5.

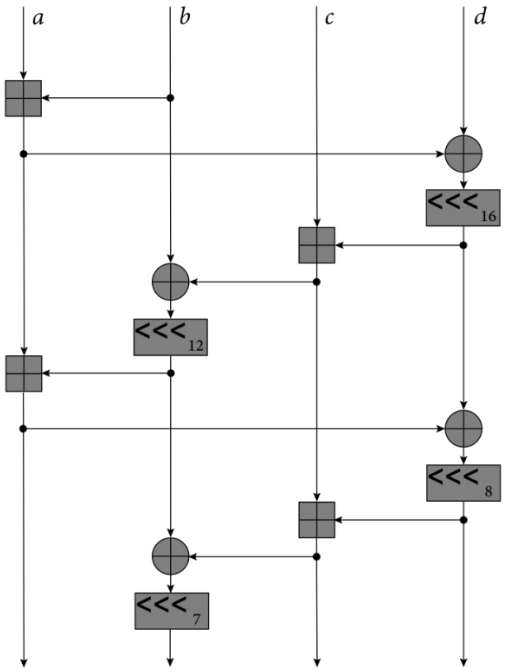

Рисунок 1.5 — Структура алгоритма CHACHA

RSA (Rivest, Shamir, Adleman) — криптографическая система с открытым ключом, которая основывается на вычислительной сложности решения задачи факторизации больших целых чисел [12]. Алгоритм включает в себя четыре этапа: этап генерация ключей, этап передачи ключей, этап шифрования и обратного преобразования. Для процесса шифрования в данном алгоритме

используется операция возведения в степень по модулю большого числа. Для обратного преобразования необходимо вычислить функцию Эйлера от этого большого числа. Структура алгоритма RSA показана на рисунке 1.6

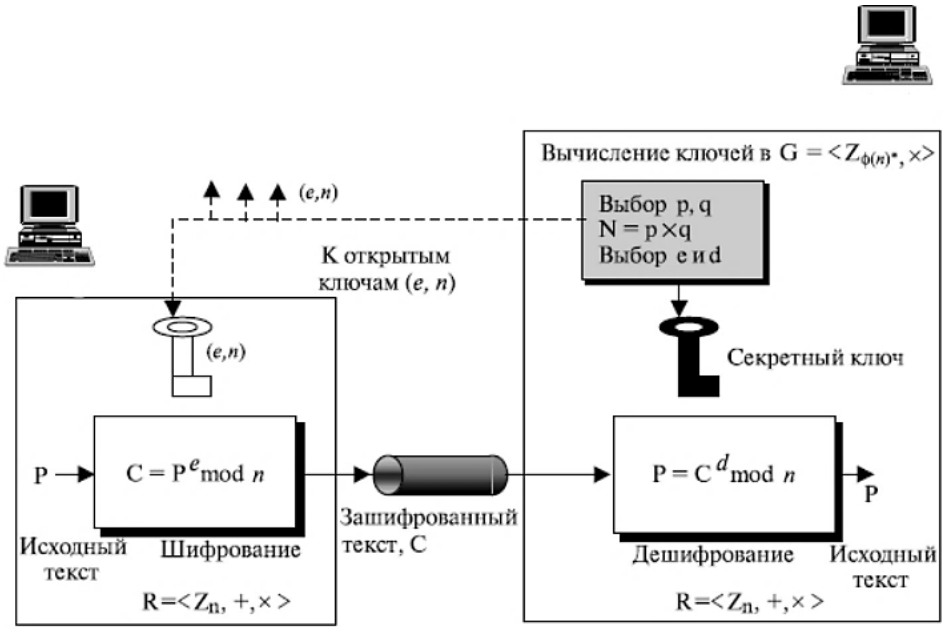

Рисунок 1.6 — Структура алгоритма RSA

DH (Diffie-Hellman) — это метод защищенного обмена криптографическими ключами по незащищенному каналу. Его защищенность состоит в сложности решения задачи вычисления дискретных логарифмов в конечном поле, в отличие от решения задачи дискретного возведения в степень. Решение проблемы дискретного логарифмирования позволит взломать алгоритм, но обратное утверждение до сих является открытым вопросом, то есть эквивалентность данных проблем не доказана. Структура алгоритма показана на рисунке 1.7.

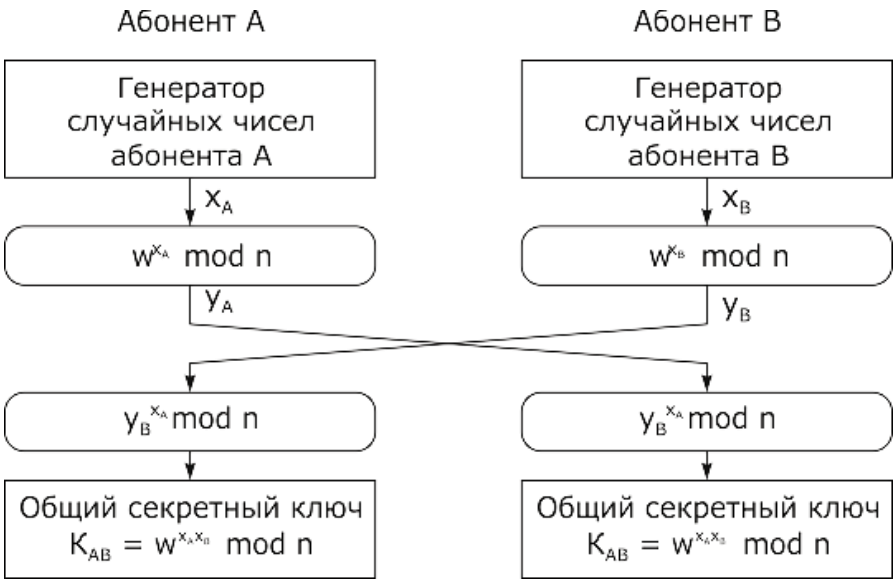

Рисунок 1.7 — Структура алгоритма DH

DSA (Digital Signature Algorithm) — криптографическая система с открытом ключом, предназначенная для создания электронной цифровой подписи. Данная система также основана на вычислительной сложности решения логарифмов в конечных полях.

Алгоритм DSA включает в себя две системы: алгоритм создания подписи сообщения и алгоритм проверки.

Оба алгоритма вычисляют хеш данных с помощью криптографической хеш-функции. Алгоритм подписи для ее создания использует полученный хеш и секретный ключ, алгоритм проверки использует хеш, цифровую подпись и открытый ключ для ее проверки.

Фактически происходит подписывание не сообщения произвольной длины, а его хеша длиной от 160 до 256 бит, поэтому в данной системе возможно появление коллизий, в результате одна и та же подпись может быть действительна для нескольких сообщений с идентичными результатами хешей. Поэтому необходимо выбрать достаточно сильную хеш-функцию, данная задача очень важна для всей системы в целом. В первой версии DSA использовалась хеш-функция SHA-1, в последней версии возможно использование любого алгоритм семейства SHA-2.

Для работы криптографической системы необходимо наличие базы соответствия между реальными реквизитами автора и открытыми ключами, а также всеми параметрами схемы цифровой. Подобной базой может служить, например, центр сертификации. На рисунке 1.8 показана структура алгоритма DSA.

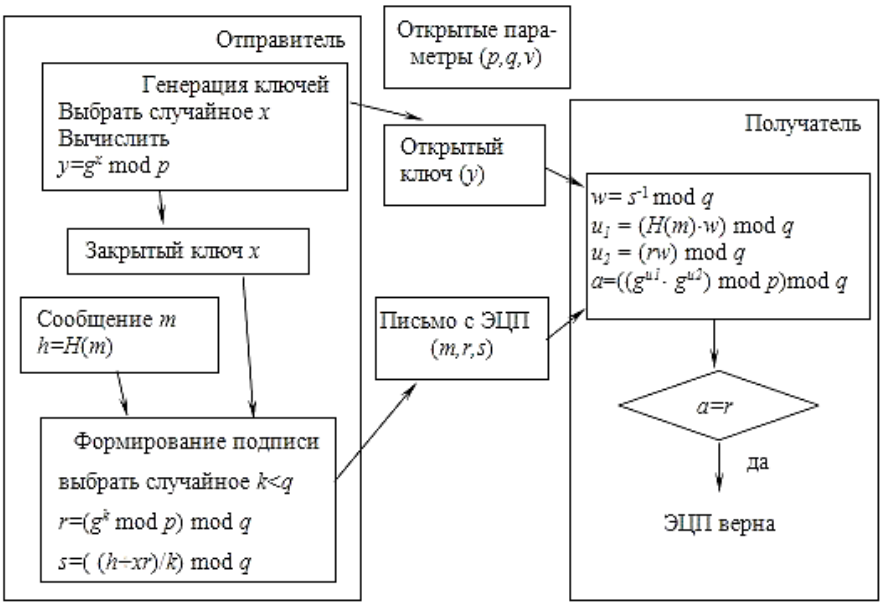

Рисунок 1.8 — Структура алгоритма DSA

Эллиптическая криптография — раздел криптографии, изучающий асимметричные криптосистемы, которые основаны на эллиптических кривых над конечными полями. Главное преимущество такого вида криптографии заключается в том, что на данный момент не существует субэкспоненциальных алгоритмов решения задачи дискретного логарифмирования.

В большинстве криптосистем современной криптографии возможно использование эллиптических кривых. Идея заключается в том, что алгоритм, который используется для конкретных конечных групп, модифицируется для использования групп рациональных точек эллиптических кривых.

ECDH (Elliptic curve Diffie-Hellman) — это вариация протокола DH с использованием эллиптической криптографии.

ECDSA (Elliptic Curve Digital Signature Algorithm) — это вариация протокола DSA с использованием эллиптической криптографии.

1.2 Обзор актуальных криптографических хеш-функций

В основе обеспечения целостности и аутентификации информации лежит отдельный вид шифрования — шифрование с помощью вычислительно необратимой функции. Частным случаем необратимой функции является хэш-функция.

Хеш-функция - функция, преобразующая массив входных данных произвольной длины в битовую строку установленной длины, выполняемое определенным алгоритмом.

Один из вариантов использования хэш-функции для организации защищенного соединения состоит в добавлении определенным образом секретного значения к данным, которые подаются на вход такой хэш-функции. Алгоритм носит название HMAC, описан в стандарте RFC 2104. Данный алгоритм является обязательным при реализации MAC для организации защищенного соединения.

На сегодняшний день подавляющую долю применений хеш-функций составляют алгоритмы семейства SHA.

SHA-1 (Secure Hash Algorithm 1) —криптографический алгоритм хеширования, который генерирует 160-битное уникальное значение из входных данных. Стандарт описан в RFC 3174. Он используется во многих криптографических протоколах и приложениях. Принципы, положенные в основу алгоритма SHA-1, являются аналогичными тем, которые были использованы при проектировании предшествующего протокола MD5.

Алгоритм SHA-1 реализует хеш-функцию, которая построена на идее функции сжатия. Данные, поступающие на вход функции сжатия —512-битный блок сообщения и выход с предыдущего блока. Выход представляет собой значение всех хеш-блоков до этого момента. Значением хеша всего сообщения является выход самого последнего блока. Структура алгоритма SHA-1 показана на рисунке 1.9.

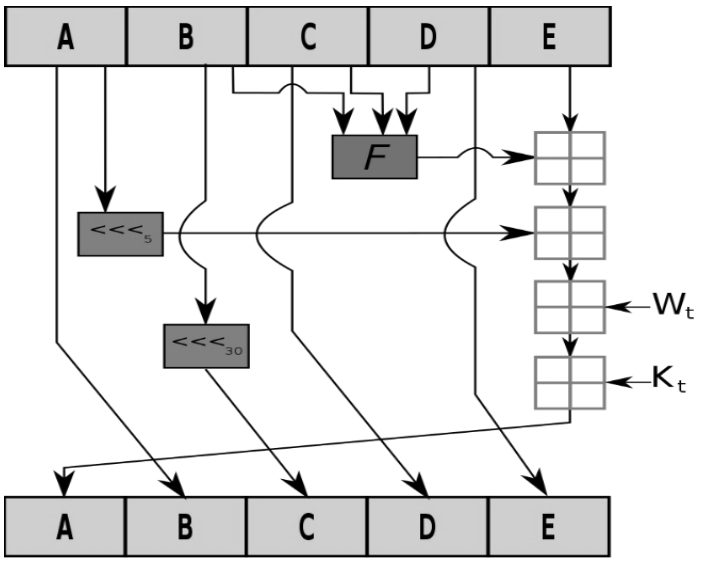

Рисунок 1.9 — Структура алгоритма DSA

Алгоритм SHA-1 содержит больше раундов, чем MD5 (80 вместо 64) и выполняется на буфере длиной 160 бит по сравнению со 128-битным буфером предшественника. Следовательно, алгоритм SHA-1 работает приблизительно на 25% медленнее, чем алгоритм MD5 на идентичной платформе.

SHA-2 (Secure Hash Algorithm 2) — семейство хеш-функций, которое включает в себя алгоритмы SHA-224, SHA-256, SHA-384, SHA-512. Данные системы построены на основе SHA-1.

На первом этапе преобразования исходное сообщение после дополнения раскладывается на блоки, а затем каждый блок раскладывается на 8 слов. Система пропускает каждый блок через цикл с 64 или 80 итерациями. На каждой из этих итераций 2 слова из 8 подвергаются процессу преобразования, функцию задают остальные слова. В итоге результаты обработки каждого блока суммируются, и полученная сумма будет являться значением хеш-функции. Структура алгоритма SHA-2 показана на рисунке 1.10.

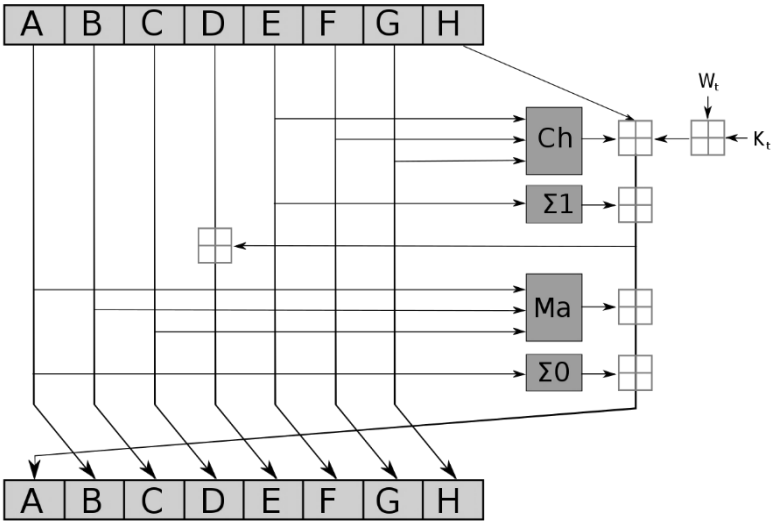

Рисунок 1.10 — Структура алгоритма SHA-2

Алгоритмы SHA-256 и SHA-512 являются хэш-функциями, которые вычисляются с 32-разрядными и 64-разрядными словами соответственно. Данные алгоритмы используют различное количество сдвигов и аддитивные константы, в остальном их структуры практически идентичны, ключевая разница содержится в количестве раундов.

SHA-224 и SHA-384 являются усеченными версиями SHA-256 и SHA-512 соответственно. Основное отличие между ними состоит в том, что данные хеш-функции вычисляются с различными начальными значениями. Их использование целесообразно, например, для обеспечения совместимости с устаревшими аппаратными вычислительными системами.

1.3 Сервисы защищенного канала

Задача защиты информации в процессе передачи по открытым каналам связи основана на организации защищенных каналов связи — криптозащищенных туннелей. Такой туннель представляет собой соединение, проведенное в открытой сети, в рамках которого происходит передача криптографически защищенных пакетов сообщений.

Защищенный канал может быть организован с помощью системных средств, которые реализованы на разных уровнях модели OSI. Перечень основных сервисов приведен в таблице 1.

Таблица 1 — Иерархия сервисов защищенного канала на модели OSI.

Уровни защищаемых протоколов	**Протоколы защищенного канала**	**Свойства протоколов защищенного канала**
Прикладной	S/MIME, PGP	Непрозрачность для приложений, независимость от транспортной инфраструктуры
Представления		
Сеансовый	SSL/TLS (HTTPS, SSTP, OpenVPN)	
Транспортный		
Сетевой	IPsec	Прозрачность для приложений, зависимость от транспортной инфраструктуры
Канальный	PPTP, L2TP	
Физический		

Защищенный канал, организованный на **прикладном** уровне, может обеспечить защиту определенной сетевой службы, например файловой, почтовой или гипертекстовой. Например, сетевой протокол S/MIME способен защитить исключительно сообщения электронной почты. При данном подходе каждая служба нуждается в разработке собственной, криптографически защищенной версии протокола.

Применительно к веб, обмен данными в глобальной сети осуществляется с помощью протокола HTTP, который является протоколом прикладного уровня передачи данных, однако в нем не предусмотрены механизмы защиты информации. Не существует сервисов прикладного уровня модели открытых систем OSI, обеспечивающих безопасность протокола HTTP, поэтому данный уровень рассматриваться не будет.

Протоколы **сеансового** уровня модели OSI выполняют функцию установки логических соединений и совершают управление над ними. Существует возможность использования на данном уровне различных программ-посредников, которые проверяют допустимость запрошенных

соединений, и которые могут обеспечивать выполнение других важных функций безопасности межсетевого взаимодействия.

Протоколы, позволяющие организовать защищенный виртуальный канал на сеансовом уровне, являются прозрачными для вышестоящих протоколов защиты и высокоуровневых протоколов предоставления различных сервисов. Однако, на данном уровне существует зависимость от приложений, которые реализуют высокоуровневые протоколы. Поэтому при реализации протоколов защиты, соответствующих этому уровню, потребуется внести изменений в высокоуровневые сетевые приложения.

Чем ниже по уровню [31] модели OSI расположены сервисы организации защищенного канала связи, тем более они прозрачны для приложений и других прикладных протоколов. На уровнях **сетевом** и **канальном,** зависимость приложений от протоколов защиты данных практически исчезает, но в таком случае возникает другая проблема — зависимость протокола защиты данных от используемой сетевой технологии.

1.4 Протоколы формирования защищенных каналов на канальном уровне.

Средства организации **VPN (Virtual Private Network)**, которые используются на канальном уровне модели открытых систем OSI, могут обеспечить инкапсуляцию трафика различных видов третьего и более высоких уровней и способны построить виртуальные защищенные туннели типа точка-точка. Достоинством организации защищенного канала на данном уровне является гарантированная прозрачность для приложений прикладного уровня и служб транспортного и сетевого уровня. Шифрованию на данном уровне подлежат как данные, так и IP-адреса сторон. К группе сервисов защитного соединения канального уровня относятся VPN-протоколы PPTP и L2TP.

Протоколы PPTP и L2TP основаны на двухточечном протоколе PPP и по сути являются его расширениями. Протокол PPP был разработан для решения задачи инкапсуляции данных и их транспортировки по каналам типа точка-точка. Протокол PPP также служит для организации асинхронных соединений.

Для доставки защищенных данных от абонента к серверу через сети общего пользования на первом этапе производится инкапсуляция данных на базе протокола PPP, затем PPTP и L2TP выполняют преобразование данных и собственную инкапсуляцию. После того как туннельный протокол доставляет пакеты из начальной точки туннеля в конечную, выполняется деинкапсуляция.

Работа протокола **PPTP (Point-to-Point Tunneling Protocol)** заключается в инкапсуляции кадров PPP в датаграммы IP для передачи их по сети. Данный протокол использует соединение TCP для управления защищенным туннелем и модифицированную версию протокола GRE для инкапсуляции PPP-кадров для защищенных данных. Полезные данные кадров PPP могут быть зашифрованы или сжаты.

Кадр PPP заключается в оболочку, включающую заголовок GRE и заголовок IP. В заголовке IP-адреса источника информации и получается соответствуют VPN-клиенту и VPN-серверу. Формат кадра в протоколе PPP показан на рисунке 1.11.

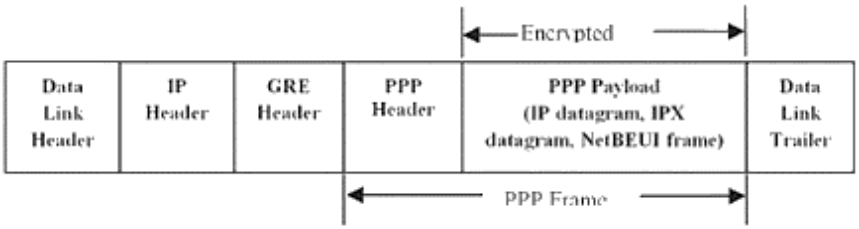

Рисунок 1.11 — Формат кадра в протоколе PPP

L2TP (Layer 2 Tunneling Protocol) — это протокол туннелирования сеансового уровня, который используется для организации соединения типа VPN. Привязан к 1701 UDP-порту.

В рамках протокола L2TP используются сообщения двух видов: управляющие и информационные. Управляющие сообщения необходимы для установления туннелей и вызовов, а также их поддержания и ликвидации. Для обеспечения функции доставки данных сообщений в рамках протокола используется надежный управляющий канал. Информационные сообщения

необходимы для инкапсулирования PPP-кадров, которые передаются в туннеле. При потере пакета он не может быть передан повторно.

Структура протокола L2TP описывает транспортировку кадров PPP и управляющих сообщений по управляющему каналу и каналу данных протокола. PPP-кадры передаются по незащищенному каналу данных, дополняясь заголовком L2TP и на следующем этапе передаются по транспорту для передачи пакетов, например, Frame Relay, ATM и подобных. Управляющие сообщения транспортируются по надежному управляющему каналу протокола L2TP с последующей передачей по тому же транспорту для пересылки пакетов. Формат кадра L2TP показан на рисунке 1.12.

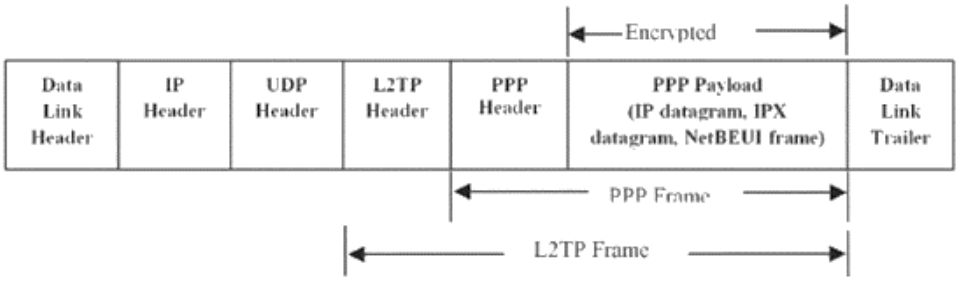

Рисунок 1.12 — Формат кадра L2TP

Управляющие сообщения содержат порядковые номера, которые используются для обеспечения транспортировки по управляющему каналу. Информационные сообщения используют порядковые номера для упорядочивания пакетов и выявления утерянных пакетов.

MPPE (Microsoft Point-to-Point Encryption) - это протокол шифрования информации, который используется поверх установленных соединений PPP. Данный протокол спроектирован на базе алгоритма RSA RC4 [30]. Протокол MPPE поддерживает ключи длиной 40, 56 и 128 бит, которые в течение сессии неоднократно сменяются. Частота смены ключей устанавливается заранее на этапе рукопожатия соединения PPP. Присутствует возможность генерации нового ключа на каждый пакет. Для протокола MPPE необходимо использования ключей шифрования, которые генерируются на этапе проверки подлинности по протоколу типа MS-CHAP.

CHAP (Challenge Handshake Authentication Protocol) — это протокол аутентификации с косвенным согласованием. Данный протокол представляет собой алгоритм проверки подлинности. Он предусматривает транспортировку не открытого пароля пользователя, а только косвенных сведений о нём. Аутентификация узла выполняется в три этапа процедуры согласования. Значение хеша вычисляется при помощи алгоритма хеширования MD5.

MS-CHAP (Microsoft Challenge Handshake Authentication Protocol) — это протокол проверки подлинности связи между клиентом и сервером без передачи клиентского пароля, который использует механизм «вызов-ответ». Протокол MS-CHAP является усовершенствованной реализацией протокола CHAP, предусматривающей механизм возврата сообщений об ошибках аутентификации и предоставляющей возможность смены пароля пользователя. MS-CHAP также обеспечивает генерацию ключей для протокола шифрования MPPE. В отличие от протокола CHAP, для генерации значений хеша применяется алгоритм MD4.

