

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет науки и технологий
имени академика М. Ф. Решетнева»
(СибГУ им. М.Ф. Решетнева)

ИНСТИТУТ Информатики и телекоммуникаций

НАПРАВЛЕНИЕ 10.04.01 Информационная безопасность

МАГИСТЕРСКАЯ ПРОГРАММА Управление информационной
безопасностью автоматизированных систем

КАФЕДРА Безопасности информационных технологий

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Вид ВКР: Магистерская диссертация

Тема ВКР: Протокол передачи информации, реализация и тестирование
шифратора с переменной фрагментацией блока
и генератора ключей для него

Дипломник / В.В. Митращук/

Руководитель / В.В. Золотарёв/

Рецензент / _____/

Ответственный за нормоконтроль / Н.И. Чугунова/

Допускается к защите

Зав. кафедрой

С.Г. Колесников

« _____ » _____ 2018 г.

Красноярск 2018 г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет науки и технологий
имени академика М. Ф. Решетнева»
(СибГУ им. М.Ф. Решетнева)

УТВЕРЖДАЮ

Зав. кафедрой С.Г. Колесников

« _____ » _____ 20__ г.

ЗАДАНИЕ

на выполнение выпускной квалификационной работы (ВКР)

Студенту Митращук Владимиру Владимировичу группы МКБ16-01
направления (специальности) 10.04.01 Информационная безопасность
направленность (профиль, специализация, магистерская программа) Управление ин-
формационной безопасностью автоматизированных систем

1. Вид ВКР: Магистерская диссертация
бакалаврская работа/дипломная работа/дипломный проект/магистерская диссертация

2. Тема ВКР: Протокол передачи информации, реализация и тестирование шифра-
тора с переменной фрагментацией блока и генератора ключей для него
утверждена приказом по университету № _____ от _____ 20__ г.

3. Срок сдачи студентом первого варианта ВКР _____

4. Срок сдачи студентом окончательного варианта ВКР _____

5. Исходные данные RFC 2200, IEEE 802.1X, RFC 2637, РД 45.187-2001, RFC 2401-
2405, RFC 2407-2412, ГОСТ 28147-89, ГОСТ Р 34.12-2015, ГОСТ Р ИСО/МЭК 8073-
96, RFC 4303, RFC 2818, RFC 4251

6. Содержание (перечень вопросов подлежащих разработке в ВКР) область примене-
ния протокола на примерах и этапы внедрения; требования для протокола и шифра-
тора; анализ существующих широкораспространенных решений; описание итогового
протокола и шифратора, схемы инициализации сессии (обмена пакетами); реализация
шифратор и генератор ключей для протокола на Windows, Linux и BeagleBoneBlack
(RocketBeagle) произведенного в Китае; тестирование скорости работы шифратора;
нагрузочное тестирование и расчет энергопотребления шифратора на
BeagleBoneBlack (RocketBeagle); реализация в программе шифратора не более трех
тестов для возможности автоматической проверки шифртекста и ОК; определение па-
раметров протокола и методики их конфигурирования.

Дата выдачи задания _____ 20__ г.

Подпись руководителя ВКР _____ / В.В. Золотарёв/

Задание принял к исполнению дата _____ 20__ г.

Подпись студента _____

АННОТАЦИЯ

Митращук Владимир Владимирович

Протокол передачи информации, реализация и тестирование шифратора с переменной фрагментацией блока и генератора ключей для него

Научный руководитель – к.т.н., доцент Золотарёв Вячеслав Владимирович

С каждым днем все большее количество автоматизированных систем различных видов находят свое широкое применение в повседневной жизни. Для каждой такой системы необходимо наличие возможности безопасного и надежного обмена информацией. В данной работе проанализированы широкораспространенные протоколы и шифраторы для них. Обоснована необходимость разработки нового протокола и шифра. В результате, реализован и протестирован кроссплатформенный шифратор разработана структура и схема обмена пакетами протокола, этапы внедрения и схемы его применения на конкретном оборудовании.

ABSTRACT

Mitrashchuk V. V.

Protocol of data exchange, realization and testing of the encrypt with alternating block fragmentation and a key generator for it

Scientific adviser – Candidate of technical sciences, associate professor V. V. Zolotarev

Every day more and more automated systems of various kinds find their wide application in everyday life. For each such system, there is a need for a safe and reliable exchange of information. In this paper, analyzes widespread protocols and cipher for them. The necessity of developing a new protocol and cipher is grounded. As a result, the cross-platform encrypt is implemented and tested, the structure and the scheme for exchanging protocol packets, the stages of implementation and the schemes of its application on specific equipment are developed.