В 2000 году опубликована спецификация протокола MS-CHAP второй версии — **MS-CHAP-V2**. Основным отличием данного протокола от первой версии является присутствие механизма поддержки взаимной аутентификации. Данный протокол не является совместимым с первой версией благодаря различиям в форматах полей сообщений, которые в данной версии также были модифицированы. Для генерации значений хеша используется алгоритм SHA-1.

1.5 Протоколы формирования защищенных каналов на сетевом уровне.

Радикальным подходом к решению задачи устранения уязвимостей компьютерных сетей является создание системы безопасности не только для отдельных классов приложений, а для всей сети в целом [13]. Применительно к IP-сетям это означает, что такие системы защиты должны функционировать на сетевом уровне модели открытых систем OSI. Преимуществом такого подхода является факт, что в сетях IP данный уровень отличается наибольшей гомогенностью: вне зависимости от вышестоящих протоколов, среды передачи

и технологий канального уровня передачи данных по сети не может быть произведена без помощи протокола IP. Поэтому реализация системы безопасности сети на таком уровне гарантирует высокую степень защиты сетевых приложений, причем без какой-либо их модификации.

IPsec (IP Security) представляет собой набор протоколов, которые обеспечивают прозрачную и безопасную защиту данных на сетевом уровне. Основная сложность при организации защищенного канала связи на базе этих протоколов состоит в том, что при установлении связи участникам обмена данными в защищенном канале необходимо согласовать множество различных параметров [14]. Они должны аутентифицировать друг друга, сгенерировать ключи защиты и обменяться ими, договориться, при помощи каких протоколов будет выполняться шифрование данных.

Протоколы, входящие в состав набора IPsec, можно разделить на две группы: протоколы защиты данных и протоколы обмена ключами. Структура алгоритма IPsec показана на рисунке 1.13.

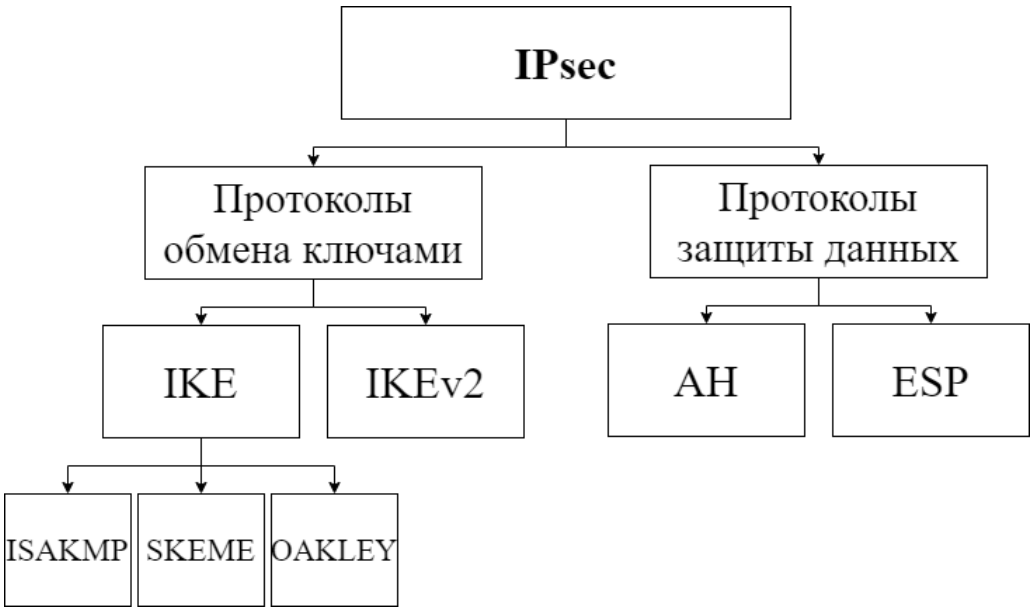

Рисунок 1.13 — Структура алгоритма IPsec.

Главная задача, которую решают протоколы обмена ключами — проведение процедуры аутентификации участников соединения и объявление параметров и ключей шифрования для защиты передаваемых данных.

IKE (Internet Key Exchange) — протокол, который используется для создания, установления, и изменения параметров установки соединения SA (Security Associations) между участниками обмена данными. SA содержат в себе информацию для установления безопасного соединения между участниками предопределенным способом. Протокол IKE основан на протоколах ISAKMP, Oakley и SKEME.

Протокол **IKEv2** является развитием протокола IKE, в котором разработчиками были решены многие проблемы первой версии, также в нем был упрощен механизм согласования ключей, а большинство расширений, таких NAT-T, Keepalives, Mode Config, стали частью данного протокола.

В результате работы IKE и IKEv2 определяются параметры защиты трафика SA и ключи шифрования. Каждый набор параметров SA является однонаправленным, внутри соединения соединение IPSec содержится пара таких SA. Все параметры SA хранятся в специальной базе и доступны администратору защищенного канала для просмотра и сброса.

Основная функция протоколов ESP и AH — защита данных участников обмена.

AH (Authentication Header) — протокол безопасности, который используется для аутентификации источника информации и контроля целостности данных [15]. Также данный протокол может использоваться для защиты от повторной передачи данных. Аутентификация участников обмена данными происходит за счет вычисления хэш-функции пакета IP (поля, которые модифицируются в процессе транспортировки пакета, например TTL, исключаются). Полученная хэш-функция присоединяется к заголовку пакета AH и отправляется другой стороне. В протоколе AH используется HMAC с алгоритмами хеш-функций MD5 или SHA-1.

На базе протокола AH не происходит шифрование данных.

ESP (Encapsulating Security Payload) — протокол, обеспечивающий конфиденциальность и защиту данных в канале связи. При использовании данного протокола присутствует возможность использования механизмов

аутентификации и обнаружения повторных пересылок пакетов. Данный протокол способен полностью инкапсулировать пакеты и может применяться как отдельно, так и совместно с протоколом AH.

Протокол ESP можно функционировать в двух режимах — транспортном и туннельном:

1. В транспортном режиме происходит шифрование только данных пакета IP, исходный заголовок при этом сохраняется. Для построения туннелей при использовании IPsec в транспортном режиме обычно используется связка с другими протоколами туннелирования, например, L2TP.

2. В туннельном режиме происходит шифрование всего исходного пакета IP, включая заголовок, маршрутную информацию, данные. Затем он встраивается в поле данных нового пакета, происходит инкапсуляция.

1.6 Протоколы формирования защищенных каналов на сеансовом уровне

SSL/TLS (Secure Sockets Layer / Transport Layer Security) — набор криптографических протоколов, обеспечивающих защищенную передачу данных в сети на сеансовом уровне.

Протоколы SSL/TLS могут выполнять все функции по организации защищенного канала связи между абонентами, включая взаимную аутентификацию сторон, обеспечение конфиденциальности, проверку целостности и аутентичности передаваемой информации. Ядром протоколов SSL/TLS является технология комплексного использования асимметричных и симметричных алгоритмов шифрования.

В семействе SSL/TLS существует шесть протоколов: SSLv2, SSLv3, TLS v1.0, TLSv1.1, TLSv1.2 и TLSv1.3. Актуальными на данный момент являются версии TLSv1.2 и TLSv1.3. На примере данных версий будет рассмотрен принцип работы системы.

Одной из ключевых фаз работы протокола TLS является рукопожатие. Рукопожатие описывает этап установки соединения. Цель рукопожатия TLS — выполнение всех криптографических преобразований для установки

защищенного соединения, включая проверку подлинности SSL-сертификата и создание ключа шифрования. Основная часть работы, связанной с протоколом TLS, выполняется на данном этапе.

В TLSv1.3 механизм рукопожатия сильно изменился, поэтому его необходимо рассматривать отдельно от предшествующей версии.

Принцип работы механизма рукопожатия TLSv1.2 (вариант с использованием протокола ECDHE для обмена ключами и протокола ECDSA для аутентификации) показан на рисунке 1.14.

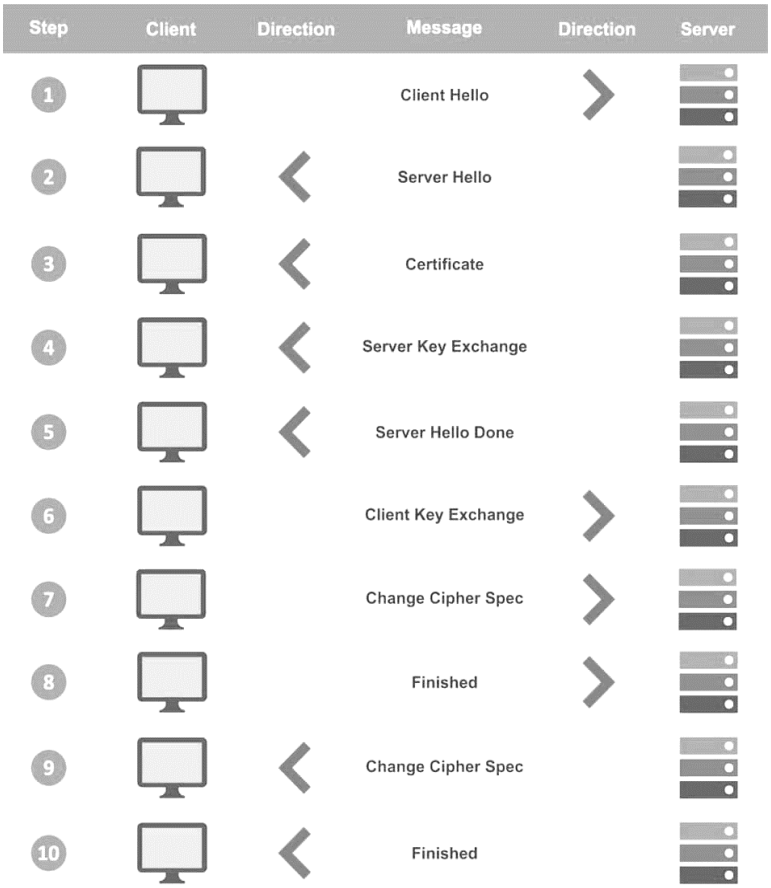

Рисунок 1.14 — Принцип работы механизма рукопожатия TLSv1.2

1. Клиент инициирует подключение сообщением «Client Hello». Оно включает в себя список наборов шифров и случайное число клиента.

2. Сервер посылает ответ «Server Hello», включающий в себя выбранный набор шифров и случайное число сервера.

3. Сервер отправляет SSL-сертификат. Затем клиент выполнит серию проверок достоверности данного сертификата, но, так как протокол DH не может

самостоятельно аутентифицировать сервер, необходимо присутствие дополнительного механизма.

4. Для проведения аутентификации, сервер выбирает случайные числа сторон и параметр DH, использующийся для получения сеансового ключа, и преобразует их с помощью своего приватного ключа [32]. Результат играет цифровой подписи: клиент будет использовать открытый ключ для проверки подписи и того факта, что именно сервер является владельцем пары ключей, затем пришлет ответ собственным параметром DH.

5. Данную фазу завершает сервер, отсылая сообщение «Server Hello Done».

6. Стороны используют параметры DH, которыми они обменялись, для получения pre-master secret. Затем каждая сторона использует pre-master secret для получения сеансового ключа.

7. Клиент отправляет сообщение «Change Cipher Spec» с целью сообщить серверу о своем переходе на шифрованный канал передачи данных.

8. Клиент отправляет сообщение «Finished» с целью сообщить, что он завершил рукопожатие со своей стороны.

9. Аналогичным образом, сервер отправляет сообщение «Change Cipher Spec».

10. Рукопожатие завершается после отправки сообщения «Finished» от сервера.

После выполнения этих шагов рукопожатие считается завершенным. В результате обе стороны обмена данными являются владельцами сеансового ключа, они могут взаимодействовать через зашифрованное и аутентифицированное соединение [18]. На следующем этапе могут быть пересланы первые пакеты данных, которые относятся к фактическому веб-сервису.

Механизм рукопожатия до TLSv1.2 требовал большое количество вычислительных ресурсов и в больших масштабах мог оказать серьезную нагрузку на сервер. Механизм рукопожатия TLSv1.2 также может замедлить

работу сервера, если происходит большое количество обращений к нему в один момент времени.

В отличие от версии 1.2, механизм рукопожатия TLSv1.3 укладывается в одну фазу. Это дает значительный прирост к производительности системы по сравнению с предшествующим протоколом.

Принцип работы механизма рукопожатия TLSv1.3 показан на рисунке 1.15.

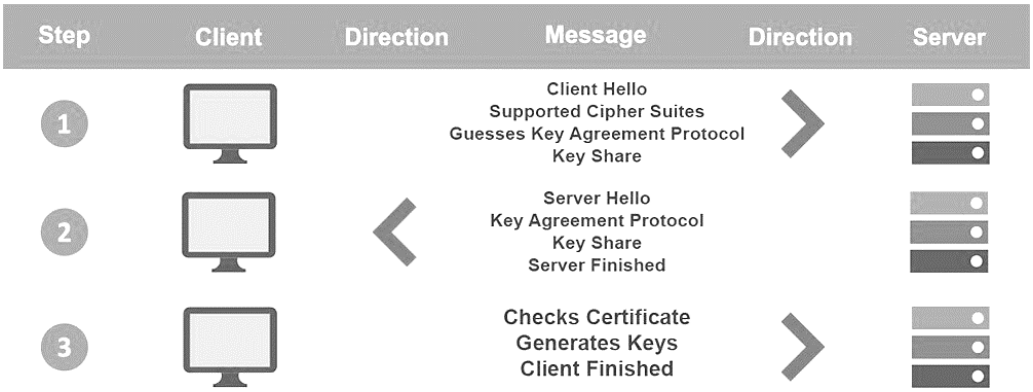

Рисунок 1.15 — Принцип работы механизма рукопожатия TLSv1.3

Сообщение «Client Hello» запускает механизм рукопожатия. В TLSv1.3 сокращено число поддерживаемых наборов шифров с 37 до 5, поэтому клиент может угадать, какие ключи или протокол обмена будут использоваться в системе и в дополнение к сообщению пересылает свою часть общего ключа из предполагаемого протокола.

Сервер посылает ответ сообщением «Server Hello». На данном этапе отправляется сертификат. Если клиент верно определил протоколы защиты с присоединенными данными и сервер дал утвердительный ответ, сервер пересылает свою часть ключа, затем вычисляет сеансовый ключ и завершает процесс передачи с помощью сообщения «Server Finished».

Когда клиента получает все необходимые данные, он верифицирует SSL-сертификат и вычисляет свою копию сеансового ключа, используя два общих ключа [20]. Когда данный процесс завершается, клиент отправляет сообщение «Client Finished».

HTTPS (HyperText Transfer Protocol Secure) — это расширение протокола HTTP, поддерживающее шифрование посредством протоколов SSL/TLS.

Основными реализациями SSL/TLS VPN являются OpenVPN и SSTP.

OpenVPN — программный продукт с открытым исходным кодом, предназначенный для реализации защищенного канала передачи данных в виртуальной частной сети. Данный продукт использует готовый набор криптографических решений, позволяющий создавать ключи шифрования и работать с сертификатами на основе библиотеки OpenSSL. Для обеспечения аутентификации используется HMAC. OpenVPN поддерживает два варианта установления соединения PSK и PKI. Данное решение совместимо со многими операционными системами.

Используемые в OpenVPN шифрования библиотеки шифрования OpenSSL поддерживают большинство криптографических алгоритмов (AES, Blowfish, Camellia и другие). По умолчанию для шифрования используется BlowFish в режиме длиной ключа 128 бит, а для HMAC используется SHA1 и 160 битный ключ. В качестве ключей используются соответственно первые 128 и 160 бит от 512 битных ключей из файла со статическим ключом.

SSTP (Secure Socket Tunneling Protocol) — протокол VPN от компании Microsoft, основанный на алгоритме защищенного канала SSL и включенный в состав их операционных систем. По умолчанию для установки соединения используется порт 443. Механизм шифрования реализован на базе SSL, для аутентификации используется SSL и PPP [33].

Протокол SSTP использует функции других криптографических систем. Единственная функция, реализуемая самим протоколом SSTP — cryptographic binding [1].

На уровне PPP происходит процесс авторизации клиента на сервере, также присутствует возможность аутентификации сервера. Методы аутентификации SSTP идентичны протоколу PPTP. Отличие от состоит в том, что обмен данными происходит внутри уже созданного защищенного канала SSL.

Аутентификация сторон обмена происходит на разных уровнях, поэтому возможна атака MITM, если злоумышленник установит соединение SSL с сервером и PPP соединение с клиентом. Для защиты от данной атаки используется механизм подписания сообщений уровня SSTP, содержащих информацию об установлении соединения с помощью ключа, выработанного в процессе аутентификации на PPP уровне [2]. После выполнения данных процедур, сервер может быть уверен в том, что хост, установивший SSL соединение, и хост, прошедший PPP аутентификацию — это один клиент (операция cryptographic binding).

В процессе PPP аутентификации между легитимными участниками обмена данными генерируется общий секрет, который невозможно получить при неавторизованном доступе к соединению PPP [3]. Кроме того, злоумышленник не сможет заставить клиента подписать сообщение SSTP, так как сам клиент считает, что он установил незащищенное соединение PPP и не имеет достоверной информации про SSTP соединение.

1.7 Выводы по главе 1

В главе рассмотрена проблема обеспечения безопасного обмена данными в глобальной сети между абонентом и облачным сервером. Определены пути решения данной проблемы.

Проведен анализ актуальных методов аутентификации, алгоритмов шифрования данных, криптографических хеш-функций. Указаны сходства и различия приведенных технологий

Рассмотрены сервисы, позволяющие организовать защищенный канал связи между абонентом и сервером на разных уровнях модели открытых систем OSI. Указаны достоинства и недостатки каждого из решений.

Разработана классификация мобильных абонентов — пользователей веб-сервисов.

Разработана классификация передаваемых данных в канале связи между мобильным абонентом и облачным сервисом.

Глава 2. Разработка модели оценки качества защищенного канала обмена данными

2.1 Постановка задачи

Целью данной работы является разработка методов аутентификации, позволяющих уменьшить потери сетевых пакетов и обеспечивающих высокую степень защищенности данных мобильных абонентов при обращении к облачному сервису в условиях динамической смены сетей.

Для достижения поставленной цели должны быть решены следующие **задачи**:

1. Разработка модели оценки качества защищенного канала передачи данных. Необходимо выделить первичные характеристики канала связи и определить оценочные. На основании результатов измерений и расчетов данных характеристик можно оценить качество канала передачи данных.
2. Разработка классификации мобильных абонентов — пользователей веб-сервисов. При помощи данной классификации существует возможность описания любого мобильного абонента.
3. Разработка классификации передаваемых данных. Показатели характеристик канала обмена данными обмена могут различаться в зависимости от типа передаваемых данных, поэтому данный параметр также необходимо учитывать при организации защищенного канала обмена данными.
4. Разработка сценариев — наборов характеристик мобильных абонентов и передаваемых данных, определяющих условия обмена данными.
5. Анализ актуальных данных о криптостойкости алгоритмов шифрования, криптографических хеш-функций, сервисов защищенного канала обмена данными и определение набора методов аутентификации, обеспечивающих максимальный уровень защищенности информации при умеренных требованиях к производительности системы абонента.
6. Разработка имитационной модели защищенного канала обмена данными между мобильным абонентом и облачным сервисом.

7. Реализация на модели набора методов аутентификации.

8. Исследование эффективности применения методов аутентификации в различных сценариях подключения мобильных абонентов к облачным сервисам при помощи экспериментов на модели и последующего анализа результатов.

Объектом исследования являются вычислительные системы и сетевое программное обеспечение, позволяющее организовать защищенный канал связи между мобильным абонентом и облачным сервисом.

Предметом исследования являются свойства алгоритмов и протоколов, обеспечивающих конфиденциальность, целостность и аутентичность данных в защищенном канале связи между мобильным абонентом и облачным сервисом.

2.2 Модель оценки качества защищенного канала обмена данными

Оценивать защищенный канал обмена данными рекомендуется по следующей методике:

1) Измерение первичных характеристик защищенного канала за счет анализа трафика аппаратно-программным комплексом в процессе обмена данными между мобильным абонентом и облачным сервисом.

2) Расчет оценочных характеристик (средних значений и коэффициентов) на основании измерений.

3) Сравнение полученных результатов с идеальными.

На первом этапе, генерация пакетов должна происходить как можно ближе к выходу измеряемого участка, в противном случае невозможно обеспечить точность из-за возрастания числа факторов, влияющих на измерения.

Следующие характеристики являются первичными и регистрируются аппаратно-программным комплексом в процессе обмена данными:

– Скорость передачи данных в момент времени;
– Количество потерянных пакетов в сеансе связи;
– Время задержки каждого пакета;
– Количество требуемых аппаратных ресурсов в момент времени.

Аппаратно-программным комплексом должен автоматически фиксироваться объем переданной информации V, время T, за которое данный объем информации был передан и рассчитываться **скорость** U передачи данных в момент времени по формуле:

$$U = \frac{V}{T} \tag{2.1}$$

Недостаточная пропускная способность, недостаточная производительность сетевого оборудования, ошибки на физическом и программном уровне канала связи при передаче данных могут проявиться в таком негативном явлении как **потеря пакетов**. Если в процессе передачи данных пакет не достигнет пункта назначения в течение определенного времени ожидания, то он считается потерянным. Потеря пакетов влечет за собой сбои в сетях, особенно в глобальных.

С помощью аппаратно-программного комплекса должны автоматически определяться и фиксироваться в системе потерянные пакеты.

Количество потерянных пакетов $P_{\text{пот}}$ определяется суммой всех таких пакетов p:

$$P_{\text{пот}} = \sum_{k=1}^{n} p_k, \tag{2.2}$$

Задержки при передаче пакетов обусловлены многими причинами. Они определяются работой узлов на различных уровнях — от физического до транспортного. При условии передачи данных в глобальной сети, основными местами возникновения задержек являются узлы доступа и шлюзы между провайдерами.

Аппаратно-программным комплексом для каждого пакета автоматически должно фиксироваться время передачи T_1 пакета IP из начальной точки, время

приема T_2 пакета в конечной точке и рассчитываться характеристика задержки для каждого пакета.

Задержка L передачи пакета IP определяется как разность полученных временных показателей:

$$L = T_2 - T_1 \tag{2.3}$$

Процесс обмена данными напрямую связан с потреблением аппаратной части ресурсов вычислительных систем абонентов. При использовании различных каналов связи показатели **загрузки** центральных процессоров устройств могут быть совершенно разными.

Аппаратно-программным комплексом в процессе обмена данными по защищенному каналу связи контролируется значение загрузки центрального процессора Z абонента в момент времени. Показатель фиксируется в % от общего количества ресурсов M центрального процессора абонента и вычисляется по формуле:

$$Z = \frac{M}{m} \times 100\%, \tag{2.4}$$

где m — количество потребляемых аппаратных ресурсов.

На основании данных, полученных при измерениях, можно рассчитать значения характеристик, которые будут учитываться при оценке качества защищенного канала передачи данных.

К оценочным характеристикам относятся:

- Средняя скорость передачи данных;
- Пропускная способность канала передачи данных;
- Коэффициент потерь пакетов;
- Средняя задержка передачи пакета;
- Среднее время восстановления сеанса связи;

– Средний показатель загрузки аппаратных ресурсов.

В течение времени измерений аппаратно-программным комплексом фиксируется минимальная и максимальная скорость передачи данных. Средняя скорость передачи данных рассчитывается по формуле:

$$U_{\text{ср}} = \frac{\sum_{i=1}^{L}(U_i)}{L} \tag{2.5}$$

Преобразование трафика в защищенном канале передачи данных, вызванное процессами туннелирования и шифрования, образует накладные расходы и приводит к снижению **пропускной способности** такого канала. Необходимо добиться максимального значения данной характеристики, обеспечивая при этом высокий уровень безопасности передаваемой информации.

Для количественной характеристики пропускной способности канала, в котором информация является однородной (только веб-трафик), применяется единица измерения бит/сек.

При этом под пропускной способностью Q канала понимается максимальное значение скорости U передачи информации, допускаемое в данном канале при заданных требованиях к точности передачи информации:

$$Q = max\, U \tag{2.6}$$

Коэффициент потерь пакетов PL определяется как отношение общего числа потерянных пакетов $P_{\text{пот}}$ между двумя контрольными точками к общему числу переданных пакетов P:

$$PL = \frac{P_{\text{пот}}}{P} \tag{2.7}$$

Средняя задержка $L_{\text{ср}}$ передачи пакета IP определяется как среднее количество задержек $T_i - T_{i-1}$ переданных пакетов P:

$$L_{\text{ср}} = \frac{\sum_{i=1}^{P}(T_i - T_{i-1})}{P} \quad (2.8)$$

При возникновении разрывов соединения в канале связи, абонентам необходимо время чтобы восстановить его и продолжить обмен данными. Чем меньше данная временная характеристика, тем выше уровень безопасности передаваемых данных. Особенно данная характеристика важна при условии обмена данными мобильных абонентов. В таких условиях среда передачи данных становится неустойчивой и характеризуется высоким количеством разрывов.

Расчет данной характеристики требует фиксации временных показателей на каждой стороне обмена данными. Учитывается время обращения сервера к клиенту t_1 и время отправки клиентом первого пакета данных серверу t_2. Процесс разрывов соединения можно симулировать строго по времени с помощью программ имитационного моделирования, поэтому время восстановления связи можно определить с довольно высокой точностью.

Время восстановления сеанса связи t рассчитывается по формуле:

$$t = t_2 - t_1 \quad (2.9)$$

2.3 Разработка классификации мобильных абонентов

Классификация создана с целью описать любого мобильного абонента веб-сервиса. Учитывается то, как он передвигается и какие аппаратные ресурсы содержит в себе. Данные факторы будут существенно влиять на качество передачи данных в реальных жизненных ситуациях, что и планируется увидеть в результатах экспериментов. В контексте данной работы было выделено два основных признака классификации.

Смена IP-адреса абонента может быть как частым явлением и происходить, например, несколько раз в минуту, так и наоборот, раз в пять минут. В данной классификации было принято решение условно обозначить данный параметр как «характер движения абонента». Абонент, который продолжительное время не меняет IP-адрес, предположительно, движется с низкой скоростью перемещения. Абонент, который часто меняет IP-адрес, движется с высокой скоростью перемещения.

Процессы аутентификации, передачи данных, шифрования/дешифрования напрямую связаны с потреблением аппаратной части ресурсов вычислительной системы. Поэтому необходимо учитывать их характеристики при проектировании защищенного канала передачи данных. В данной классификации выделены две группы абонентов: к первой относятся устройства с аппаратными ресурсами малой мощности, то есть влияние процесса обмена данными на их загрузку является существенным, во вторую входят устройства с мощностью аппаратных ресурсов достаточной для того, чтобы данным показателем в процессе обмена данными можно было пренебречь (высокой мощности). Так как основная часть нагрузки направлена на центральный процессор устройства, именно его характеристики будут учитываться во время принятия решения, какой из групп классификации будет принадлежать мобильный абонент.

Классификация мобильных абонентов показана на рисунке 2.1.

Рисунок 2.1 — Классификация мобильных абонентов

2.4 Разработка классификации данных

Мобильные абоненты могут передавать/принимать различную информацию с разной частотой. От типа информации, зависит качество её обмена. Необходимо классифицировать данные, чтобы в дальнейшем определить сценарии обмена данными.

Регулярно обновляемые данные, которые показывают текущее состояние абонента можно представить как поток большого количество пакетов малого размера. К такому типу данных могут относится сведения о геопозиции, состоянии аппаратных ресурсов устройства в реальном времени или данные о траектории движения абонента. В классификации такой тип предлагается обозначить как «динамический поток данных». В случае, если данными при обмене являются файлы большого объема в единственном числе, и они не обновляются в реальном времени, то такой тип данных в классификации относится к группе «статических данных». К такому типу данных могут относится, например, пакеты обновлений встроенного программного обеспечения устройства, файлы журналов операционной системы,

дистрибутивы.

Не все данные нуждаются в комплексной защите. Злоумышленник может получить доступ к информации, утечка которой не несет значимых последствий для пользователя и Интернет-ресурса, так и получить доступ к данным, раскрытие или пропажа которых приведут к существенному ущербу для обеих сторон. В классификации предлагается выделить две группы: «высокий класс защищенности данных» и «низкий класс защищенности данных».

Классификация передаваемых данных показана на рисунке 2.2.

Рисунок 2.2 — Классификация передаваемых данных

2.5 Разработка сценариев обмена данными

Используя полученные группы в приведенных классификациях, можно составить комбинации решений «Абонент + Данные», тем самым определить сценарии обмена данными мобильного абонента и веб-сервера.

Сценарий — условия, при которых происходит обмен данными.

Для каждого сценария будет предложено решение об использовании определенного метода аутентификации, исходя из результатов экспериментов. Это может оказать помощь при проектировании какого-либо устройства, которое

обращается к веб-сервису посредством беспроводной технологии передачи данных.

Для удобства записи каждой группы классификации в таблицы решений, предлагается использовать сокращения в виде переменных S/s (Speed), P/p (Power), D/d (Data), E/e (Encrypt). Соответствие групп классификаций и переменных приведено в таблице 2.

Таблица 2 — Группы классификаций мобильных абонентов и передаваемых данных

Мобильный абонент	Характер движения	Высокая скорость = **S**
		Низкая скорость = **s**
	Аппаратные ресурсы	Высокая мощность = **P**
		Малая мощность = **p**
Данные	Тип	Статические = **D**
		Динамические = **d**
	Класс защищенности	Высокий класс = **E**
		Низкий класс = **e**

Итоговое число решений (сценариев) — 16.

Сценарии записываются в следующем виде: SPDE, SPDe, SPdE, SpDE, SPde, SpdE, SpDe, Spde, sPDE, sPDe, sPdE, spDE, sPde, spDe, spdE, spde.

Пример 1: Автомобиль движется со скоростью 100 км/ч по трассе (S) и передает каждые 3 секунды информацию о своем местоположении серверу в общий доступ (d, e), используя на борту устройство с беспроводной технологией передачи LTE и одноядерным ЦП с тактовой частотой ядра 400 МГц (p).

Пример 2: Человек идет по торговому центру (s) и получает пакет обновлений безопасности на свое мобильное устройство (D, E), используя при этом технологию передачи Wi-Fi. Устройство оснащено восьмиядерным ЦП с тактовой частотой ядра 2.8 ГГц (P).

Для решения поставленных в данной работе задач, данных решений будет достаточно, однако, сценарии могут дополняться новыми критериями и

корректироваться в зависимости от потребностей производителя оборудования. В таком случае, предложенный для исходного сценария метод аутентификации может являться не лучшим решением, но всё еще предлагается в качестве рекомендуемого.

2.6 Выводы по главе 2

В данной главе выполнена постановка задачи разработки методов аутентификации мобильных абонентов для доступа к облачным сервисам.

Разработана модель оценки качества защищенного канала обмена данными. Определен набор первичных характеристик, измеряемых программно-аппаратным комплексом. Определен набор оценочных характеристик рассчитываемых на основе результатов измерений.

Разработана система сценариев – условий обмена данными в защищенном канале между мобильным абонентом и облачным сервисом.

Глава 3. Разработка методов аутентификации и реализация их на модели

3.1 Разработка методов аутентификации

Любое действие в интернете — это обмен данными, при котором каждый раз устройство делает запрос к необходимому серверу и получает от него ответ. Стандартно такой обмен данными происходит по протоколу HTTP, основной недостаток которого — незащищенность передаваемых данных, что абсолютно не приемлемо, например, когда речь идет о передаче конфиденциальной информации [11]. Однако, отсутствие механизмов шифрования позволит достигнуть максимальных результатов в экспериментах при реализации HTTP-аутентификации на модели. Данные значения могут быть приняты как эталонные. В таблицы решений данный метод аутентификации входить не будет.

В данной главе разработаны методы аутентификации на основе TLS и VPN, избраны комбинации шифров на основе актуальных данных о криптостойкости алгоритмов защищенного канала.

Разработка методов на основе TLS. Криптосистемы в TLS объединяются в типовые наборы шифров. Чтобы начать обмен информацией по защищённому каналу, клиент и сервер должны согласовать используемый набор [35]. Параметры шифров и обмена сопутствующей информацией должны быть совместимы между собой. Согласование проводится на этапе установления соединения. Композиции наборов существенно отличаются в TLSv1.3 и в предшествующих версиях. В TLSv1.2 и младше в набор шифров входят:

- Алгоритм обмена ключами (RSA, DH, ECDH, DHE, ECDHE, PSK)
- Алгоритм цифровой подписи (RSA, ECDSA, DSA)
- Алгоритм массового шифрования (AES, CHACHA20, Camellia, ARIA)
- Алгоритм кода аутентификации сообщений (SHA-256, POLY1305)

Пример: ECDHE-ECDSA-AES256-SHA384

В TLSv1.3 криптосистемы, служащие для аутентификации узлов и получения общего секрета, отделены от шифров, что потребовало изменения организации реестра и механизма нумерации наборов шифров. Наборов шифров

TLSv1.3 меньше, а их имена содержат только указания на шифр и хеш-функцию. Наборы строго фиксированы, состав закреплён в RFC, каждому приписан свой индекс, реестр ведёт организация IANA.

Пример: TLS_AES_256_GCM_SHA384

TLSv1.3 разрешает исключительно аутентифицированное шифрование, которое не требует использования дополнительной хеш-функции для вычисления кода аутентификации сообщения (в схемах аутентифицированного шифрования некоторый аналог хеш-функции встроен в саму схему) [15]. Тем не менее, хеш-функция требуется для вычисления сессионных симметричных ключей. Поэтому в набор TLSv1.3 входит указание на тип хеш-функции.

TLSv1.3 существенно ужесточает принципы включения наборов шифров в реализации, допущены только два шифра: AES и ChaCha20.

Выбор алгоритмов обмена ключами. Обеспечение совершенной прямой секретности. Совершенная прямая секретность — свойство некоторых протоколов согласования ключа, которое гарантирует, что сессионные ключи, полученные при помощи набора ключей долговременного пользования, не будут скомпрометированы при компрометации одного из долговременных ключей [24]. С помощью комплектов шифров, которые не обеспечивают прямой секретности, злоумышленник, восстановивший закрытый ключ сервера, может расшифровать все ранее записанные зашифрованные сообщения.

Необходимо поддерживать наборы шифров, включающие алгоритм обмена ключами ECDHE для обеспечения совершенной прямой секретности. Для поддержки более широкого круга клиентов также можно использовать наборы, включающие алгоритм DHE в качестве резервного варианта. Использовать алгоритм обмена ключами RSA можно только в случае крайней необходимости [16]. Данный алгоритм все еще является широко используемым, но не обеспечивает совершенной прямой секретности в процессе обмена ключами.

Выбор алгоритма цифровой подписи. Для организации защищённого соединения в TLS используются асимметричные криптосистемы,

криптосистемы цифровой подписи. На практике большинством облачных сервисов используются алгоритмы RSA и ECDSA.

Криптография с эллиптической кривой может обеспечить относительно тот же уровень безопасности, что и алгоритм RSA, при меньшем размере ключа [17]. Ключи меньшего размера требуют меньшей полосы пропускания для настройки SSL/TLS, что означает, что алгоритм ECDSA лучше подходит для генерации ключей мобильных абонентов.

Криптостойкость алгоритмов ECDSA и RSA принято считать эквивалентной.

Выбор симметричных шифров. В состав криптонабора TLS входит симметричный шифр. С его помощью осуществляется шифрование потока данных, передаваемых через TLS-сокет. Шифры AES и ChaCha20 шифры закрывают основную часть TLS-трафика современных облачных сервисов [19].

Алгоритм AES отличается тем, что реализация с высокой производительностью на архитектурах, не имеющих достаточного объема памяти и ряда специальных команд, сталкивается с серьезными трудностями. По сравнению с AES, шифр ChaCha20 проще по алгоритмической структуре и составу операций [21]. Он алгоритмически ближе к шифрам, предназначенным для микроконтроллеров и других встроенных применений, что позволяет получить достаточно быстрые, безопасные и легко портируемые программные реализации на разном аппаратном обеспечении, что в условии подключения мобильного абонента к облачному сервису дает свои преимущества.

Криптографическая стойкость алгоритма ChaCha20, как правило, считается эквивалентной AES.

Выбор длины ключа. Использование слишком короткого ключа небезопасно, но использование слишком длинного ключа приведет к ухудшению производительности и излишнему уровню безопасности [22]. Для большинства веб-сервисов использование ключей ECDSA, превышающих 256 бит, приводит к увеличению нагрузки на аппаратные ресурсы клиента и сервера и может негативно сказаться качестве передачи данных. Также нет необходимости в

увеличении силы обмена эфемерным ключом за пределами 256 битов для ECDHE.

Результат разработки методов аутентификации на основе TLS. С помощью команды *openssl ciphers* можно получить полный список наборов шифров SSL/TLS. Он представлен на рисунке 3.1.

Рисунок 3.1 — Список наборов шифров SSL/TLS

Основываясь на представленных рекомендациях и исследованиях, методом отсеивания были избраны методы аутентификации, которые обеспечивают надежное шифрование трафика при лучших показателях производительности. Данные методы аутентификации представлены в таблице 3.

Таблица 3 — Методы аутентификации на основе TLS

Имя пакета	**Kx**	**Au**	**Enc**	**Mac**
TLS_CHACHA20_POLY1305_SHA256	any	any	CHACHA20/ POLY1305(256)	AEAD
TLS_AES_256_GCM_SHA384	any	any	AESGCM(256)	AEAD
ECDHE-ECDSA-CHACHA20-POLY1305	ECDH	ECDSA	CHACHA20/ POLY1305(256)	AEAD
ECDHE-ECDSA-AES256-GCM-SHA384	ECDH	ECDSA	AESGCM(256)	AEAD

Разработка методов на основе протоколов туннелирования. Протокол PPTP был объектом множества анализов безопасности, в нём были обнаружены

различные серьёзные уязвимости [23]. Известные относятся к используемым протоколам аутентификации PPP, устройству протокола MPPE и интеграции между аутентификациями MPPE и PPP для установки сессионного ключа.

MPPE использует RC4 поток для шифрования. Не существует метода для аутентификации цифробуквенного потока и поэтому данный поток уязвим для атаки, делающей подмену битов. Злоумышленник легко может изменить поток при передаче и заменить некоторые биты, чтобы изменить исходящий поток без опасности своего обнаружения [27]. Данная подмена битов может быть обнаружена с помощью протоколов, считающих контрольные суммы. Использовать данный протокол для создания защищенного канала связи крайне не рекомендуется.

Протокол L2TP так же, как и PPTP для защиты информации может использовать шифрование MPPE, но так как этот метод не является надежным, L2TP должен использоваться в связке с IPsec.

IPSec является одним из самых безопасных протоколов VPN за счет использования криптостойких алгоритмов шифрования. Он позволяет использовать различные алгоритмы для шифрования данных и для проверки целостности передаваемых данных, и аутентификации. Для проверки целостности данных и аутентификации IPsec использует протоколы SHA-1, SHA-2 и MD5, для шифрования данных - DES, 3DES, Blowfish, AES, Camelia, Twofish с различными вариантами длины ключа, для проверки целостности данных, аутентификации и шифрования используется AES. Большинство данных алгоритмов являются актуальными на данный момент и могут использоваться.

Алгоритм Camellia не раз подвергался криптоанализу [34]. Самими же разработчиками была доказана стойкость алгоритма к линейному и дифференциальному криптоанализу, а также к использованию усеченных и невозможных дифференциалов, методу бумеранга, интерполяции, сдвиговым атакам и ряду других атак. Camellia на данный момент является одним из наиболее стойких алгоритмов шифрования. Однако, в ходе криптоанализа

большинством экспертов было установлено, что алгоритм Camellia существенно проигрывает в скорости алгоритму AES и предъявляет достаточно высокие требования к оперативной и энергозависимой памяти, поэтому для устройств с ограниченными аппаратными ресурсами данный алгоритм не считается пригодным наряду с AES.

Использование необратимых подстановок, зависимость узлов замен от ключа, большой размер узлов замен, переменный размер ключа, сложная схема выработки ключевых элементов - требует выполнения 521 цикла шифрования, что существенно затрудняет переборную атаку на алгоритм Blowfish, однако, делает его непригодным для использования в системах, где ключ часто меняется и на каждом ключе шифруется небольшие по объему данные. Алгоритм лучше всего подходит для систем, в которых на одном и том же ключе шифруются большие массивы данных [25]. Известно, что вариант Blowfish с уменьшенным количеством раундов является уязвимым для атак на основе открытых текстов на сравнительно слабых ключах. Реализации Blowfish с 16 раундами шифрования не подвержены подобным атакам.

Изучение Twofish с сокращенным числом раундов показало, что алгоритм обладает большим запасом прочности, и, по сравнению с остальными шифрами конкурса AES, он оказался самым стойким. Недостатком алгоритма Twofish является тот факт, что он не очень эффективен в отношении аппаратного обеспечения [28]. Он требует больше памяти и больше циклов в порядке для шифрования данных по сравнению с AES. Данное влияние на устройства высокого класса не будет критичным, но для устройств низкого уровня разрыв производительности AES и Twofish может быть очень значительным.

Рассмотренные алгоритмы шифрования (AES, Blowfish, Camellia, Twofish) используются в библиотеке OpenSSL, на базе которой работает протокол OpenVPN. Следовательно, его использование также является возможным при выборе определенных стойких алгоритмов шифрования.

Протокол SSTP для организации защищенного канала использует SSLv.3 и, следовательно, предлагает аналогичные преимущества, что и OpenVPN,

поэтому также может использоваться при выборе сильных алгоритмов шифрования.

Выбор хэш-функций. Основное требование к хэш-функции - минимальная вероятность коллизий - нахождения другой входной строки, на котором хэш-функция выдаст тот же результат.

SHA-1, SHA-256, SHA-512 различаются по размеру вывода, размеру внутреннего состояния, размеру блока, размеру сообщения и раундам [26]. За десять лет с момента изобретения SHA-1 не было известно ни об одном практическом способе генерации коллизий, но в 2017 году сотрудники компании Google и Центра математики и информатики в Амстердаме предоставили первый алгоритм генерации коллизий для SHA-1. В качестве защиты от атаки на отыскание коллизий SHA-1 компания Google рекомендует перейти на более качественные криптографические хеш-функции SHA-256 и SHA-512.

SHA-512 будет тратить на четверть больше раундов (80 против 64), чем SHA-256. Но каждый раунд у SHA-512 обрабатывает 64-битовые операнды, а у SHA-256 — 32-битовые [29]. Удвоенное КПД по операндам в сочетании с гарантированной потерей четверти скорости на дополнительных раундах даст 1.6 раз преимущества. На производительность алгоритма могут оказать влияние дополнительные затраты в рамках доступа к памяти, использования шины и так далее — но в целом можно говорить о том, что SHA-512 на рабочих объемах данных будет примерно в 1.5 раза быстрее, чем SHA-256.

Важный вывод состоит в том, что расчет хешей урезанных вариантов SHA-224 и SHA-384 проводится за такое же время, сколько и при использовании основных SHA-256 и SHA-512.

Результат разработки методов на основе VPN-туннелей. Основываясь на представленных рекомендациях и исследованиях, были избраны методы аутентификации при использовании технологий VPN, которые обеспечивают высокую степень безопасности процесса передачи данных при лучших показателях производительности системы. Данные методы аутентификации представлены в таблице 4.

Таблица 4 — Методы аутентификации на основе VPN

Имя решения	VPN-протокол	Алгоритм шифрования	Алгоритм хеш-функции
OpenVPN-TLS(AES256)-SHA-1	OpenVPN	TLS(AES256)	SHA-1
OpenVPN-TLS(Blowfish)-SHA-1	OpenVPN	TLS(Blowfish)	SHA-1
SSTP-TLS(AES256)-SHA256	SSTP	TLS(AES256)	SHA256
L2TP/IPsec(PSK)-ECP256-AES256GCM-SHA256	L2TP/IPsec	AES256GCM	SHA256
L2TP/IPsec(PSK)-ECP256-Blowfish-SHA256	L2TP/IPsec	Blowfish	SHA256
L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	L2TP/IPsec	Camellia256	SHA256
IKEv2-ECP256-AES256GCM-SHA-256	IPsec (IKEv2)	AES-256GCM	SHA256
IKEv2-ECP256-Blowfish-SHA-256	IPsec (IKEv2)	Blowfish	SHA256
IKEv2-ECP-256-Camellia256-SHA-256	IPsec (IKEv2)	Camellia256	SHA256

3.2 Разработка имитационной модели

Процесс разработки модели, позволяющей имитировать обращения мобильного абонента к облачному сервису по защищенному каналу передачи данных, включает в себя три этапа:

1. Реализация веб-сервера (HTTP) и незащищенного канала передачи данных между мобильным абонентом и облачным сервисом.

Данный этап включает в себя процесс установки, настройки и оптимизации веб-сервера nginx и первоначальную настройку маршрутизаторов ROUTER1 и ROUTER2 (WAN, NAT, Routes).

2. Реализация защищенного канала передачи данных с помощью протоколов безопасности (HTTPS, SSTP, OpenVPN, L2TP/IPsec, IKEv2).

Данный этап включает в себя процесс модификации веб-сервера и переход на протокол HTTPS, установку VPN-серверов на маршрутизаторе ROUTER1 и VPN-клиентов на маршрутизаторе ROUTER2 (PPP, Firewall, NAT, Routes).

3. Разработка программ имитационного моделирования, позволяющих имитировать поведение мобильного абонента в сети Интернет.

Данные программы представляют собой реализацию Dual-WAN с автоматическим переключением канала.

Схема сети имитационной модели представлена на рисунке 3.2.

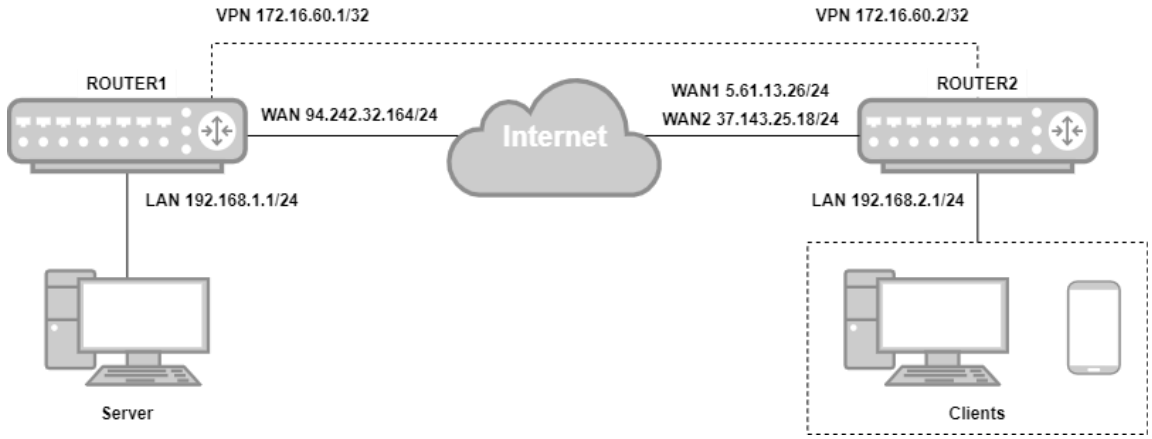

Рисунок 3.2 — Схема сети имитационной модели

Аппаратная составляющая модели

Конфигурация веб-сервера: Dell Optiplex 5060 Micro (Intel Core i7-8700T, 8GB RAM, 512GB SSD), Ubuntu v18.04.4 LTS.

Конфигурация абонента: Lenovo ThinkCentre 3209B4G (Intel Core i5-3550, 4GB RAM, 256GB SSD), Ubuntu v18.04.4 LTS.

С точки зрения аппаратной составляющей, решения для клиента и сервера могут быть гораздо менее производительны представленных вариантов. Для модели были выбраны такие решения, чтобы показателями их производительности можно было пренебречь при проведении экспериментов.

Силами операционной системы создается подключение между мобильным абонентом и сервером, инициируется процесс передачи данных, регистрируются показатели передачи с помощью встроенных средств или стороннего ПО, автоматизируется данный процесс при помощи программ имитационного моделирования. Реализовать эти операции можно практически на любой современной ОС. Для создания условий, приближенных к идеальным, на модели используются идентичные ОС для абонента и сервера.