СОДЕРЖАНИЕ

| | |
|--|-----------|
| ВВЕДЕНИЕ | 5 |
| 1 ПОСТАНОВКА ЗАДАЧИ И ФОРМУЛИРОВАНИЕ ТРЕБОВАНИЙ | 8 |
| 1.1 Область применения | 8 |
| 1.2 Этапы внедрения | 9 |
| 1.3 Требования к протоколу | 13 |
| 1.4 Выводы | 15 |
| 2 ПРОТОКОЛ БЕЗОПАСНОГО ОБМЕНА ИНФОРМАЦИЕЙ..... | 16 |
| 2.1 Анализ безопасности протоколов защищенного обмена информацией | 16 |
| 2.2 Выбор подходящего протокола | 28 |
| 2.3 Описание «Шифратора 125»..... | 33 |
| 2.4 Описание «Протокола 125»..... | 40 |
| 2.5 Выводы | 49 |
| 3 КОНФИГУРИРОВАНИЕ ПРОТОКОЛА, РЕЛИЗАЦИЯ И ТЕСТИРОВАНИЕ ШИФРАТОРА И ГЕНЕРАТОРА КЛЮЧЕЙ..... | 50 |
| 3.1 Реализация «Шифратора 125» и генератора ключей | 50 |
| 3.2 Тесты ОК и шифртекста «Шифратора 125»..... | 57 |
| 3.3 Конфигурирование протокола | 65 |
| 3.4 Выводы | 69 |
| ЗАКЛЮЧЕНИЕ | 70 |
| БИБЛИОГРАФИЧЕСКИЙ СПИСОК | 72 |
| ПРИЛОЖЕНИЯ..... | 77 |

ВВЕДЕНИЕ

Современные технологии все больше входят в нашу жизнь. Эти технологии представляют собой автоматизированные системы (АС), основанные на электронике.

На заре эры электронных технологий главной задачей было создать элементную базу, которая позволяла бы решать задачи автоматизации. Область использования таких технологий была небольшой из-за доступности только узкому кругу специалистов. Эта область включала в себя промышленность и военные структуры.

Но следующим этапом распространения электронных устройств автоматизации стала возможность их использования практически каждым человеком. Первой массовой областью подобных технологий стали телевидение и связь, которые появились практически у каждого человека. Затем распространение получили компьютеры, смартфоны и многие другие электронные устройства, которые позволяли автоматизировать большое количество рутинных операций и просто экономить время.

Сегодня мировое сообщество уже давно осознало необходимость третьего этапа распространения технологий. Это можно заметить по появлению многочисленных проектов под знаками «Open Source» и «Open Hardware», заводов предоставляющих возможность производства собственных микросхем любому человеку по заказу и многое другое.

Кратко этот этап развития можно охарактеризовать созданием условий для того, чтобы каждый человек мог производить необходимые ему технологии, решать любые задачи автоматизации при помощи них. Речь идет о создании условий для ремесла разработки автоматизированных систем или, кратко, «Ремесла Автоматики» (РА). РА преимущественно использует технологии, соответствующие критериям «Open Source» и «Open Hardware».

Инфраструктура необходимая для организации условий РА состоит из четырех элементов:

- производство модулей (в первую очередь, необходимо спроектировать аппаратные компоненты);
- операционная система и среда разработки (для создания программ управления аппаратными компонентами);

- протокол коммуникаций (для передачи информации между аппаратными компонентами);
- система контроля версий (для возможности развития проектов на базе аппаратных компонентов).

В данной работе пойдет речь о разработке протокола безопасного обмена информацией («Протокол 125») с учетом главных принципов РА: «Open Source» и «Open Hardware». Конечная цель создания данного протокола – обеспечение безопасного обмена информацией между устройствами РА, в первую очередь, для устройств, находящихся в отношениях «ведущее-зависимое». «Протокол 125» должен обеспечивать защиту не только конфиденциальности информации, но и ее целостности, доступности. Его действие должно быть направлено прежде всего на защиту передачи телеметрии устройств, информации об изменении конфигурации, команд управления, критичных обновлений системы и уже потом на защиту информации больших объемов, например, видеопотоков.

Цель работы: решение проблемы защиты передачи телеметрии, информации об изменении конфигурации, команд управления, критичных обновлений системы между компонентами автоматизированных систем, находящимися в отношениях «ведущее-зависимое» устройство любой глубины, при помощи протокола безопасного обмена информацией.