Маршрутизаторы сервера и клиента: MikroTik RB1100AHX4 (AL21400 4x1.4 GHz, 1GB RAM, 128 MB NAND, IPsec hardware acceleration), RouterOS v6.46.4 (Stable).

Данный маршрутизатор имеет высокую производительность, достаточную для создания множества VPN-соединений. Гибкий в настройке, позволяет реализовать различные сценарии прохождения трафика.

Маршрутизатор MikroTik RB1100AHX4 поддерживает все протоколы шифрования IPSec на аппаратном уровне, что должно снизить нагрузку на ЦП устройства. Для сравнения производительности метода аутентификации, использующего IPsec с другими методами, имеет смысл отключение данной функции. Однако, учитывая нагрузку ЦП при использовании функции аппаратного ускорения IPsec и без неё, можно сделать вывод о количестве потребляемого ресурса ЦП устройства, поэтому эксперименты будут производиться в двух режимах.

После реализации метода аутентификации на описанной выше модели, начинается следующий этап — проведение экспериментов. Абонент инициирует подключение к веб-серверу и принимает файлы (при этом на процесс передачи данных оказано воздействие, вызванное программами имитационного моделирования. С помощью встроенных в операционную систему клиента и маршрутизатора служб, значения характеристик канала фиксируются и заносятся в итоговую таблицу для дальнейшего анализа. Затем реализуется другой метод аутентификации и снова проводятся эксперименты.

.

3.3 Реализация веб-сервиса и незащищенного канала обмена данными

Процесс реализации веб-сервиса включает в себя следующие этапы:

1. Перед установкой http-сервера необходимо задается статический ip-адрес хосту.
2. Производится установка nginx с помощью менеджера пакетов apt.
3. Настраивается Firewall. Прохождение трафика разрешается для портов 80 и 443.
4. Создается директория для веб-сервиса, настраиваются права доступа для созданной директории, создается файл страницы для проверки соединения
5. Создается файл конфигурации. В блоке server файла конфигурации необходимо указать все адреса и порты, на которых нужно принимать соединения для этого сервера, а также указать имя сервера.
6. Создаются ссылки на конфигурационный файл в директориях nginx.

Настройка маршрутизатора сервера (ROUTER1)

Все настройки проводятся поверх базовой конфигурации маршрутизатора.

1. Задается имя WAN интерфейса, устанавливается внешний IP-адрес, указывается стандартный маршрут.
2. Для обращения к веб-серверу из сети Интернет, необходимо добавить правила проброса портов на маршрутизаторе сервера. Обращения из интернета к порту 80, перенаправляются во внутреннюю сеть на устройство 192.168.1.100 и порт 80. Обращения из интернета к порту 443, перенаправляются во внутреннюю сеть на устройство 192.168.1.100 и порт 443.

Настройка маршрутизатора клиента (ROUTER2)

Все настройки проводятся поверх базовой конфигурации маршрутизатора.

1. Задается имя WAN1 интерфейса, устанавливается внешний IP-адрес, указывается стандартный маршрут с метрикой 1.
2. Задается имя WAN2 интерфейса, устанавливается внешний IP-адрес, указывается стандартный маршрут с метрикой 2.

3. В настройках NAT создаются правила маскарадинга для каждого интерфейса

3.4 Реализация защищенных каналов обмена данными

Для доступа к веб-сервису по протоколу HTTPS, необходим сертификат из Центра Сертификации. Let’s Encrypt является одним из таких центров. Перед получением файла сертификата нужно подтвердить право на владение доменом. Для этого существует специальное программное обеспечение - протокол ACME, который запускается на веб-сервере.

В конфигурационный файл nginx вносятся изменения. В нем указывается сертификат, полученный с помощью Let’s Encrypt, объявляется порт SSL 443, объявляется используемая эллиптическая кривая для протоколов обмена ключами, перечисляются наборы шифров и версии SSL/TLS с помощью директив ssl_protocols и ssl_ciphers.

После проведенных операций становится возможно подключение к веб-сервису с помощью метода аутентификации HTTPS.

Реализация методов с использованием VPN-протоколов

В RouterOS настройка VPN-подключения происходит в меню /ppp. Порядок настройки туннеля происходит по определенному сценарию, но с некоторыми отличиями в зависимости от используемого протокола туннелирования. Также для разрешения прохождения трафика определенного типа необходимо создать правила в таблице Firewall маршрутизатора и объявить маршруты для доступа к удаленной сети. На примере протокола L2TP далее приведен алгоритм установки в RouterOS VPN-сервера и клиента.

/ppp/profile

В первую очередь необходимо создать профиль VPN-подключения. Профили используются для определения значений по умолчанию для записей доступа пользователей, хранящихся в подменю /ppp/secret. Ключевыми параметрами профиля является необходимость использования сжатия и стандартного шифрования данных (MPPE-128 для протоколов PPTP и L2TP). Параметр change-tcp-mss позволяет изменять значение MSS в передаваемых

пакетах. Его необходимо включить. В результате протокол TCP будет разбивать поток на сегменты, не превышающие MSS, и передаваемые IP-пакеты не будут требовать фрагментации. Остальные параметры рекомендуется установить по умолчанию.

Следующим этапом является создание интерфейса VPN-сервера. Для каждого протокола туннелирования в данном меню присутствует особый набор параметров. Основные из них - протокол туннелирования, используемый порт, keepalive, тип аутентификации и тип ключа шифрования. Также есть возможность указать сертификат, находящийся в меню system/certificates. После того как произойдет разрыв соединения и по туннелю перестанут идти пакеты, сервер будет отправлять keepalive пакеты в туннель, и, если в течении указанных секунд не будет подтверждения, туннель будет считаться закрытым. Интерфейс установки VPN-сервера показан на рисунке 3.3.

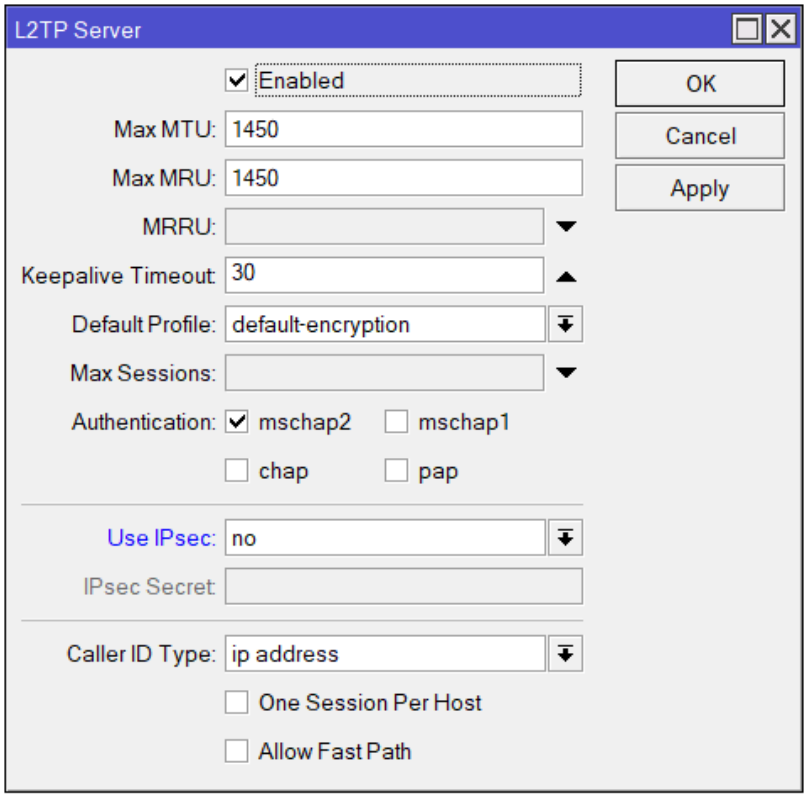

Рисунок 3.3 — Параметры VPN-сервиса в RouterOS

В подменю /ppp/secret необходимо создать аккаунт пользователя. Указывается логин, пароль, профиль и используемый сервис. Также следует указать виртуальный IP-адрес VPN-подключения.

В таблице фильтров Firewall необходимо создать разрешающие правила для конкретных типов протоколов и портов. Данные правила поднимаются вверх в таблице.

Для того, чтобы обе сети могли обмениваться информацией друг с другом между ними должна быть настроена маршрутизация. В настройках маршрута должен быть установлен адрес сети получателя, интерфейс, через который можно получить доступ к сети, то есть виртуальный IP-адрес VPN-подключения, и интерфейс, с которого будут идти запросы к сети получателя. Интерфейс настройки маршрута показан на рисунке 3.4.

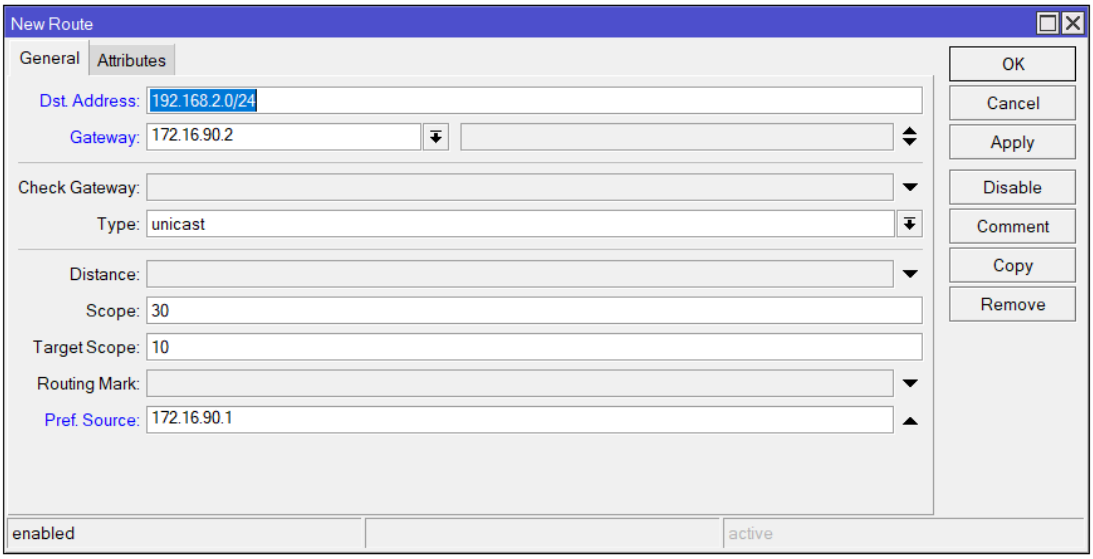

Рисунок 3.4 — Настройка маршрута в RouterOS

Следующий этап — установка VPN-клиента на маршрутизаторе абонента.

В первую очередь необходимо создать интерфейс VPN-клиента. Для каждого протокола туннелирования в данном меню присутствует особый набор параметров. В данном подменю указывается протокол туннелирования, адрес VPN-сервера, логин, пароль, тип аутентификации. При необходимости указывается сертификат клиента. Интерфейс настройки показан на рисунке 3.5.

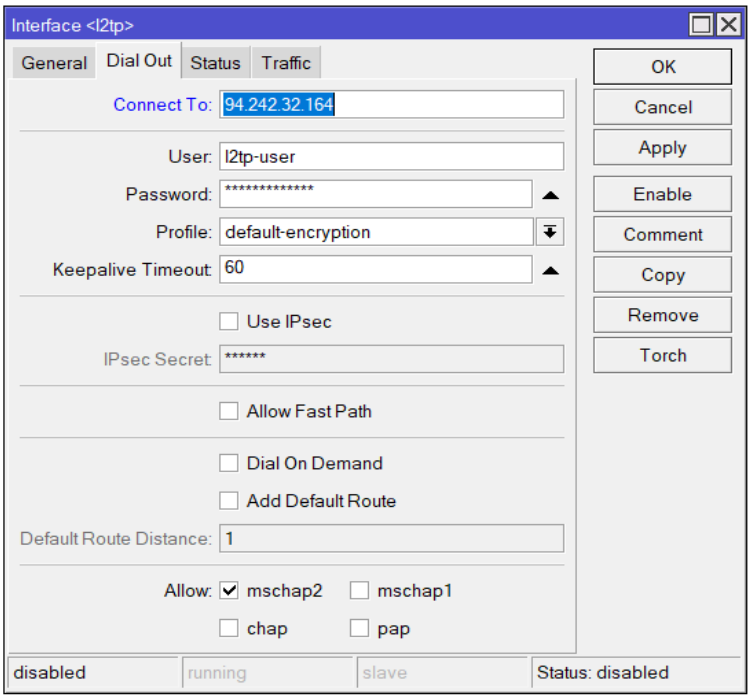

Рисунок 3.5 — Настройка VPN-клиента в RouterOS

Для доступа к сети VPN-сервера также необходимо указать маршрут до нее. Параметры маршрута зеркальны параметрам на сервере.

На данном этапе возможно подключение клиента к серверу по защищенному каналу связи и дальнейший обмен информацией.

Коды конфигураций VPN-туннелей (SSTP, OpenVPN, L2TP) приведены в Приложениях А–В.

Для аутентификации удаленного клиента при использовании некоторых решений с помощью VPN-протоколов потребуется настройка инфраструктуры открытых ключей. Это повышает уровень защищенности канала в сети, в частности от атак активного типа.

Для начала необходимо сгенерировать сертификат для встроенного в RouterOS центра сертификации. Данному сертификату будут доверять все серверы и клиенты. С его помощью будут подписаны клиентские и серверные сертификаты. Важно при генерации правильно указать региональные параметры, длину ключа и срок действия сертификата. Его также нужно подписать. Интерфейс настройки показан на рисунке 3.6.

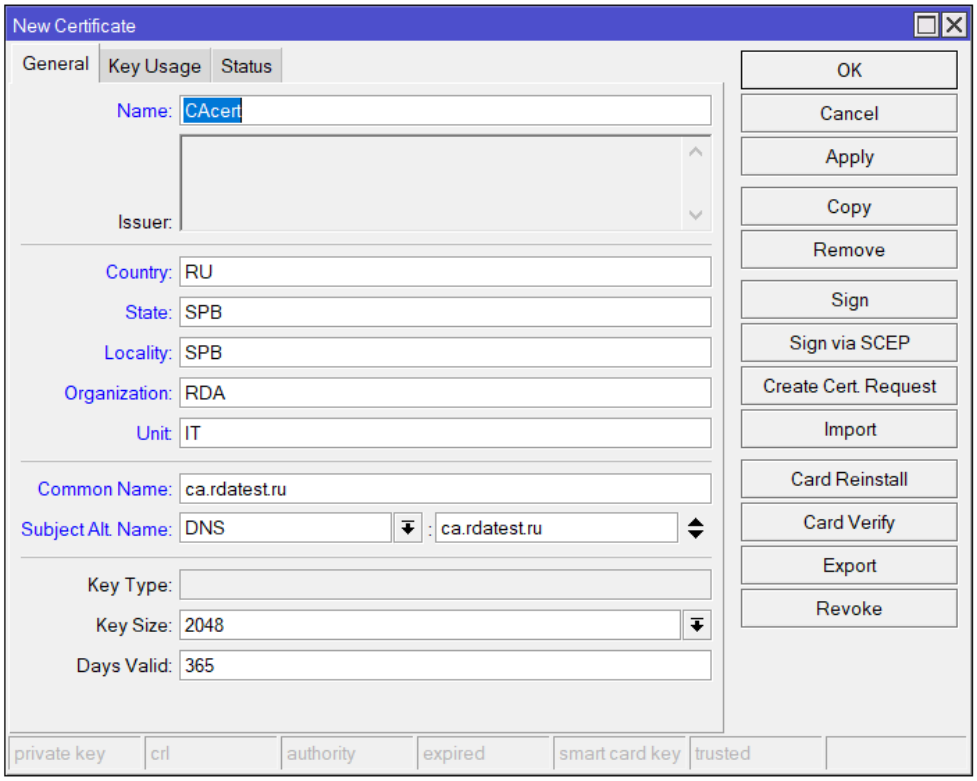

Рисунок 3.6 — Создание сертификата в RouterOS

Следующий этап - создание серверного SSL-сертификата. Таким же образом необходимо указать региональные параметры, длину ключа и срок действия сертификата. Его нужно подписать с помощью сертификата удостоверяющего центра.

Далее создается сертификат клиента. Его нужно подписать с помощью центра сертификации.

Необходимо экспортировать клиентскую подпись с приватным ключом и сложным паролем в файл формата PKCS12. Пароль необходим для расшифровки клиентского сертификата при установке подключения абонента. Файл необходимо скачать с маршрутизатора и отправить абоненту, который в свою очередь установит его в своей операционной системе. PKCS12 также содержит в себе сертификат удостоверяющего центра.

После всех выполненных процедур, может быть установлен защищенный канал связи с помощью сертификатов.

Коды генерации сертификатов приведены в Приложении Д.

Блок IPsec в MikroTik является достаточно большим и функциональным в

RouterOS. Он был сильно модифицирован разработчиками в обновлении v6.44. Новая структура позволяет реализовать любые возможные решения IPsec в транспортном и туннельном режиме.

Алгоритм настройки IPsec в режиме IKEv2 состоит из следующих этапов:

1. Необходимо указать настройки в меню IPsec mode config. Оно отвечает за сетевые настройки, которые получает абонент, подключившись к серверу.

2. В настройках IPsec profile необходимо указать настройки алгоритмов шифрования с помощью которых IPsec устанавливает соединение между клиентом и сервером. Данный процесс является фазой 1.

3. Создание нового IPsec peer на публичном IP-адресе. Фактически на данном этапе производится включение сервера IPsec в режиме IKE2. Необходимо указать ранее созданный профиль фазы 1.

4. Как только стороны обменялись ключами начинается работа фазы 2. Ключи используются для шифрования каждого IP-пакета. Необходимо указать алгоритмы, с помощью которых данная операция будет производится. Все параметры указываются в меню IPsec proposal.

5. Необходимо создать группу политик IPsec policy group.

6. Создание IPsec identity. В данном подменю необходимо указывается пользователь, сертификаты, группы политик, mode config.

7. Для того чтобы разрешить IPsec подключение снаружи, необходимо добавить три разрешающих правила в таблицу Firewall. Необходимо открыть порты UDP500 UDP4500 и разрешить IPsec-ESP.

Настройка IPsec-клиента выполняется практически аналогичным образом, некоторые параметры указываются зеркально. Ключевым моментом является установка SA. Данные параметры должны полностью соответствовать параметрам сервера. Интерфейс настройки параметров SA в RouterOS показан на рисунке 3.7.

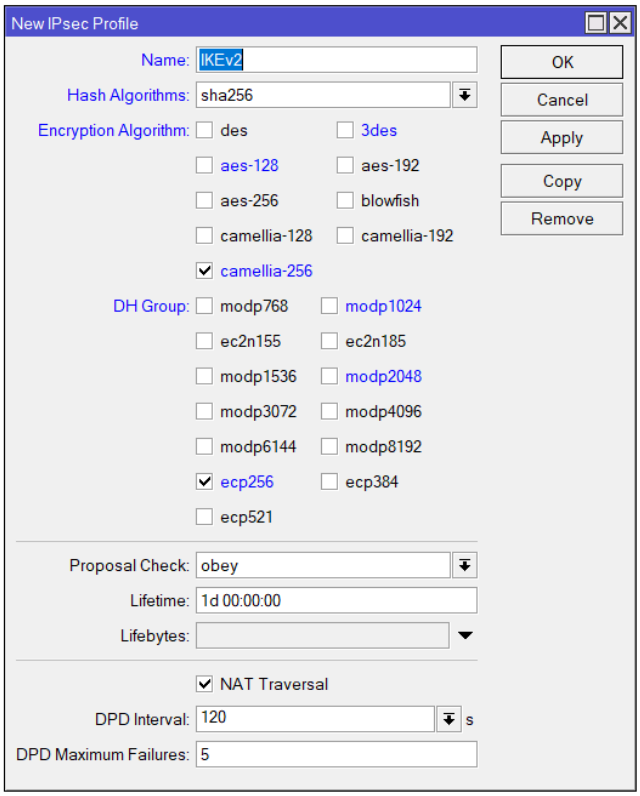

Рисунок 3.7 — Создание сертификата в RouterOS

При настройке IPsec в транспортном режиме, необходимо указать службу L2TP в параметрах политики и выбрать метод аутентификации PSK в identity. Настройки шифрованного канала в данном варранте практически идентичны IKEv2.

Код конфигурации VPN-туннеля IPsec IKEv2 приведен в Приложении Г.

3.5 Разработка программ имитационного моделирования, позволяющих имитировать поведение мобильного абонента в сети Интернет

Во время загрузки файла на клиентском маршрутизаторе ROUTER2 вручную будет производиться запуск программы имитационного моделирования, при этом будут возникать разрывы соединения с последующим его восстановлением через определенное время. IP-адрес во время разрыва может измениться в зависимости от типа программы, причем для абонента данный эффект будет прозрачен.

В системе RouterOS предусмотрен раздел для хранения файлов программ

имитационного моделирования (RouterOS scripting), расположенный в /system/script. Их запуск может быть реализован тремя способами: вручную, по расписанию и при возникновении события. При проведении экспериментов, программы удобнее всего запускать вручную после начала процесса передачи данных.

При проведении экспериментов используются следующие программы имитационного моделирования:

– **disabledW1** Отключение интерфейса WAN1 на 20 секунд. Сразу после отключения, активным становится интерфейс WAN2. При возобновлении работы интерфейса WAN1, трафик движется через его маршрут, благодаря меньшей метрике. IP-адрес сменяется дважды.

– **disabledALLactiveW2** Переключение WAN с временем разрыва 10 секунд. Отключаются два интерфейса одновременно. По истечении времени разрыва активным становится только WAN2. IP-адрес сменяется один раз.

– **disabledALLactiveW1** Отключение WAN с временем разрыва 10 секунд. Отключаются два интерфейса одновременно. По истечении времени разрыва активным становится только WAN1. IP-адрес не изменяется.

Коды программ приведены в Приложении Г.

3.6 Выводы по главе 3

Разработаны методы аутентификации мобильных абонентов для доступа к облачным сервисам на основе TLS и VPN-туннелей. Проведен анализ актуальных данных о криптостойкости параметров методов аутентификации. Предложен список решений, обеспечивающих высокий уровень безопасности информации в процессе обмена по каналу связи, при умеренных требованиях к аппаратным ресурсам системы абонента.

Разработана модель, позволяющая имитировать потоки данных между мобильным абонентом и облачным сервисом по защищенному каналу связи на основе реальной сети с системой маршрутизаторов MikroTik, работающих под управлением операционной системы RouterOS.

Разработаны программы имитационного моделирования, позволяющие

имитировать поведение мобильных абонентов в глобальной сети при помощи встроенного в RouterOS инструмента RouterOS scripting.

Глава 4. Эксперименты и исследования.

4.1 Разработка методики проведения экспериментов

Перед проведением экспериментов необходимо подготовить набор данных, которыми стороны будут обмениваться по защищенному каналу связи. Сервер генерирует набор файлов, располагает их в директориях веб-сервиса и предоставляет к ним доступ мобильному абоненту по соответствующим ссылкам.

Для создания экспериментального файла, необходимо получить установленный объем случайных данных из /dev/urandom и записать их в файл. Провести такую операцию без привлечения дополнительного ПО можно с помощью встроенной утилиты dd:

dd if=/dev/urandom of=file bs=50K count=1 iflag=fullblock

Итоговые данные приведены в таблице 5.

Таблица 5 — Набор данных для проведения экспериментов

Тип данных в экспериментах	**Объем данных**	**Команда генерации dd**
Данные большого объема	100 Мб	dd if=/dev/urandom of=file bs=100M
	1 Гб	dd if=/dev/urandom of=file bs=1000M
Поток данных	50 Кб	dd if=/dev/urandom of=file bs=50K
	75 Кб	dd if=/dev/urandom of=file bs=75K
	100 Кб	dd if=/dev/urandom of=file bs=100K

Для загрузки файлов по протоколу HTTP и HTTPS используется консольная утилита wget. Ее плюс заключается в том, что она имеет возможность автоматической докачки файла после разрыва соединения. Для этого необходимо использовать флаг -c:

wget -c https://rdatest.ru/download/file100

Также в утилите wget присутствует возможность скачивания нескольких файлов одной командой. Для этого необходимо создать текстовый файл,

который будет содержать ссылки на необходимые файлы и в команде wget использовать флаг -i:

wget -i files

Для регистрации всех сообщений в отдельный лог-файл в утилите wget используется флаг -o:

wget https://rdatest.ru/download/file100 -o log

Анализируя информацию лог-файла, можно фиксировать показатели скорости в промежуток времени.

Для контроля загрузки ЦП абонента используется утилита atop. Она способна работать в режиме демона. В данном режиме сбора статистики программа записывает в журнал событий все данные по загрузке системы. Для проведения экспериментов Утилита снимает показания каждую секунду, и записывает их в лог-файл:

atop -a -w /var/log/atop.log 1

Для контроля пакетов и анализа сетевого трафика используется утилита tshark. Прослушивая определенный сетевой интерфейс системы, существует возможность регистрации повторной передачи пакета, что свидетельствует о факте его потери при обмене данными. Это можно реализовать с помощью следующих фильтров:

– tcp.analysis.lost_segment — указывает разрыв в порядковых номерах во время процедуры захвата. Потеря пакета может привести к дублированию ACK, что приводит к повторной передаче.

– tcp.analysis.retransmission — отображает все повторные передачи во время процедуры захвата.

В системе RouterOS инструмент Graphs является средством мониторинга различных параметров роутера MikroTik, также он имеет возможность отображать их на графиках в веб-интерфейсе устройства. Во время проведения экспериментов с помощью данного инструмента будет фиксироваться показатель загрузки центрального процессора маршрутизатора в момент времени. Затем данные значения экспортируются в файл и представляются на

итоговых графиках. Для включения режима мониторинга центрального процессора в инструменте Graphs необходимо активировать параметр CPU load.

4.2 Проведение экспериментов

При реализации на модели методов аутентификации на основе VPN используется протокол передачи данных HTTP без TLS во избежание двойного шифрования, которое негативно влияет на качество обмена данными и усиливает нагрузку на аппаратные ресурсы сторон обмена, при этом не дает серьезного увеличения криптостойкости метода. Эксперименты при реализации защищенного канала на основе HTTPS проводились отдельно.

Эксперименты также проводились без реализации защищенного канала передачи данных (только при работе протокола HTTP) для получения эталонных значений в каждом эксперименте.

Для измерения скорости обмена данными в канале связи в качестве передаваемой информации выступает файл объемом 1Гб, наполненный случайными данными. Показатели скорости регистрируются в системе в процессе передачи данных в интервале 30 секунд. Был проведен ряд экспериментов (10 подходов измерений), для графиков были выбраны интервалы с наибольшей стабильностью показателя скорости каждого метода аутентификации. Результаты измерений показаны на рисунке 4.1.

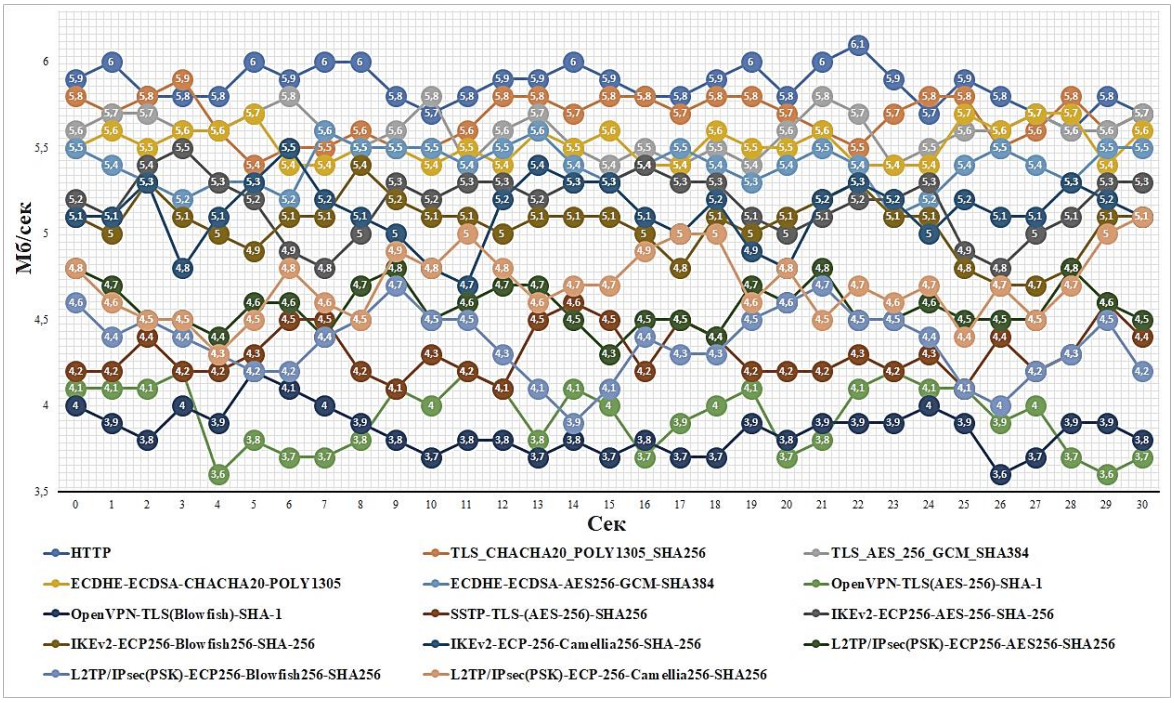

Рисунок 4.1 — Графики скорости в канале обмена данными при различных методах аутентификации

На некоторых графиках прослеживаются частые преломления, подъемы и спуски кривой скорости, в частности при реализации методов аутентификации на основе VPN-туннелей L2TP/IPsec и SSTP. Данные преломления вызваны накладными расходами в процессе инкапсуляции трафика совместно с процедурой шифрования. На основании этого можно сделать вывод о меньшей стабильности скорости передачи данных при этих методах, причем наименьшую стабильность показала реализация на основе метода L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256. Если среда обмена данными неустойчива, то использование такого метода аутентификации может негативно сказаться на качестве обмена.

Для данных интервалов были вычислены значения средней скорости и зафиксированы значения пропускной способности.

Графики средней скорости передачи данных при различных методах аутентификации показаны на рисунке 4.2.

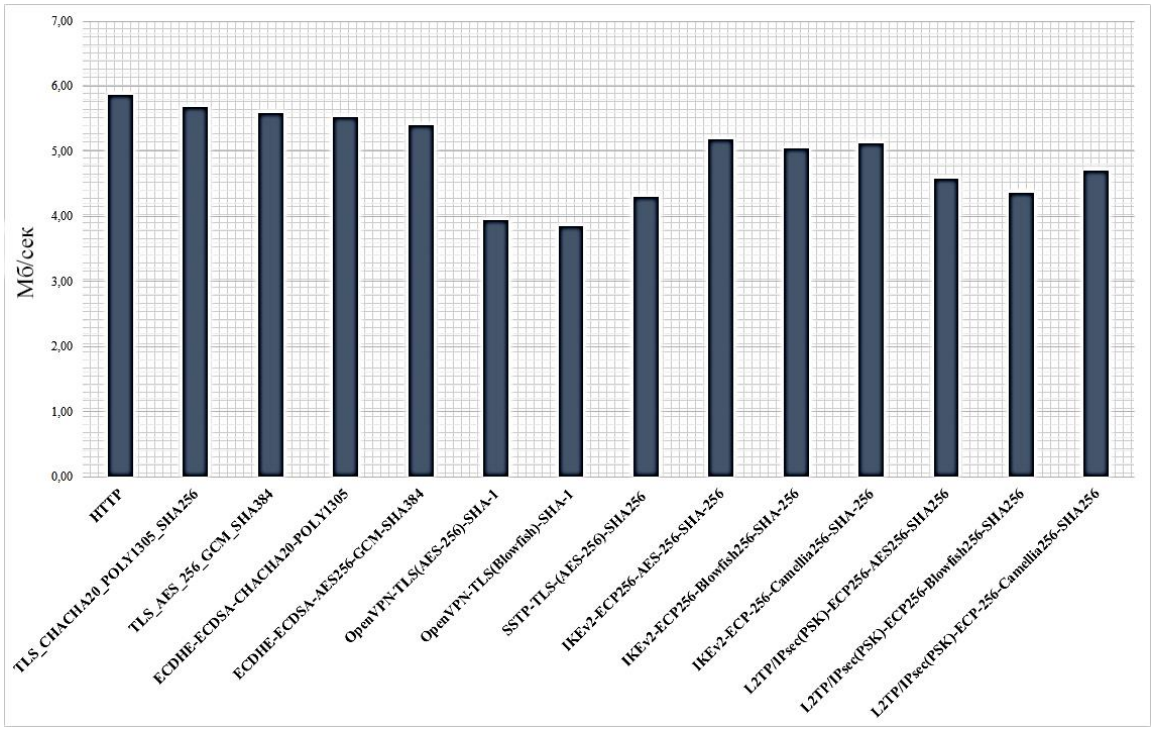

Рисунок 4.2 — Графики средней скорости передачи данных при различных методах аутентификации

Графики пропускной способности канала обмена данными при различных методах аутентификации показаны на рисунке 4.3.

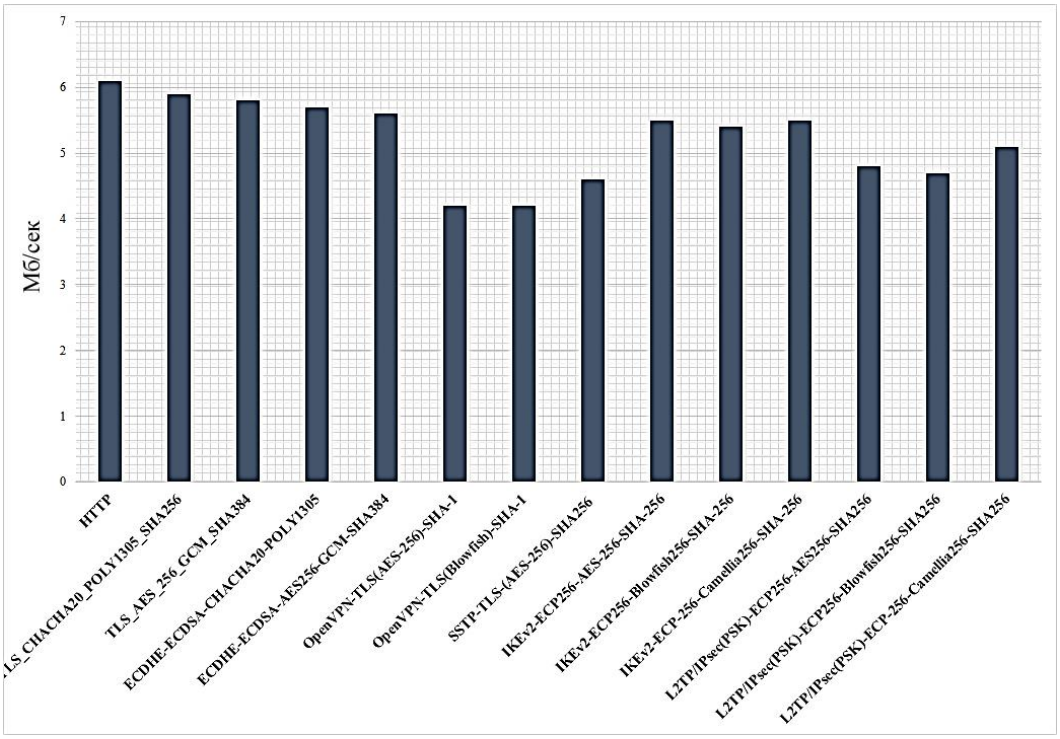

Рисунок 4.3 — Графики пропускной способности канала обмена данными при различных методах аутентификации

Значение средней скорости и пропускной способности канала связи тесно взаимосвязаны между собой, поэтому результаты проведенных экспериментов можно рассматривать совместно. Наилучшие показатели при проведении экспериментов показали методы на основе решений HTTPS (потери скорости составляют 2-6%), они используют практически всю полосу пропускания при использовании любых криптографических алгоритмов с минимальными отличиями между собой на уровне погрешностей. Хорошие показатели также видны в реализациях на основе методов IKEv2 (потери скорости составляют 10%). Значительное снижение средней скорости и пропускной способности канала связи наблюдается в реализациях на основе методов OpenVPN и SSTP (потери скорости составляют 30% и 25% соответственно). Если скорость является ключевым параметром в системе обмена данными, то данные методы

аутентификации не рекомендуются к применению.

Принцип проведения эксперимента по измерению скорости обмена данными в канале связи для потока данных отличается от предыдущего, так как фиксировать скорость на малых объемах данных проблематично. Удобнее оперировать показателем времени передачи потока данных и рассчитать среднюю скорость исходя из данной величины. В качестве передаваемых данных выступает массив из 150 элементов объемом 50Кб, 75Кб и 100Кб, наполненные случайными данными. Время скачивания такого потока данных при каждом методе аутентификации показан на рисунке 4.4.

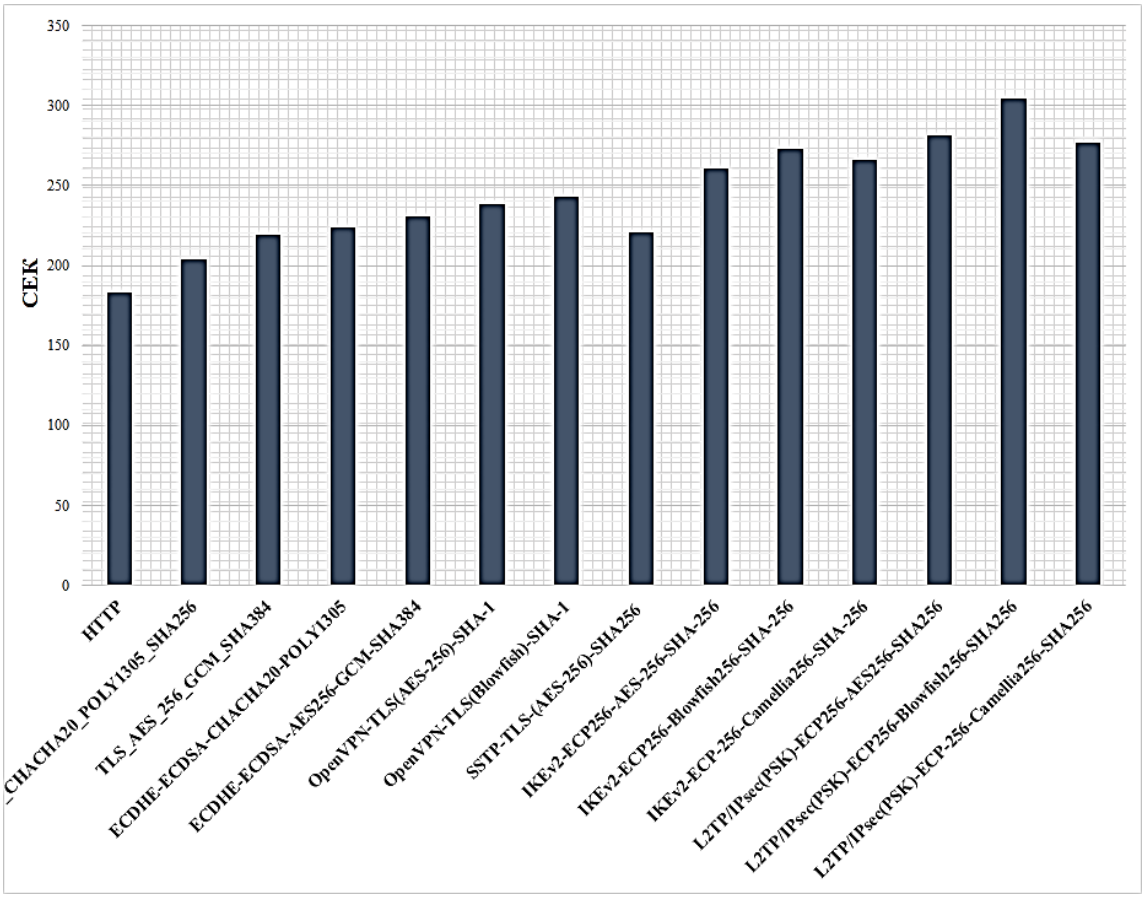

Рисунок 4.4 — Время скачивания потока данных при различных методах аутентификации

Для потока данных также были рассчитаны значения средней скорости в заданном интервале. Графики средней скорости потока данных при различных методах аутентификации показаны на рисунке 4.5.

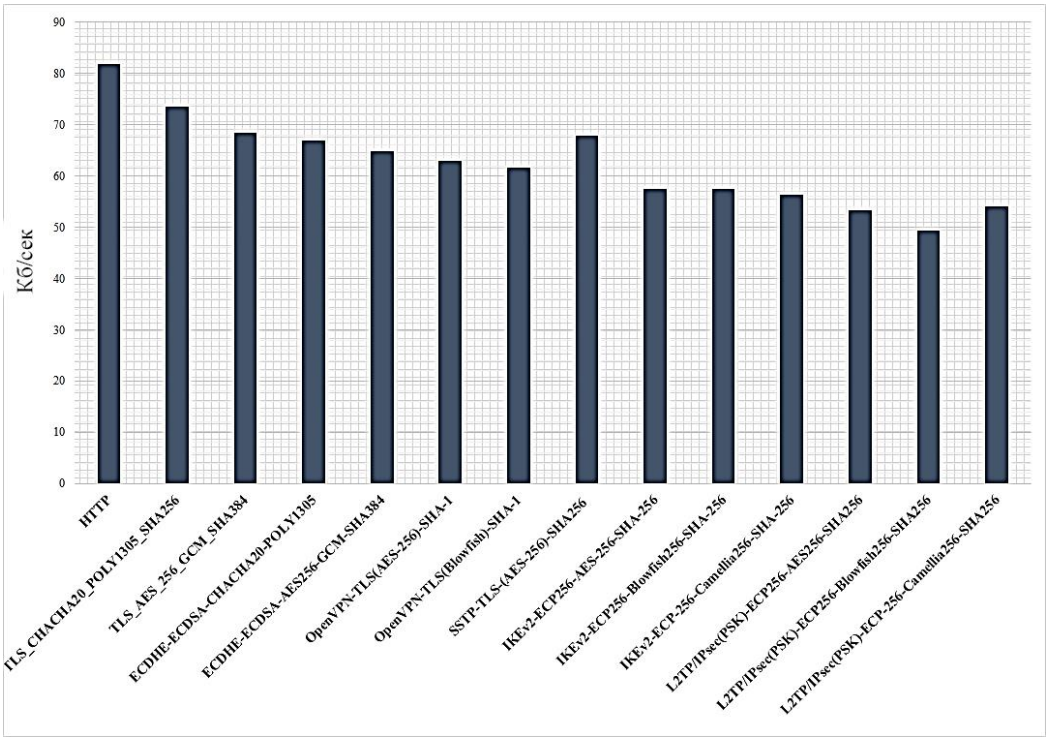

Рисунок 4.5 — Средняя скорость передачи потока данных при различных методах аутентификации

На полученных графиках заметно снижение показателя средней скорости при использовании методов аутентификации на основе VPN. Необходимо также отметить хорошие показатели при реализации на основе протокола SSTP (потери скорости составляют 15%), в отличие от предыдущего эксперимента, где данный протокол показал самые низкие результаты. Разница в значениях минимальна при использовании различных криптографических алгоритмов на базе одного протокола.

Во время проведения экспериментов по определению параметра отказоустойчивости используются программы имитационного моделирования, с помощью которых оказано воздействие на физические интерфейсы маршрутизатора абонента. В качестве в качестве передаваемой информации выступает файл объемом 1Гб, наполненный случайными данными. Для проведения экспериментов был выбран временной интервал процесса передачи данных длиной 30 секунд. Во временном интервале с 5 по 15 секунду интерфейсы находятся в отключенном состоянии. Был проведен ряд

экспериментов (10), для конечных результатов избраны лучшие временные показатели. На графиках, изображенных на рисунке 4.6, показана зависимость скорости передачи данных от времени при использовании того же интерфейса, что и до разрыва соединения.

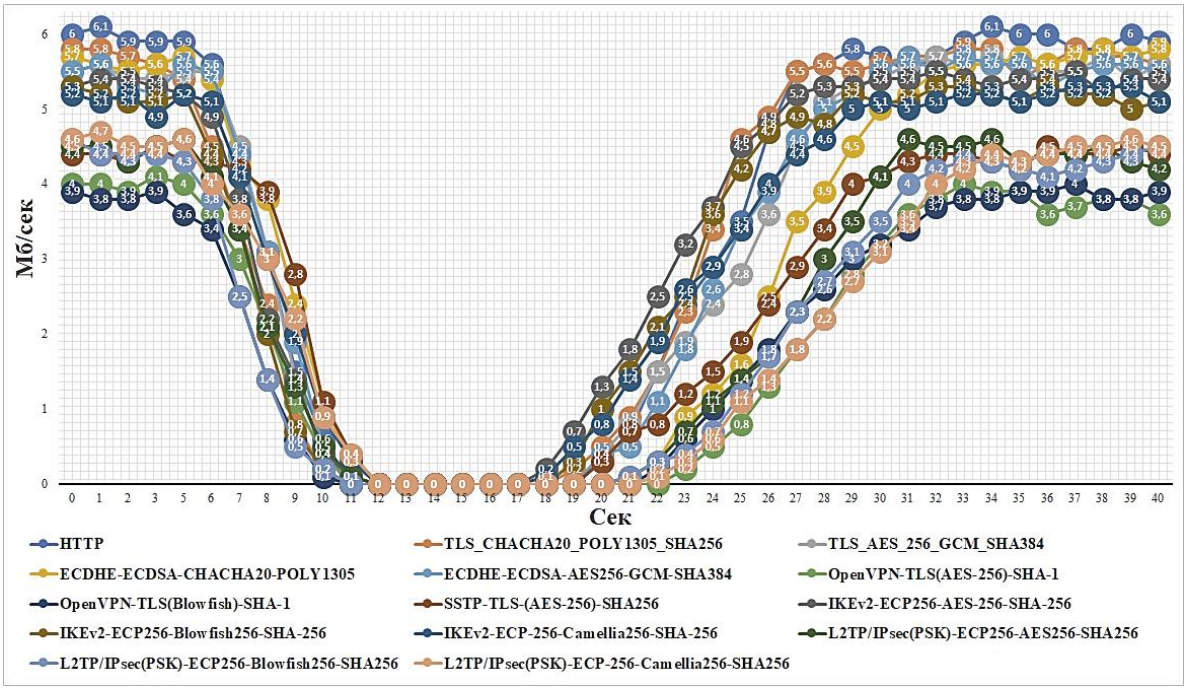

Рисунок 4.6 — Графики скорости в эксперименте по определению параметра отказоустойчивости при использовании одного интерфейса

На графиках, изображенных на рисунках 4.7 показана зависимость скорости передачи данных от времени при смене физического интерфейса с последующей сменой IP-адреса.

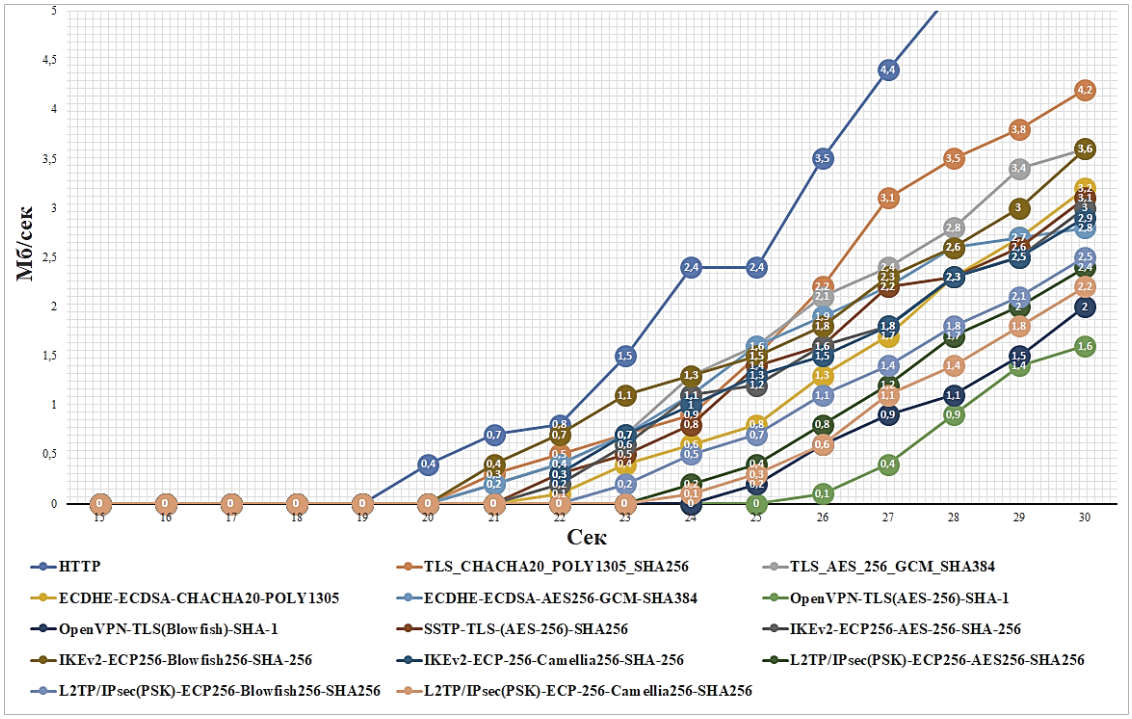

Рисунок 4.7 — Графики скорости в эксперименте по определению параметра отказоустойчивости при использовании двух интерфейсов

Фиксируя показатели скорости передачи данных в данных экспериментах при различных реализациях, определяется время восстановления сеанса связи в защищенном канале обмена данными. Оно соответствует временному интервалу между включением физического интерфейса маршрутизатора и временем получения первого пакета потока данных. Результаты расчетов для одного интерфейса показаны на рисунке 4.8.