Задачи:

- 1) четко определить область применения протокола, описать конкретные примеры его использования и этапы внедрения, на основе этих данных сформулировать требования к протоколу;
- 2) проанализировать существующие широкораспространённые протоколы с возможностью защиты информации и используемые ими шифраторы, с целью определения их соответствия требованиям, предложить итоговый вид подходящего протокола и шифратора, дать их подробное описание и схему инициализации сессии (обмена пакетами);
- 3) реализовать шифратор и генератор ключей для протокола с возможностью запуска на Windows, Linux и BeagleBoneBlack (PocketBeagle) произведенного в

Китае, провести тестирование скорости его работы на ноутбуке и плате BeagleBoneBlack (PocketBeagle) произведенной в Китае, привести примеры использования на конкретном оборудовании, осуществить нагрузочное тестирование и расчет энергопотребления при работе шифратора на плате BeagleBoneBlack (PocketBeagle) произведенной в Китае;

4) включить непосредственно в генератор ключей шифратора не более трех тестов для возможности автоматической проверки шифртекста и ОК, которые, по возможности, должны иметь лучшие показатели определения случайности последовательности среди остальных тестов, взаимодополнять и уменьшать недостатки друг друга, на основании результатов тестирования шифратора сформулировать рекомендации по интерпретации полученных результатов;

5) определить параметры протокола, сформулировать методику конфигурирования протокола в зависимости от возможных каналов обмена данными с целью обеспечения максимально допустимого уровня защиты передаваемой информации, учитывая особенности каждого канала.

Выпускная квалификационная работа содержит 92 страницы основного текста и состоит из аннотации, введения, трех глав, заключения, библиографического списка из 47 источников и 3 приложений; основной текст включает 30 рисунков, 5 таблиц, 10 формул.

1 ПОСТАНОВКА ЗАДАЧИ И ФОРМУЛИРОВАНИЕ ТРЕБОВАНИЙ

1.1 Область применения

«Протокол 125» разрабатывается с целью использования в АС, где устройства находятся в строгих взаимоотношениях «ведущее-зависимое». «Зависимое» устройство может выступать в роли «ведущего» для устройств, подключенных к нему, как «ведомые». У «зависимого» может быть только один «ведущий», а у ведущего может быть неограниченное количество «зависимых». Данный способ организации устройств способен иметь любую глубину последовательных цепочек «ведущее-зависимое» (Рисунок 1.1).

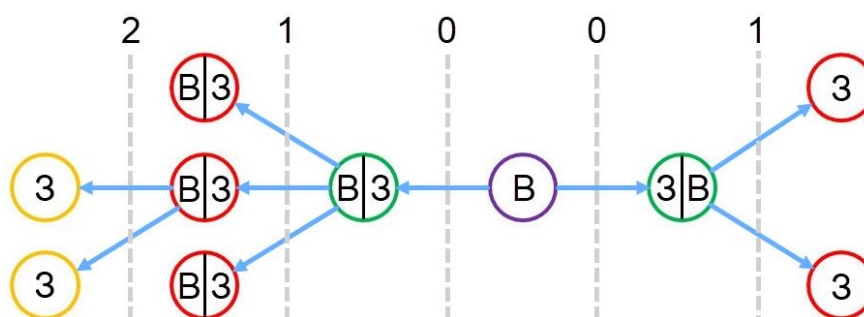


Рисунок 1.1 – Схема подключения устройств «ведущее-зависимое»

«Протокол 125» создается, в первую очередь, для передачи телеметрии, информации об изменении конфигурации, команд управления, критичных обновлений системы устройств. Учитывая, что область применения определяется, в том числе, по скорости передачи информации, предусмотрены следующие этапы интеграции протокола в АС по типу передаваемой информации:

- телеметрия (скорость UART до 115.2 килобит/с);
- большие объемы потоковой телеметрии (скорость UART до 921.6 килобит/с, либо обслуживание нескольких UART с меньшей скоростью);
- звук (от 64 килобит/с по стандарту кодека звука G.711 до 320 килобит/с – наивысшее качество кодирования формата записи звука MP3);
- видео (свыше 384 килобит/с – качество бизнес-ориентированных видеоконференций со сжатием видео).

Расширение скоростных возможностей «Протокола 125» предполагается при помощи:

- использования многоядерных процессоров или графических процессоров;
- повышения вычислительной производительности и компактности устройств;
- удешевления существующих решений;
- повышение энергоэффективности устройств.

1.2 Этапы внедрения

В прошлом разделе была рассмотрена одна из ключевых характеристик для протоколов безопасного обмена информацией – скорость передачи данных. Здесь приводятся этапы внедрения «Протокола 125» не по отдельной характеристике, а в целом, для различных АС.