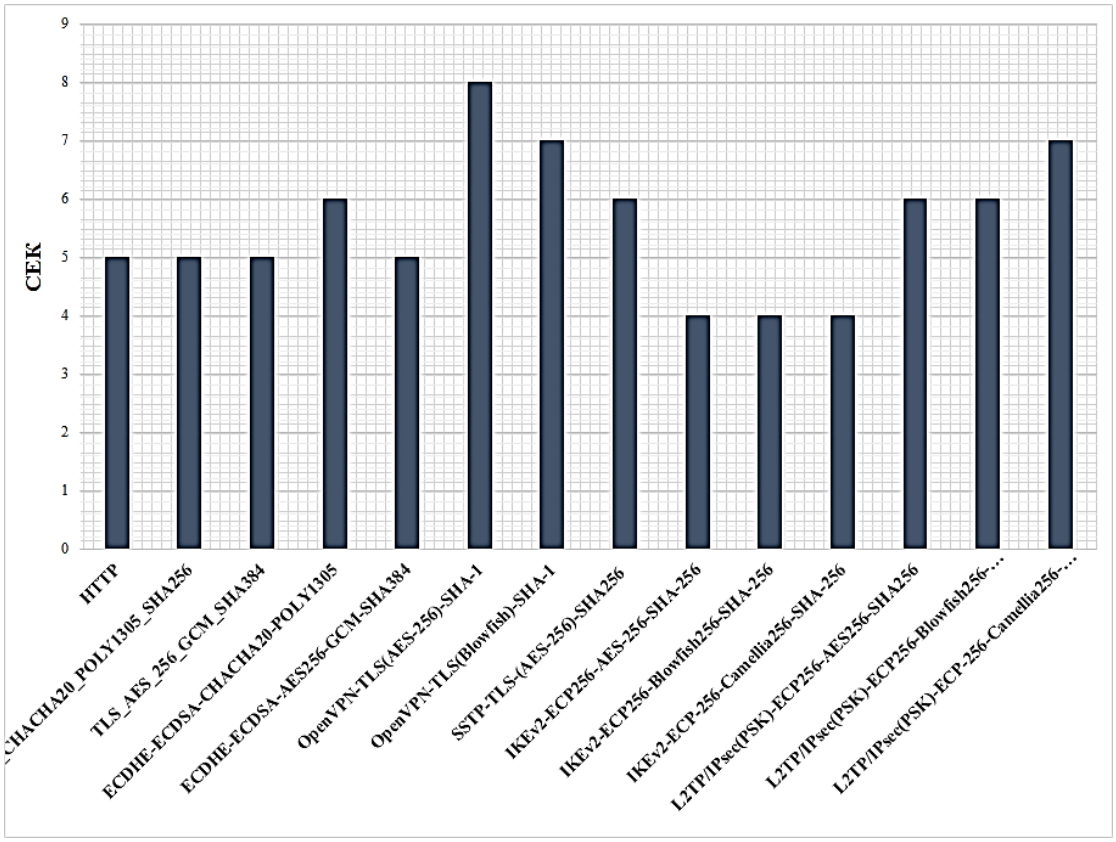

Рисунок 4.8 — Время восстановления сеанса связи при использовании одного интерфейса

Результаты расчетов характеристики времени восстановления сеанса связи для двух интерфейсов показаны на рисунке 4.9.

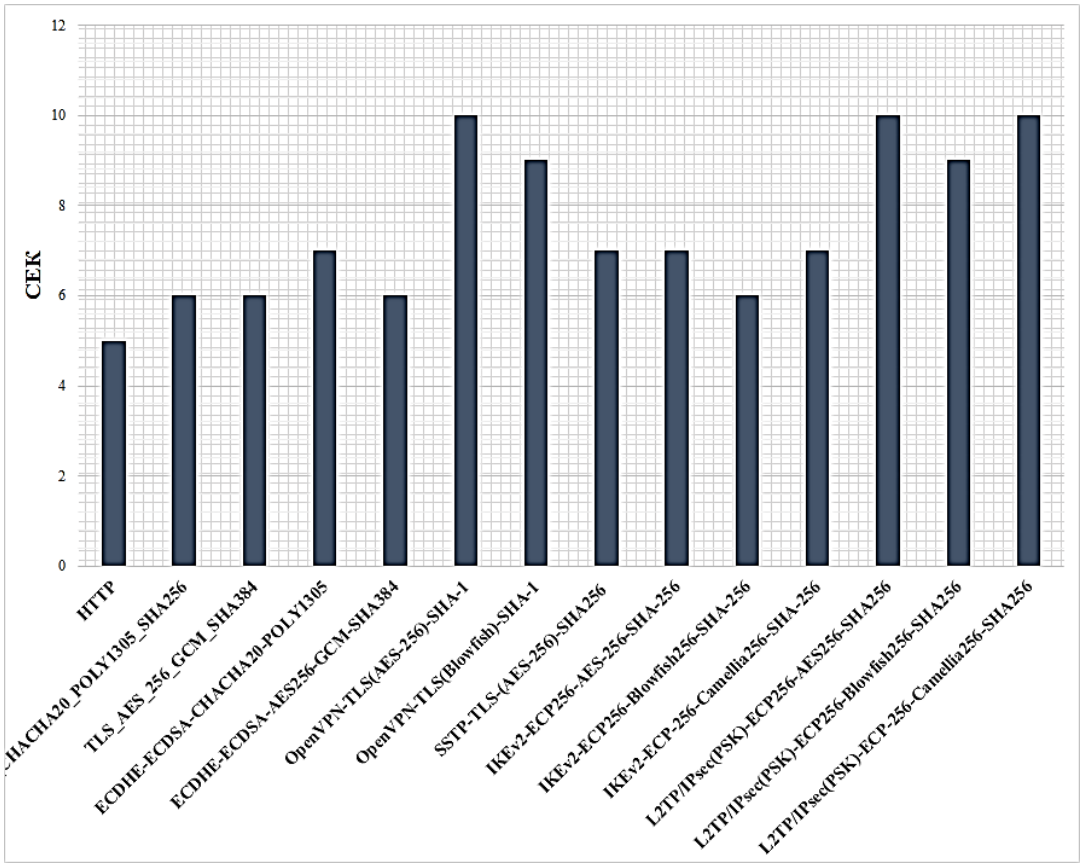

Рисунок 4.9 — Время восстановления сеанса связи при использовании двух интерфейсов

На полученных графиках видно, что использование нескольких физических интерфейсов передачи данных увеличивает время восстановления сеанса связи. Однако, полученные временные значения пропорциональны значениям в предыдущем эксперименте с некоторой погрешностью. Поэтому результаты экспериментов по определению времени восстановления сеанса связи при использовании одного интерфейса и двух интерфейсов во время процедуры анализа будут рассматриваться совместно друг с другом. Методы аутентификации HTTPS показывают лучшие результаты среди всех методов вне зависимости от набора шифров (+25% ко времени восстановления сеанса). Наибольшее время восстановление сеанса связи показала реализация на основе OpenVPN и L2TP/IPsec (+100% ко времени восстановления сеанса), лучший показатель среди VPN-туннелей отмечен у метода аутентификации SSTP (+25% ко времени восстановления сеанса).

Времени восстановления сеанса связи соответствует время повторной аутентификации абонентов при организации защищенных каналов обмена данными.

При проведении экспериментов над методами аутентификации HTTPS показатели нагрузки фиксируются на центральном процессоре абонента. При использовании протоколов туннелирования показатели нагрузки фиксируются на центральном процессоре маршрутизатора клиента ROUTER2, так как данные решения реализованы на его аппаратной основе. В качестве передаваемой информации выступает файл объемом 1Гб, наполненный случайными данными.

На рисунке 4.10 показаны графики загрузки центрального процессора маршрутизатора абонента ROUTER2 в процессе передачи данных в интервале 30 секунд. Для графиков были выбраны интервалы с наибольшей стабильностью показателя загрузки ЦП каждого метода аутентификации.

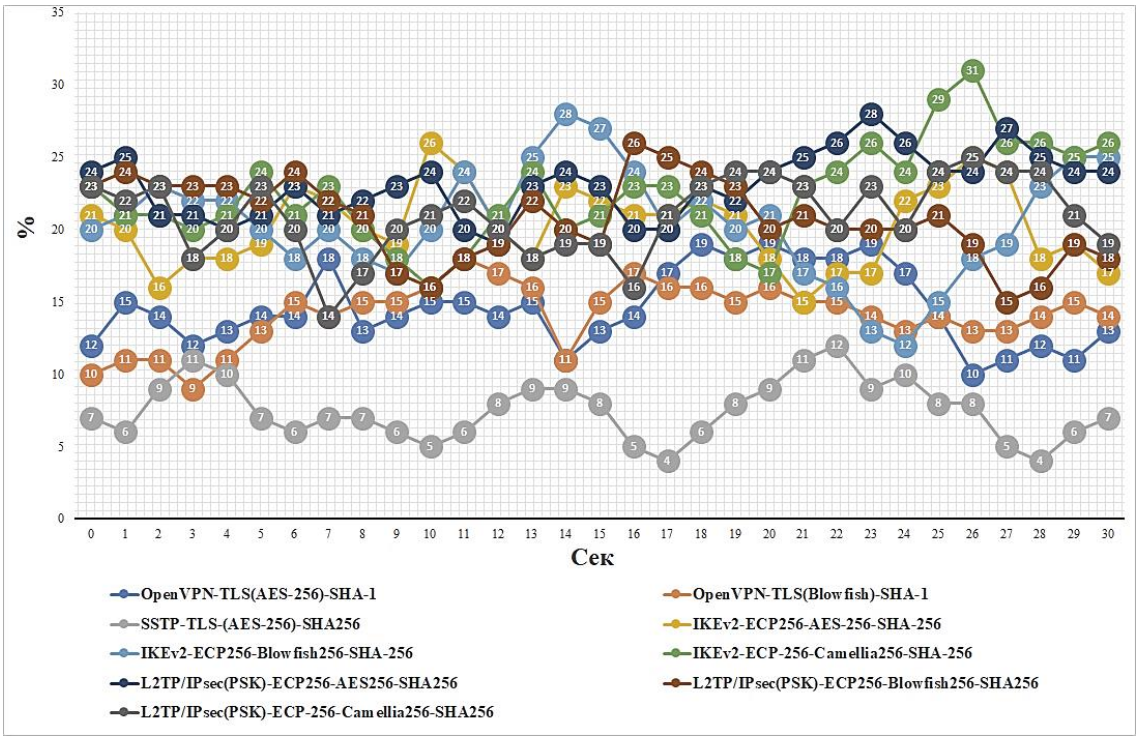

Рисунок 4.10 — Графики загрузки центрального процессора (VPN)

Для данной характеристики также рассчитывается среднее значение. Результаты показаны на рисунке 4.11.

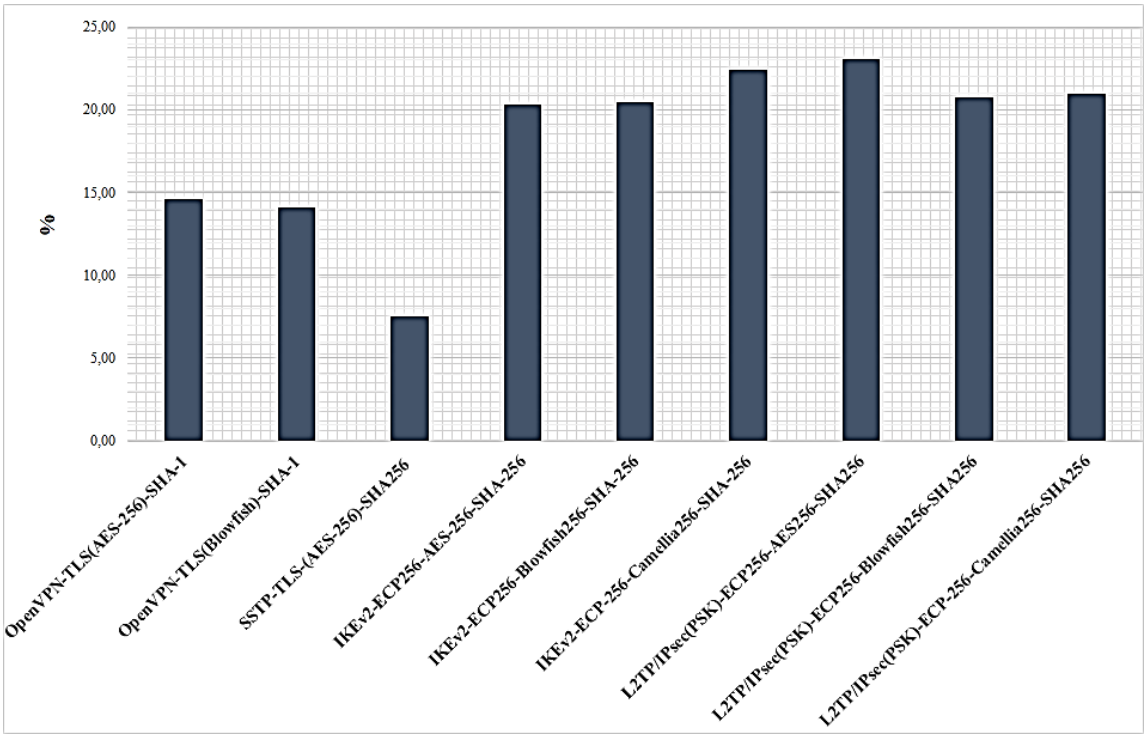

Рисунок 4.11 — Среднее значение загрузки центрального процессора (VPN)

На графиках прослеживается значительный перевес процента загрузки центрального процесса при использовании методов аутентификации, включающих протоколы IPsec (+15% по сравнению с лучшим результатом на основе SSTP). Следовательно, для систем с низкой производительностью данные методы аутентификации будут непригодны. Лучшим выбором в таком случае будет метод аутентификации на основе SSTP (загрузка ЦП — 7%), так как он демонстрирует самые низкие показатели загрузки ЦП в экспериментах.

Необходимо также зафиксировать показатели загрузки центрального процессора мобильного абонента при использовании методов аутентификации HTTPS в процессе передачи данных в интервале 30 секунд. Проведен ряд экспериментов (10 подходов измерений), для графиков были выбраны интервалы с наибольшей стабильностью показателя загрузки ЦП каждого метода аутентификации. Графики загрузки ЦП абонента показаны на рисунке 4.12

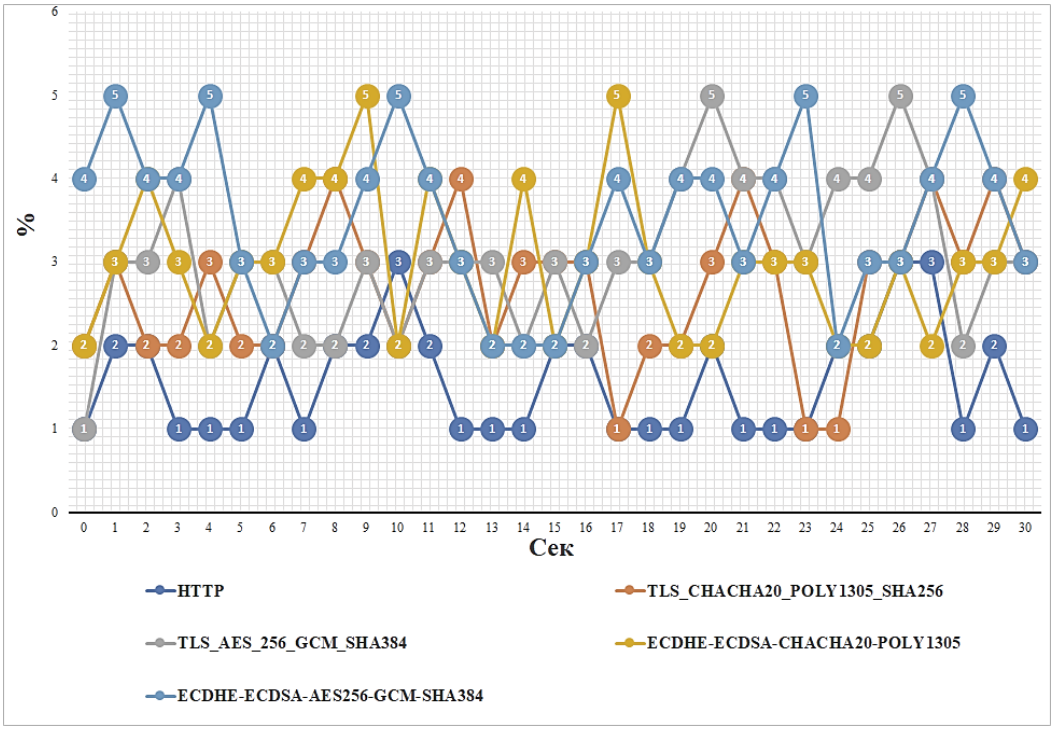

Рисунок 4.12 — Графики загрузки ЦП абонента (HTTPS)

Для полученных данных также рассчитывается среднее значение в данном временном интервале по такому же принципу, как и при использовании VPN-туннелей. Результаты расчетов показаны на рисунке 4.13.

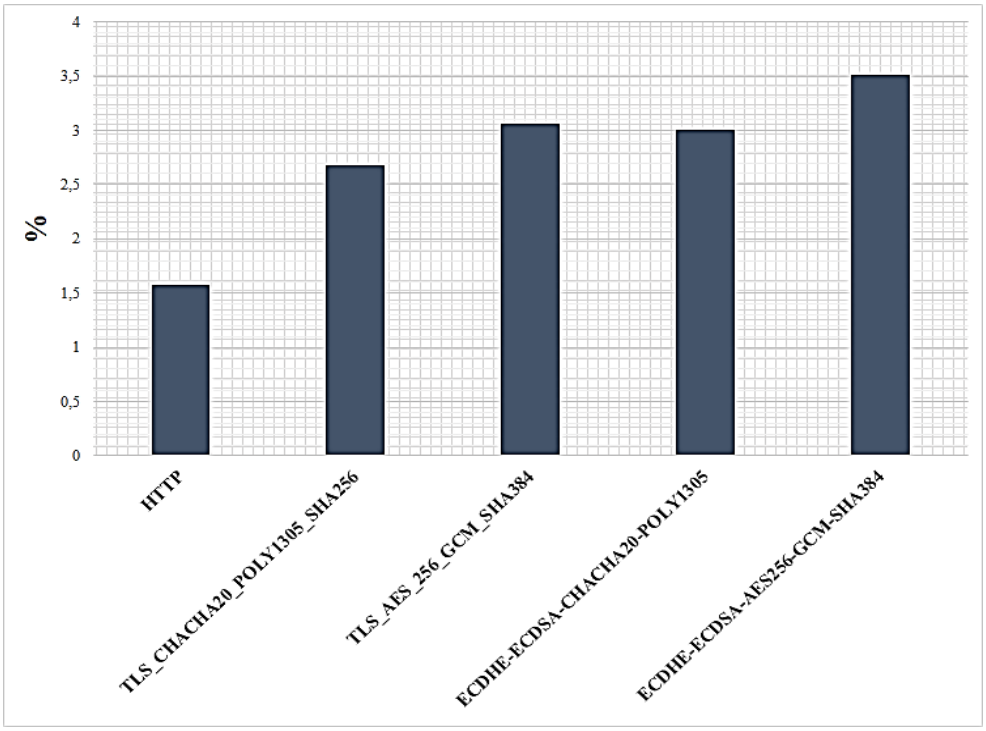

Рисунок 4.13 — Среднее значение загрузки центрального процессора (HTTPS)

По результатам экспериментов можно сделать вывод о том, что все методы аутентификации на основе решения HTTPS не предъявляют высоких требований к ресурсам системы (загрузка ЦП +2% от HTTP). Все методы показали приблизительно одинаковые результаты, разницу в 1% можно считать минимальной погрешностью.

Для расчета потерь пакетов, инициируется передача файла объемом 10Мб. В системе автоматически регистрируется количество повторных передач пакетов, что свидетельствует о факте их потери на пути следования и рассчитывается соотношение потерянных пакетов к общему числу пактов. Полученный процент потерянных пакетов в процессе обмена данными показано на рисунке 4.14.

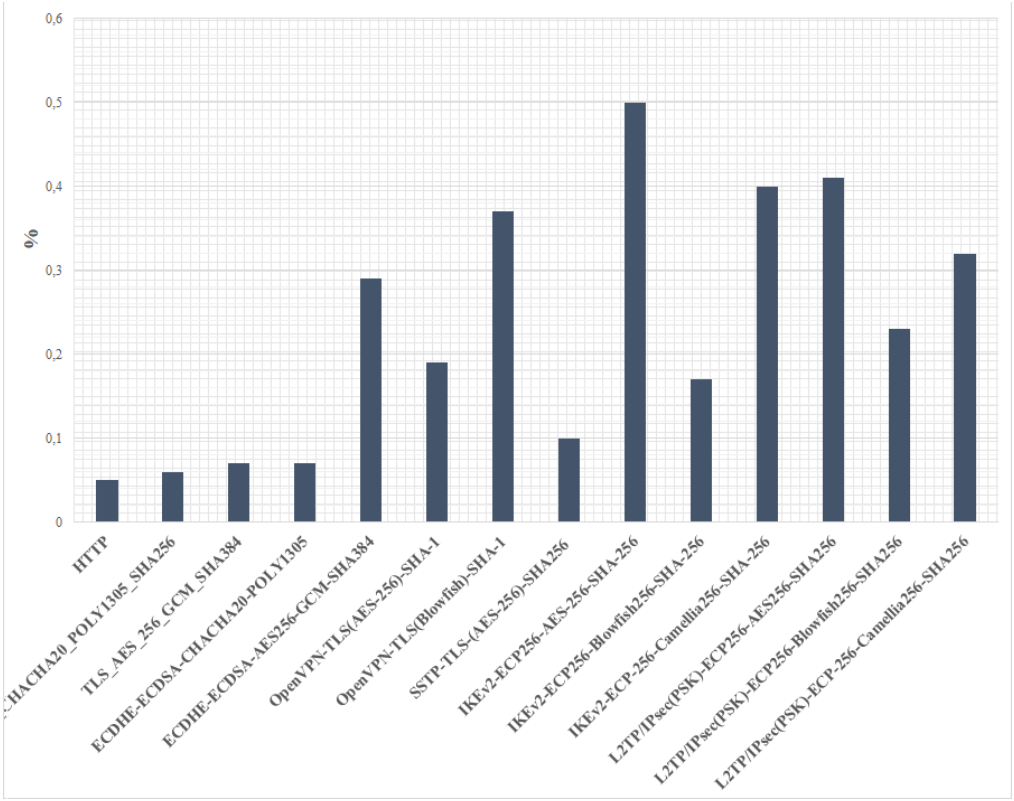

Рисунок 4.14. — Процент потерь пакетов в процессе обмена данными при различных методах аутентификации

Для расчета задержек передачи пакетов в канале передачи данных использовался метод отправки ICMP-пакетов равной длины. Проведен ряд экспериментов (10 подходов измерений) по определению задержек и выбран

временной интервал в 30 секунд с максимальной стабильностью. Графики задержек пакетов при различных методах аутентификации показаны на рисунке 4.15.

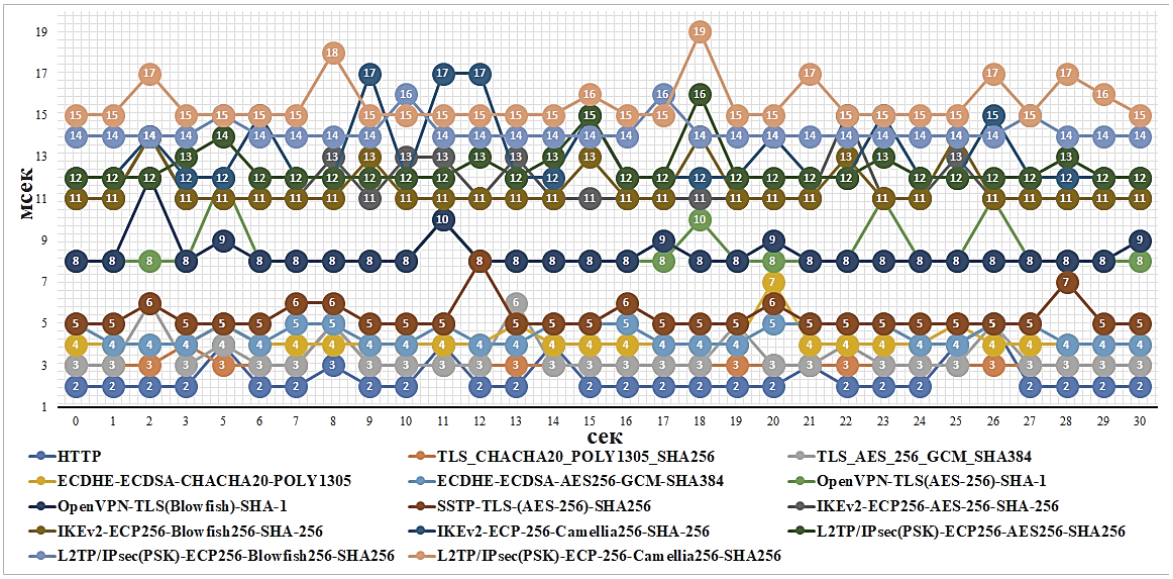

Рисунок 4.15 — Задержки передачи пакетов при различных методах аутентификации

На кривых заметны редкие преломления в сторону увеличения времени задержки, поскольку на поток данных оказано воздействие множеством факторов на пусти следования пакетов на промежуточных узлах от отправителя к получателю, в остальном показатели времени задержки передачи пакетов определены вокруг одного значения.

Для данной характеристики также рассчитывается среднее значение в данном интервале. График средних задержек передачи пакетов при различных методах аутентификации показан на рисунке 4.16.

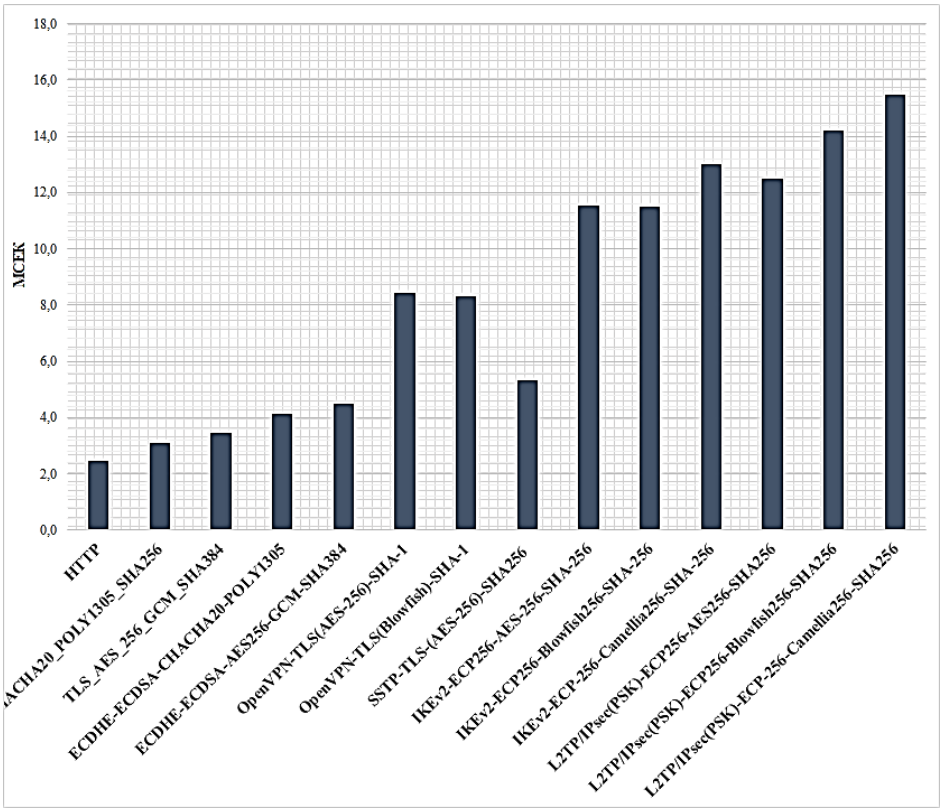

Рисунок 4.16 — Средние задержки передачи пакетов при различных методах аутентификации

На данных графиках прослеживается высокий рост показателей задержек в реализациях на основе IPsec (+700% ко времени задержки). Для организации процесса обмена динамическими данными, данный метод аутентификации не является пригодным.

На данном этапе экспериментальная часть работы закончена. Далее необходимо проанализировать результаты экспериментов и разработать таблицы решений, позволяющие определить лучшие методы аутентификации для каждого сценария обмена данными.

4.3 Анализ результатов экспериментов. Таблицы решений

На первом этапе анализа была проведена сортировка методов аутентификации по каждой характеристике в порядке убывания, от лучшего к худшему, в зависимости от полученных результатов экспериментов. На рисунке 4.17 показаны полученные группы в результате сортирровки.

	Средняя задержка передачи пакета		Пропускная способность канала обмена данными
1	TLS_CHACHA20_POLY1305_SHA256	1	TLS_CHACHA20_POLY1305_SHA256
2	ECDHE-ECDSA-CHACHA20-POLY1305	2	TLS_AES_256_GCM_SHA384
3	TLS_AES_256_GCM_SHA384	3	ECDHE-ECDSA-CHACHA20-POLY1305
4	ECDHE-ECDSA-AES256-GCM-SHA384	4	ECDHE-ECDSA-AES256-GCM-SHA384
5	SSTP-TLS(AES-256)-SHA256	5	IKEv2-ECP256-AES-256-SHA-256
6	OpenVPN-TLS(Blowfish)-SHA-1	6	IKEv2-ECP-256-Camellia256-SHA-256
7	OpenVPN-TLS(AES-256)-SHA-1	7	IKEv2-ECP256-Blowfish256-SHA-256
8	IKEv2-ECP256-Blowfish256-SHA-256	8	L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256
9	IKEv2-ECP256-AES-256-SHA-256	9	L2TP/IPsec(PSK)-ECP256-AES256-SHA256
10	L2TP/IPsec(PSK)-ECP256-AES256-SHA256	10	L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256
11	IKEv2-ECP-256-Camellia256-SHA-256	11	SSTP-TLS(AES-256)-SHA256
12	L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256	12	OpenVPN-TLS(AES-256)-SHA-1
13	L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	13	OpenVPN-TLS(Blowfish)-SHA-1
	Средняя скорость передачи потока данных		Загрузка ЦП
1	TLS_CHACHA20_POLY1305_SHA256	1	TLS_CHACHA20_POLY1305_SHA256
2	TLS_AES_256_GCM_SHA384	2	ECDHE-ECDSA-CHACHA20-POLY1305
3	SSTP-TLS(AES-256)-SHA256	3	TLS_AES_256_GCM_SHA384
4	ECDHE-ECDSA-CHACHA20-POLY1305	4	ECDHE-ECDSA-AES256-GCM-SHA384
5	ECDHE-ECDSA-AES256-GCM-SHA384	5	SSTP-TLS(AES-256)-SHA256
6	OpenVPN-TLS(AES-256)-SHA-1	6	OpenVPN-TLS(Blowfish)-SHA-1
7	OpenVPN-TLS(Blowfish)-SHA-1	7	OpenVPN-TLS(AES-256)-SHA-1
8	IKEv2-ECP256-AES-256-SHA-256	8	IKEv2-ECP256-AES-256-SHA-256
9	IKEv2-ECP256-Blowfish256-SHA-256	9	IKEv2-ECP256-Blowfish256-SHA-256
10	IKEv2-ECP-256-Camellia256-SHA-256	10	L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256
11	L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	11	L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256
12	L2TP/IPsec(PSK)-ECP256-AES256-SHA256	12	IKEv2-ECP-256-Camellia256-SHA-256
13	L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256	13	L2TP/IPsec(PSK)-ECP256-AES256-SHA256
	Время восстановления сеанса связи		Коэффициент потерь пакетов
1	TLS_CHACHA20_POLY1305_SHA256	1	TLS_CHACHA20_POLY1305_SHA256
2	TLS_AES_256_GCM_SHA384	2	ECDHE-ECDSA-CHACHA20-POLY1305
3	ECDHE-ECDSA-AES256-GCM-SHA384	3	TLS_AES_256_GCM_SHA384
4	IKEv2-ECP256-Blowfish256-SHA-256	4	ECDHE-ECDSA-AES256-GCM-SHA384
5	ECDHE-ECDSA-CHACHA20-POLY1305	5	SSTP-TLS(AES-256)-SHA256
6	SSTP-TLS(AES-256)-SHA256	6	IKEv2-ECP256-Blowfish256-SHA-256
7	IKEv2-ECP256-AES-256-SHA-256	7	OpenVPN-TLS(AES-256)-SHA-1
8	IKEv2-ECP-256-Camellia256-SHA-256	8	L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256
9	OpenVPN-TLS(Blowfish)-SHA-1	9	L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256
10	L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256	10	OpenVPN-TLS(Blowfish)-SHA-1
11	OpenVPN-TLS(AES-256)-SHA-1	11	IKEv2-ECP-256-Camellia256-SHA-256
12	L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	12	L2TP/IPsec(PSK)-ECP256-AES256-SHA256
13	L2TP/IPsec(PSK)-ECP256-AES256-SHA256	13	IKEv2-ECP256-AES-256-SHA-256

Рисунок 4.17 — Результаты проведенных экспериментов

На следующем этапе, для каждой группы сценариев X/x составляется список методов аутентификации, в котором методы сортируются на основании результатов экспериментов по каждому элементу X и x:

– Распределение мест для методов аутентификации в группе S/s (характер движения) осуществляется по результатам расчетов среднего времени повторной аутентификации (ниже среднее время повторной аутентификации — S) и процентов потерь пакетов (ниже процент — S)

– Распределение мест для методов аутентификации в группе P/p

(аппаратные ресурсы) осуществляется по результатам эксперимента по определению параметра загрузки центрального процессора системы (ниже средней показатель загрузки ЦП — P).

– Распределение мест для методов аутентификации в группе D/d (тип данных) осуществляется по результатам расчетов средней задержки пакета (ниже показатель средней задержки пакета — d), расчетов средней скорости передачи потока данных (выше средняя скорость – D), расчетов параметров пропускной способности канала передачи данных (выше пропускная способность канала передачи данных — D).

– Распределение мест для методов аутентификации в группе E/e (Класс защищенности данных) осуществляется по показателю криптостойкости методов аутентификации (выше криптостойкость — E).

Затем в данных списках методы аутентификации разделяются на две группы оценок: методы, показывающие лучшие результаты при данном условии — Y, методы, показывающие худшие результаты — N, по каждому элементу X и x. Полученные данные будут являться исходными для таблиц решений. Группа исходных данных показаны на рисунке 4.18.

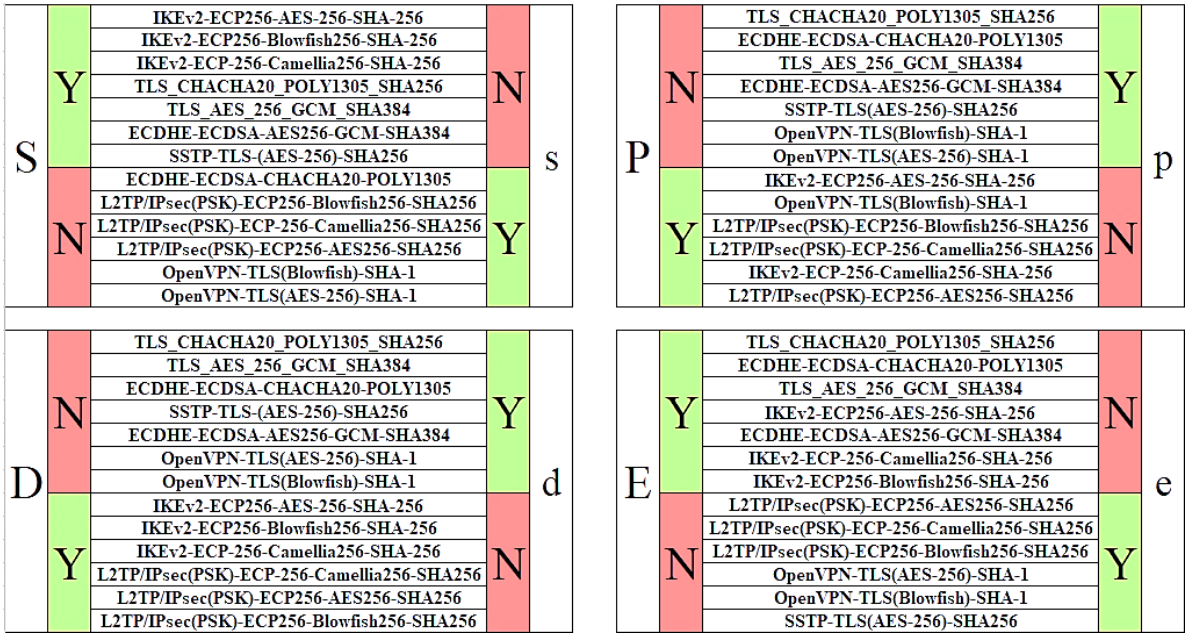

Рисунок 4.18 — Группы исходных данных для таблиц решений

Для составления итоговых таблиц решений предлагается следующий

агоритм:

1) В подготовленной таблице зависимости сценарий-метод, для каждого метода аутентификации определяется промежуточная оценка вида YYNY.

2) Выделяются три группы сортировки методов аутентификации: рекомендуемые к использованию методы ($Y \geq 3$), использование метода допустимо ($Y = 2$), нерекомендуемые к использованию методы ($Y \leq 1$).

3) Создаются ранжированные таблицы решений по каждому параметру сценариев. В группе рекомендуемых к использованию и допустимых методов аутентификации распределяются места. При распределении мест, главным образом учитывается количество положительных элементов Y в оценке метода. При равном количестве элементов Y, места распределяются в зависимости от ранга метода в таблице исходных данных по данному параметру сценариев. Методы, нерекомендуемые к использованию получают оценку 0.

Процесс расстановки промежуточных оценок и определения групп рекомендуемых к использованию методов аутентификации показан на рисунке 4.19.

	SPDE	SPDe	SPdE	SpDE	SPde	SpdE	SpDe	Spde	sPDE	sPDe	sPdE	spDE	sPde	spDe	spdE	spde
TLS_CHACHA20_POLY1305_SHA256	YNNY	YNNN	YNYY	YYNY	YNYN	YYYY	YYNN	YYYN	NNNY	NNNN	NNYY	NYNY	NNYN	NYNN	NYYY	NYYN
TLS_AES_256_GCM_SHA384	YNNY	YNNN	YNYY	YYNY	YNYN	YYYY	YYNN	YYYN	NNNY	NNNN	NNYY	NYNY	NNYN	NYNN	NYYY	NYYN
ECDHE-ECDSA-CHACHA20-POLY1305	NNNY	NNNN	NNYY	NYNY	NNYN	NYYY	NYNN	NYYN	YNNY	YNNN	YNYY	YYNY	YNYN	YYNN	YYYY	YYYN
ECDHE-ECDSA-AES256-GCM-SHA384	YNNY	YNNN	YNYY	YYNY	YNYN	YYYY	YYNN	YYYN	NNNY	NNNN	NNYY	NYNY	NNYN	NYNN	NYYY	NYYN
OpenVPN-TLS(AES-256)-SHA-1	NNNN	NNNY	NNYN	NYNN	NNYY	NYYN	NYNY	NYYY	YNNN	YNNY	YNYN	YYNN	YNYY	YYNY	YYYN	YYYY
OpenVPN-TLS(Blowfish)-SHA-1	NNNN	NNNY	NNYN	NYNN	NNYY	NYYN	NYNY	NYYY	YNNN	YNNY	YNYN	YYNN	YNYY	YYNY	YYYN	YYYY
SSTP-TLS-(AES-256)-SHA256	NNYN	NNYY	NNNN	NYYN	NYNY	NYNN	NYYY	NYNY	YNYN	YNYY	YNNN	YYYN	YNNY	YYYY	YYNN	YYNY
IKEv2-ECP256-AES-256-SHA-256	YNYY	YNYN	YNNY	YYYY	YNNN	YYNY	YYYN	YYNN	NNYY	NNYN	NNNY	NYYY	NNNN	NYYN	NYNY	NYNN
IKEv2-ECP256-Blowfish256-SHA-256	YYYY	YYYN	YYNY	YNYY	YYNN	YNNY	YNYN	YNNN	NYYY	NYYN	NYNY	NNYY	NYNN	NNYN	NNNY	NNNN
IKEv2-ECP-256-Camellia256-SHA-256	YYYY	YYYN	YYNY	YNYY	YYNN	YNNY	YNYN	YNNN	NYYY	NYYN	NYNY	NNYY	NYNN	NNYN	NNNY	NNNN
L2TP/IPsec(PSK)-ECP256-AES256-SHA256	NYYN	NYYY	NYNY	NNYY	NYNY	NNNN	NNYY	NNNY	YYYN	YYYY	YYNN	YNYN	YYNY	YNYY	YNNN	YNNY
L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256	NYYN	NYYY	NYNY	NNYY	NYNY	NNNN	NNYY	NNNY	YYYN	YYYY	YYNN	YNYN	YYNY	YNYY	YNNN	YNNY
L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	NYYN	NYYY	NYNY	NNYY	NYNY	NNNN	NNYY	NNNY	YYYN	YYYY	YYNN	YNYN	YYNY	YNYY	YNNN	YNNY

Рисунок 4.19 — Расстановка промежуточных оценок и определение групп рекомендуемых к использованию методов аутентификации

Зеленым цветом выделены рекомендуемые к использованию методы при данном сценарии, желтым — допустимые к использованию методы, красным — нерекомендуемые к использованию методы.

На следующем этапе создаются ранжированные таблицы решений и распределяются итоговые оценки для каждого метода аутентификации. Ранжирование проводилось по каждому параметру из группы сценариев.

Ранжированная таблица решений для сценариев с приоритетным параметром S показана на рисунке 4.20.

	SPDE	SPDe	SPdE	SpDE	SPde	SpdE	SpDe	Spde
TLS_CHACHA20_POLY1305_SHA256	4	0	3	4	3	1	5	1
TLS_AES_256_GCM_SHA384	5	0	4	5	4	2	6	2
ECDHE-ECDSA-CHACHA20-POLY1305	0	0	7	7	0	5	0	7
ECDHE-ECDSA-AES256-GCM-SHA384	6	0	5	6	5	3	7	3
OpenVPN-TLS(AES-256)-SHA-1	0	0	0	0	11	8	12	5
OpenVPN-TLS(Blowfish)-SHA-1	0	0	0	0	10	9	11	4
SSTP-TLS-(AES-256)-SHA256	0	7	0	8	6	0	2	8
IKEv2-ECP256-AES-256-SHA-256	3	6	6	1	0	4	1	6
IKEv2-ECP256-Blowfish256-SHA-256	1	1	1	2	1	6	3	0
IKEv2-ECP-256-Camellia256-SHA-256	2	2	2	3	2	7	4	0
L2TP/IPsec(PSK)-ECP256-AES256-SHA256	9	5	10	11	9	0	10	0
L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256	7	3	8	9	7	0	8	0
L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	8	4	9	10	8	0	9	0

Рисунок 4.20 — Ранжированная таблица решений для сценариев с приоритетным параметром S

Ранжированная таблица решений для сценариев с приоритетным параметром s показана на рисунке 4.21.

	sPDE	sPDe	sPdE	spDE	sPde	spDe	spdE	spde
TLS_CHACHA20_POLY1305_SHA256	0	0	9	11	0	0	6	10
TLS_AES_256_GCM_SHA384	0	0	8	10	0	0	5	9
ECDHE-ECDSA-CHACHA20-POLY1305	5	0	1	2	7	7	1	3
ECDHE-ECDSA-AES256-GCM-SHA384	0	0	7	9	0	8	4	8
OpenVPN-TLS(AES-256)-SHA-1	0	5	2	4	1	2	2	2
OpenVPN-TLS(Blowfish)-SHA-1	0	6	3	5	2	3	3	1
SSTP-TLS-(AES-256)-SHA256	4	4	0	1	6	1	7	4
IKEv2-ECP256-AES-256-SHA-256	6	0	0	3	0	9	8	0
IKEv2-ECP256-Blowfish256-SHA-256	0	8	1	13	0	0	0	0
IKEv2-ECP-256-Camellia256-SHA-256	0	7	10	12	0	0	0	0
L2TP/IPsec(PSK)-ECP256-AES256-SHA256	1	1	4	6	3	4	0	5
L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256	3	3	6	8	5	6	0	7
L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	2	2	5	7	4	5	0	6

Рисунок 4.21 — Ранжированная таблица решений для сценариев с приоритетным параметром s

Ранжированные таблицы решений для сценариев с приоритетными параметрами P, p, D, d, E, e приведены в приложении Ж.

При помощи полученных таблиц решений, определяется набор методов аутентификации, рекомендуемых для реализации в защищенном канале обмена данными между мобильным абонентом и облачным сервисом при определенных условиях среды обмена данными.

Необходимо поддерживать методы аутентификации, получившие самую высокую оценку — 1, но на результат выбора метода аутентификации может быть оказано воздействие множества факторов (невозможность программной реализации в мобильном абоненте какого-либо протокола, зависимость от других прикладных приложений в системе мобильного абонента, недостаточная производительность аппаратных ресурсов), поэтому в таблицах предложено несколько (три и более) вариантов методов аутентификации для каждого сценария. В порядке убывания выбирается приоритетный метод.

При необходимости, возможна реализация нескольких методов

аутентификации в одном канале связи, например, в случае смены сценария обмена данными (смена типа передаваемых данных и их класса защищенности, смена характера движения абонента) или при организации нескольких каналов одновременно. В таком случае необходимо предусмотреть механизм переключения метода аутентификации в его программной реализации на мобильном устройстве.

Сценарии могут дополняться новыми группами, их список может расширятся в зависимости от потребностей производителя мобильного устройства — участника обмена данными. В таком случае, при создании нового сценария для его оценки можно использовать результаты экспериментов, отраженные в исходных данных для составления таблиц решений (Рисунок 4.17).

4.4 Выводы по главе 4

Разработана методика проведения экспериментов на имитационной модели. Выбрано программное обеспечение для проведения экспериментов.

Проведен ряд экспериментов по измерению скорости обмена данными в канале связи, вычислены значения средней скорости и зафиксированы значения пропускной способности канала обмена данными.

Также проведены эксперименты по измерению скорости обмена данными в канале связи для потока данных малого объема и рассчитаны значения средней скорости в процессе обмена такими данными.

В ходе экспериментов были определены значения параметра отказоустойчивости, такие как время восстановления сеанса связи в защищенном канале обмена данными при использовании одного и двух физических интерфейсов обмена данными

Проведены эксперименты по определению процента загрузки центрального процессора в процессе обмена данными и рассчитаны средние значения загрузки.

Определены значения задержек передачи пакетов и рассчитаны средние задержки передачи пакетов.

Определены проценты потерь пакетов во время сеанса связи.

Заключение

В магистерской работе предложено решение задачи по организации устойчивого к разрывам соединения защищенного канала между мобильным абонентом и веб-сервисом с помощью криптостойких методов аутентификации.

В процессе решения данной задачи были получены следующие результаты:

1. Разработана модель оценки качества защищенного канала передачи данных.
2. Разработана классификация мобильных абонентов — пользователей веб-сервисов.
3. Разработана классификация передаваемых данных.
4. Разработаны сценарии обмена данными — наборы характеристик мобильных абонентов и передаваемых данных, определяющих условия такого обмена данными.
5. Проведен анализ актуальных данных о криптостойкости алгоритмов шифрования, криптографических хеш-функций, сервисов защищенного канала обмена данными и определен набор методов аутентификации, обеспечивающих максимальный уровень защищенности информации при умеренных требованиях к производительности системы абонента.
6. Разработана имитационная модель защищенного канала обмена данными между мобильным абонентом и облачным сервисом на основе реальной сети с системой маршрутизаторов MikroTik, работающих под управлением операционной системы RouterOS. На модели реализован полученный набор методов аутентификации.
8. С помощью метода модельных экспериментов проведено исследование эффективности применения методов аутентификации в различных сценариях подключения мобильных абонентов к облачным сервисам. Результатом исследования является таблицы решений, определяющие приоритет использования метода аутентификации при установленном сценарии обмена данными.

Список литературы

1. Аверченков В. И., Рытов М. Ю., Гайнулин Т. Р., Голембиовская О.М. Формализация выбора решения при проектировании комплексных систем защиты информации от несанкционированного доступа / Известия Волгоградского государственного технического университета.-Волгоград: ВолГТУ. 2011. №11(84).
2. Афанасьев А. А., Веденьев Л. Т., Воронцов А. А. Аутентификация. Теория и практика обеспечения безопасного доступа к информационным ресурсам: учебное пособие / А. А. Афанасьев, Л. Т. Веденьев, А. А. Воронцов — М.: Издательство «Горячая линия-Телеком», 2012. — 235 с.
3. Борзенкова С.Ю., Чечуга О.В. Модель принятия решения при управлении системой защиты информации / Известия Тульского государственного университета. Технические науки. 2013. № 3.
4. Баранова Е. К. Информационная безопасность. История специальных методов криптографической деятельности: Учебное пособие / М.: Риор, 2008. — 86 с.
5. Запечников С.В. Криптографические методы защиты информации: Учебное пособие / С.В. Запечников, О.В. Казарин, А.А. Тарасов. — М.: Юрайт, 2016. — 311 с.
6. Малюк А. А. Информационная безопасность: концептуальные и методологические основы защиты информации / А. А. Малюк — М.: ГЛТ, 2016. — 280 с.
7. Панасенко С. П. Алгоритмы шифрования. Специальный справочник / С. П. Панасенко — М.: Издательство «БХВ-Петербург», 2009. — 465 с.
8. Партыка Т. Л. Информационная безопасность: Учебное пособие / М.: Форум, 2018. — 54 с.
9. Росенко А.П. Внутренние угрозы безопасности конфиденциальной информации: Методология и теоретическое исследование: монография – М.: Красанд, 2010. — 160 с.
10. Рябко Б.Я. Криптографические методы защиты информации / Б.Я.

Рябко, А.Н. Фионов. — М.: ГЛТ, 2013. — 142 с.

11. Сапожников А.В. Современная оценка качества передачи данных по каналам связи / Т-COMM: Телекоммуникации и транспорт. Серия «Общие и комплексные проблемы естественных и точных наук», 2011. — 13 с.

12. Сингх С. Книга шифров: Тайная история шифров и их расшифровки / С. Сингх — М.: Издательство «АСТ», 2009. — 145 с.

13. Фергюсон Н., Шнайер Б. Практическая криптография / Н. Фергюсон, Б. Шнайер М.: «Диалектика», 2004. — 43 с.

14. Черемушкин А. В. Криптографические протоколы. Основные свойства и уязвимости: учебное пособие / А. В. Черемушкин — М.: Издательский центр «Академия», 2009. — 64 с.

15. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы и исходный код на С / Б. Шнайер М.: Вильямс, 2016. — 274 с.

16. Al-Shaer E., Marrero W., El-Atway A., AlBadani K. Network Configuration in a Box: Towards End-to-End Verification of Network Reachability and Security / Proceedings of 17th International Conference on Network Communications and Protocol (ICNP’09), Princeton, 2009. — 12 с.

17. Allard F. An application of the context transfer protocol: IPsec in IPv6 mobility environment / International journal of communication networks and distributed systems, vol. 1, no. 1, 2008. — 13 с.

18. Antoun C. Mobile research methods: Opportunities and challenges of mobile research methodologies / Institute for Social Research, Ann Arbor, USA, 2015. — 24 с.

19. Barker E., Chen L., Roginsky A., Smid M. Recommendation for pair-wise key establishment schemes using discrete logarithm cryptography / NIST Special Publication 800, 56A, 2013.

20. Bonneau J. Mironov I. Cache-collision timing attacks against aes / Cryptographic Hardware and Embedded Systems — CHES 2006, New York, Springer, 2006.

21. Burgess M., Canright G., Engo-Monsen K. A graph-theoretical model of

computer security / International Journal of Information Security. 2004.

22. Dobraunig C., Eichlseder M., Mendel F. Analysis of SHA-512/224 and SHA-512/256 / Graz University of Technology, Austria, 2015.

23. Hankerson, D., Vanstone, S., Menezes, A. Guide to Elliptic Curve Cryptography / Springer Professional Computing. New York, Springer, 2004. — 31 с.

24. Hussain S., Wadhwa N. Performance Analysis of AES and TwoFish Encryption Schemes Communication Systems and Network Technologies // (CSNT) 2011 International Conference, 2011. — 27 с.

25. Manual: IP/IPsec. — URL: https://wiki.mikrotik.com/wiki/Manual:IP/IPsec (дата обр. 28.05.2020).

26. Manual: PPP AAA. — URL: https://wiki.mikrotik.com/wiki/Manual:PPP_AAA (дата обр. 28.05.2020).

27. MikroTik RouterOS and Windows XP IPSec/L2TP — URL: https://wiki.mikrotik.com/wiki/MikroTik_RouterOS_and_Windows_XP_IPSec/L2TP (дата обр. 23.05.2020).

28. Kasper E., Schwabe P. Faster and timing, attack resistant aes-gsm / Cryptographic Hardware and Embedded Systems — CHES, New York, Springer, 2009. — 21 с.

29. Matsumoto S., Reischuk R. IKP: Turning a PKI Around with Decentralized Automated Incentives / In IEEE Symposium on Security and Privacy, 2017. — 24 с.

30. Meyers R., Desoky A. An Implementation of the Blowfish Cryptosystem / Signal Processing and Information Technology, 2008. ISSPIT 2008. IEEE International Symposium on — IEEE, 2008. — 64 с.

31. Oppliger R. Introduction SSL and TLS: Theory and Practice — 2nd. / Artech House, USA 2016. — 327 с.

32. Ristic I. Bulletproof SSL and TLS: Understanding and Deploying SSL/TLS and PKI to Secure Servers and Web Applications / Feisty Duck, 2014. — 311 с.

33. Shafagh H., Hithnawi A. Security Comes First, A Public-key

Cryptography Framework for the Internet of Things / IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS), 2014. — 10 с.

34. Shannon S. Computer — Networking and Networks / New York, USA, Nova Science Publishers, 2006. — 49 с.

35. Thakur J. Kumar N. DES AES and Blowfish: Symmetric key cryptography algorithms simulation-based performance analysis / International journal of emerging technology and advanced engineering, vol. 1, no. 2, 2011. — 42 с.

Приложение А

Код конфигурации в туннеле SSTP

Конфигурация SSTP-сервера (ROUTER1)

```
/ppp profile
add name="sstp-profile" change-tcp-mss=yes
/interface sstp-server server
set authentication=mschap2 port=4443 certificate=none
enabled=yes verify-client-certificate=no force-aes=yes
/ppp secret
add name=sstp-user password=sstp-password
profile="sstp-profile" service=sstp local-
address=172.16.70.1 remote-address=172.16.70.2
/ip firewall filter
add action=accept chain=input dst-port=4443
protocol=tcp comment="sstp"
/ip route
add dst-address=192.168.2.0/24 gateway=172.16.70.2 pref-
src=172.16.70.1 comment="sstp-route"
```

Конфигурация SSTP-клиента (ROUTER2)

```
/interface sstp-client
add connect-to=94.242.32.164 name="sstp"
password=sstp-password user=sstp-user verify-server-
address-from-certificate=no verify-server-certificate=no
allow=mschap2
/ip route
add dst-address=192.168.1.0/24 gateway=172.16.70.1
pref-src=172.16.70.2 comment="sstp-route"
```

Приложение Б

Код конфигурации в туннеле OpenVPN

Конфигурация OpenVPN-сервера (ROUTER1)

```
/ppp profile add name="ovpn-profile" change-tcp-
mss=yes
/interface ovpn-server server
set enabled=yes mode=ip auth=sha1 cipher=aes128
/ppp secret
add name=ovpn-user password=ovpn-password
profile="ovpn-profile" service=ovpn local-
address=172.16.80.1 remote-address=172.16.80.2
/ip firewall filter
add action=accept chain=input dst-port=1194
protocol=tcp comment="ovpn"
/ip route
add dst-address=192.168.2.0/24 gateway=172.16.80.2
pref-src=172.16.80.1 comment="ovpn-route"
```

Конфигурация OpenVPN-клиента (ROUTER2)

```
/interface ovpn-client
add connect-to=94.242.32.164 auth=sha1 cipher=aes128
name="ovpn" password=ovpn-password user=ovpn-user
/ip route
add dst-address=192.168.1.0/24 gateway=172.16.80.1
pref-src=172.16.80.2 comment="ovpn-route"
```

Приложение В

Код конфигурации в туннеле L2TP

Конфигурация L2TP-сервера (ROUTER1)

```
/ppp profile
add name="l2tp-profile" change-tcp-mss=yes use-
encryption=no
/interface l2tp-server
server set authentication=mschap2 enabled=yes
/ppp secret
add name=l2tp-user password=l2tp-password
profile="l2tp-profile" service=l2tp local-
address=172.16.90.1 remote-address=172.16.90.2
/ip firewall filter
add chain=input dst-port=1701 protocol=udp
comment="l2tp"
/ip route
add dst-address=192.168.2.0/24 gateway=172.16.90.2
pref-src=172.16.90.1 comment="l2tp-route"
```

Конфигурация L2TP-клиента (ROUTER2)

```
/interface l2tp-client
add connect-to=94.242.32.164 name="l2tp"
password=l2tp-password user=l2tp-user allow=mschap2
/ip route
add dst-address=192.168.1.0/24 gateway=172.16.90.1
pref-src=172.16.90.2 comment="l2tp"
```

Приложение Г

Код конфигурации в туннеле IKEv2

Конфигурация IKEv2-сервера (ROUTER1)

```
/ip ipsec mode-config
add address-pool="pool vpn.rdatest.ru" address-
prefixlength=32 name="modeconf vpn.rdatest.ru" split-
include=0.0.0.0/0
/ip ipsec profile
add dhgroup=modp2048,modp1536,modp1024
encalgorithm=aes-256,aes-192,aes128 hash-algorithm=sha256
name="profile vpn.rdatest.ru" nat-traversal=yes
proposalcheck=obey
/ip ipsec peer
add exchange-mode=ike2 address=0.0.0.0/0 local-
address=94.242.32.164 name="peer 94.242.32.164"
passive=yes send-initialcontact=yes profile="profile
vpn.rdatest.ru"
/ip ipsec proposal
add auth-algorithms=sha512,sha256,sha1 enc-
algorithms=aes-256-cbc,aes-256-ctr,aes-256-gcm,aes-192-
ctr,aes-192-gcm,aes-128-cbc,aes-128-ctr,aes-128-gcm
lifetime=8h name="proposal vpn.rdatest.ru" pfs-group=none
/ip ipsec policy add template=yes dst-
address=10.0.100.0/24 protocol=all src-address=0.0.0.0/0
group="group vpn.rdatest.ru" proposal="proposal
vpn.rdatest.ru" ipsec-protocols=esp action=encrypt
/ip ipsec identity
add auth-method=digital-signature
certificate=vpn.rdatest.ru remote-
certificate=client@vpn.rdatest.ru generate-policy=port-
```

```
strict match-by=certificate mode-config="modeconf
vpn.rdatest.ru" peer="peer 94.242.32.164" policy-
template-group="group vpn.rdatest.ru" remote-id=user-
fqdn:client@vpn.rdatest.ru
    /ip firewall filter
    add chain=input port=500,4500 protocol=udp
comment="IPsec-PORTS"
    add chain=input protocol=ipsec-esp comment="IPsec-
ESP"
```

Конфигурация IKEv2-клиента (ROUTER2)

```
    /ip ipsec profile
    add dhgroup=modp2048,modp1536,modp1024
encalgorithm=aes-256,aes-192,aes128 hash-algorithm=sha256
name="profile vpn.rdatesr.ru" nat-traversal=yes
proposalcheck=obey
    /ip ipsec peer
    add address=vpn.rdatest.ru exchange-mode=ike2
name="peer vpn.rdatest.ru"
    profile="profile vpn.rdatest.ru"
    /ip ipsec proposal
    add auth-algorithms=sha512,sha256,sha1 enc-
algorithms=aes-256-cbc,aes-256-ctr,aes-256-gcm,aes-192-
ctr,aes-192-gcm,aes-128-cbc,aes-128-ctr,aes-128-gcm
lifetime=8h name="proposal vpn.rdatest.ru" pfs-group=none
    /ip ipsec policy group
    add name="group vpn.rdatest.ru"
    /ip ipsec policy
    add comment="policy template vpn.rdatest.ru" dst-
address=0.0.0.0/0 group="group vpn.rdatest.ru"
```

```
proposal="proposal vpn.rdatest.ru" src-
address=192.168.1.0/24 template=yes
    /ip firewall address-list
    add address=192.168.2.0/24 list=LAN
    /ip ipsec mode-config
    add name="modeconf client@vpn.rdatest.ru"
responder=no src-address-list=LAN
    /ip ipsec identity add auth-method=digital-signature
certificate=client@vpn.rdatest.ru generate-policy=port-
strict mode-config="modeconf client@vpn.rdatest.ru" my-
id=user-fqdn:client@vpn.rdatest.ru peer="peer
vpn.rdatest.ru" policy-template-group="group
vpn.rdatest.ru" remote-id=fqdn:vpn.rdatest.ru
```

Приложение Д

Код программ имитационного моделирования

disabledW1

```
/interface ethernet set ether1 disabled=yes
:delay 10s
/interface ethernet set ether1 disabled=no
:log info message="disabledW1";
```

disabledALLactiveW2

```
/interface ethernet set ether2 disabled=yes
/interface ethernet set ether1 disabled=yes
:delay 10s
/interface ethernet set ether2 disabled=no
:log info message="disabledALLactiveW2";
```

disabledALLactiveW1

```
/interface ethernet set ether2 disabled=yes
/interface ethernet set ether1 disabled=yes
:delay 10s
/interface ethernet set ether1 disabled=no
:log info message=" disabledALLactiveW1";
```

Приложение Е

Код генерации сертификатов

```
/certificate
add name=CAcert country=RU state="SPB" locality=SPB
organization=RDA common-name=ca.rdatest.ru subject-alt-
name=DNS:ca.rdatest.ru key-size=4096 days-valid=3650
trusted=yes key-usage=digital-signature,key-
encipherment,data-encipherment,key-cert-sign,crl-sign
/certificate
sign CAcert
/certificate
add name=SERVERcert country=RU state="SPB"
locality=SPB organization=RDA common-name=vpn.rdatest.ru
subject-alt-name=DNS:vpn.rdatest.ru key-size=2048 days-
valid=3650 trusted=yes key-usage=tls-server
/certificate sign SERVERcert
ca=CAcert
/certificate
add name=CLIENTcert country=RU state="SPB"
locality=SPB organization=RDA common-
name=client@vpn.rdatest.ru subject-alt-name=email:
client@vpn.rdatest.ru key-size=2048 days-valid=3650
trusted=yes key-usage=tls-client
/certificate sign CLIENTcert
ca= CAcert
/certificate
export-certificate CLIENTcert type=pkcs12
export-passphrase=password
```

Приложение Ж

Ранжированные таблицы решений для сценариев с приоритетными параметрами P, p, D, d, E, e

	SPDE	SPDe	SPdE	SPde	sPDE	sPDe	sPdE	sPde
TLS_CHACHA20_POLY1305_SHA256	9	0	5	11	0	0	11	0
TLS_AES_256_GCM_SHA384	8	0	4	10	0	0	10	0
ECDHE-ECDSA-CHACHA20-POLY1305	0	0	10	0	6	0	1	7
ECDHE-ECDSA-AES256-GCM-SHA384	7	0	3	9	0	0	9	0
OpenVPN-TLS(AES-256)-SHA-1	0	0	0	6	0	7	7	4
OpenVPN-TLS(Blowfish)-SHA-1	0	0	0	7	0	8	8	5
SSTP-TLS-(AES-256)-SHA256	0	7	0	8	5	4	0	6
IKEv2-ECP256-AES-256-SHA-256	3	6	9	0	4	0	0	0
IKEv2-ECP256-Blowfish256-SHA-256	2	5	2	5	0	6	6	0
IKEv2-ECP-256-Camellia256-SHA-256	1	2	1	2	0	5	3	0
L2TP/IPsec(PSK)-ECP256-AES256-SHA256	4	1	6	1	1	1	2	1
L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256	6	4	8	4	3	3	5	3
L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	5	3	7	3	2	2	4	2

Рисунок Ж.1 — Ранжированная таблица решений для сценариев с приоритетным параметром P

	SpDE	SpdE	SpDe	Spde	spDE	spDe	spdE	spde
TLS_CHACHA20_POLY1305_SHA256	2	1	3	1	4	0	2	0
TLS_AES_256_GCM_SHA384	3	2	4	2	5	0	3	0
ECDHE-ECDSA-CHACHA20-POLY1305	7	4	0	6	2	7	1	3
ECDHE-ECDSA-AES256-GCM-SHA384	4	3	5	3	6	8	4	0
OpenVPN-TLS(AES-256)-SHA-1	0	7	7	5	8	3	6	2
OpenVPN-TLS(Blowfish)-SHA-1	0	6	6	4	7	2	5	1
SSTP-TLS-(AES-256)-SHA256	8	0	1	7	1	1	7	4
IKEv2-ECP256-AES-256-SHA-256	1	5	2	8	3	9	8	0
IKEv2-ECP256-Blowfish256-SHA-256	5	8	8	0	9	0	0	0
IKEv2-ECP-256-Camellia256-SHA-256	6	9	11	0	12	0	0	0
L2TP/IPsec(PSK)-ECP256-AES256-SHA256	1	0	12	0	13	6	0	7
L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256	9	0	9	0	10	4	0	5
L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	10	0	10	0	11	5	0	6

Рисунок Ж.2 — Ранжированная таблица решений для сценариев с приоритетным параметром p

	SPDE	SPDe	SpDE	SpDe	sPDE	sPDe	spDE	spDe
TLS_CHACHA20_POLY1305_SHA256	9	0	6	12	0	0	13	0
TLS_AES_256_GCM_SHA384	8	0	5	11	0	0	12	0
ECDHE-ECDSA-CHACHA20-POLY1305	0	0	1	0	6	0	3	9
ECDHE-ECDSA-AES256-GCM-SHA384	7	0	4	10	0	0	11	8
OpenVPN-TLS(AES-256)-SHA-1	0	0	0	9	0	8	10	6
OpenVPN-TLS(Blowfish)-SHA-1	0	0	0	8	0	7	9	5
SSTP-TLS-(AES-256)-SHA256	0	7	10	2	5	4	2	1
IKEv2-ECP256-AES-256-SHA-256	3	6	1	1	4	0	1	7
IKEv2-ECP256-Blowfish256-SHA-256	2	5	3	7	0	6	8	0
IKEv2-ECP-256-Camellia256-SHA-256	1	4	2	6	0	5	7	0
L2TP/IPsec(PSK)-ECP256-AES256-SHA256	5	2	8	4	2	2	5	4
L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256	4	1	7	3	1	1	4	2
L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	6	3	9	5	3	3	6	3

Рисунок Ж.3 — Ранжированная таблица решений для сценариев с приоритетным параметром D

	SPdE	SPde	SpdE	Spde	sPdE	sPde	spdE	spde
TLS_CHACHA20_POLY1305_SHA256	1	1	1	1	1	0	2	5
TLS_AES_256_GCM_SHA384	2	2	2	2	2	0	3	6
ECDHE-ECDSA-CHACHA20-POLY1305	6	0	4	6	1	6	1	3
ECDHE-ECDSA-AES256-GCM-SHA384	3	3	3	3	3	0	4	7
OpenVPN-TLS(AES-256)-SHA-1	0	5	6	4	5	1	5	1
OpenVPN-TLS(Blowfish)-SHA-1	0	6	7	5	6	2	6	2
SSTP-TLS-(AES-256)-SHA256	0	4	0	7	0	7	7	4
IKEv2-ECP256-AES-256-SHA-256	7	0	5	8	0	0	8	0
IKEv2-ECP256-Blowfish256-SHA-256	4	7	8	0	7	0	0	0
IKEv2-ECP-256-Camellia256-SHA-256	5	8	9	0	8	0	0	0
L2TP/IPsec(PSK)-ECP256-AES256-SHA256	9	10	0	0	10	4	0	9
L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256	10	11	0	0	11	5	0	10
L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	8	9	0	0	9	3	0	8

Рисунок Ж.4 — Ранжированная таблица решений для сценариев с приоритетным параметром d

	SPDE	SPdE	SpDE	SpdE	sPDE	sPdE	spDE	spdE
TLS_CHACHA20_POLY1305_SHA256	4	1	2	1	0	2	4	2
TLS_AES_256_GCM_SHA384	5	2	3	2	0	3	5	3
ECDHE-ECDSA-CHACHA20-POLY1305	0	6	7	4	4	1	1	1
ECDHE-ECDSA-AES256-GCM-SHA384	6	3	4	3	0	4	6	4
OpenVPN-TLS(AES-256)-SHA-1	0	0	0	8	0	10	12	5
OpenVPN-TLS(Blowfish)-SHA-1	0	0	0	9	0	11	13	6
SSTP-TLS-(AES-256)-SHA256	0	0	11	0	6	0	3	8
IKEv2-ECP256-AES-256-SHA-256	3	7	1	5	5	0	2	7
IKEv2-ECP256-Blowfish256-SHA-256	2	5	6	7	0	6	8	0
IKEv2-ECP-256-Camellia256-SHA-256	1	4	5	6	0	5	7	0
L2TP/IPsec(PSK)-ECP256-AES256-SHA256	7	8	8	0	1	7	9	0
L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256	9	10	10	0	3	9	11	0
L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	8	9	9	0	2	8	10	0

Рисунок Ж.5 — Ранжированная таблица решений для сценариев с приоритетным параметром E

	SPDe	SPde	SpDe	Spde	sPDe	sPde	spDe	spde
TLS_CHACHA20_POLY1305_SHA256	0	11	12	5	0	0	0	10
TLS_AES_256_GCM_SHA384	0	10	11	4	0	0	0	9
ECDHE-ECDSA-CHACHA20-POLY1305	0	0	0	8	0	7	9	4
ECDHE-ECDSA-AES256-GCM-SHA384	0	9	10	3	0	0	8	8
OpenVPN-TLS(AES-256)-SHA-1	0	3	4	2	6	2	3	3
OpenVPN-TLS(Blowfish)-SHA-1	0	2	3	1	5	1	2	2
SSTP-TLS-(AES-256)-SHA256	6	1	1	7	1	6	1	1
IKEv2-ECP256-AES-256-SHA-256	7	0	2	6	0	0	7	0
IKEv2-ECP256-Blowfish256-SHA-256	4	7	8	0	7	0	0	0
IKEv2-ECP-256-Camellia256-SHA-256	5	8	9	0	8	0	0	0
L2TP/IPsec(PSK)-ECP256-AES256-SHA256	3	6	7	0	4	5	6	7
L2TP/IPsec(PSK)-ECP256-Blowfish256-SHA256	1	4	5	0	2	3	4	5
L2TP/IPsec(PSK)-ECP-256-Camellia256-SHA256	2	5	6	0	3	4	5	6

Рисунок Ж.6 — Ранжированная таблица решений для сценариев с приоритетным параметром e