Этапы внедрения включают в себя следующие виды АС:

- электронный замок (простая система взаимодействия устройств, использующая шифрование, первый шаг испытаний);
- умный дом (масштабирование от простой до сложной АС, включающей большое количество разнородных компонентов, второй шаг испытаний);
- беспилотник (сложная система с удаленным управлением, конфигурированием, обновлением и разными типами каналов связи, третий этап внедрений);
- определение других подходящих АС и интеграция в них (четвертый этап внедрения).

Электронный замок является первым этапом внедрения, потому что имеет простую, законченную и понятную схему функционирования (Рисунок 1.2). Состоит из двух основных частей: микроконтроллера и передатчика. Кроме этого, он не требователен к скорости передачи протокола, потому что передает минимум информации, которая требуется для блокировки, разблокировки устройства, замены ключей и выставления прав доступа для ключей.

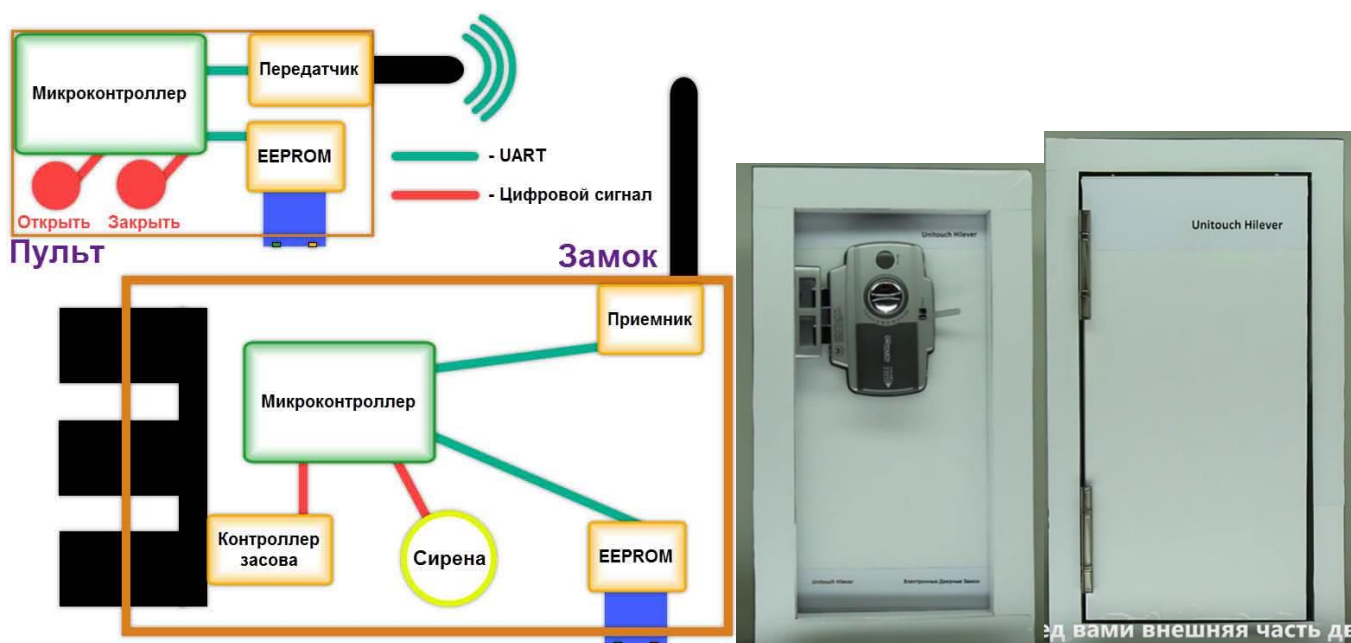


Рисунок 1.2 – Схема скрытого электронного замка

Ведущим устройством выступает замок, ключи – ведомые. Количество одновременно работающих ключей может быть несколько, и они могут обладать различным уровнем доступа, например: доступные засовы для разблокировки, возможность создания или удаления новых ключей, возможность замены своего ключа, доступ к журналу открытий замка и другим лог-файлам системы, возможность настройки конфигурации системы замка и т.п.

«Умный дом» представляет из себя более сложную систему (Рисунок 1.3). Организация подобной АС может серьезно отличаться в зависимости от поставленных целей и задач, иметь очень большую глубину устройств. В «Умном доме» беспроводное соединение используется редко, как альтернативный проводному вариант, когда требуется связь в труднодоступное место, либо в процессе перемещения из любой точки дома для обеспечения мобильности, например, смартфон. Связанно это со следующими преимуществами проводного подключения: трудно создавать помехи, стабильная и на порядки более высокая скорость, повышенная безопасность конфиденциальности (невозможность прослушивания канала без врезки в кабель или коммутатор). Например, устройства охраны периметры рекомендуется подключать только проводным способом для их стабильной работы.

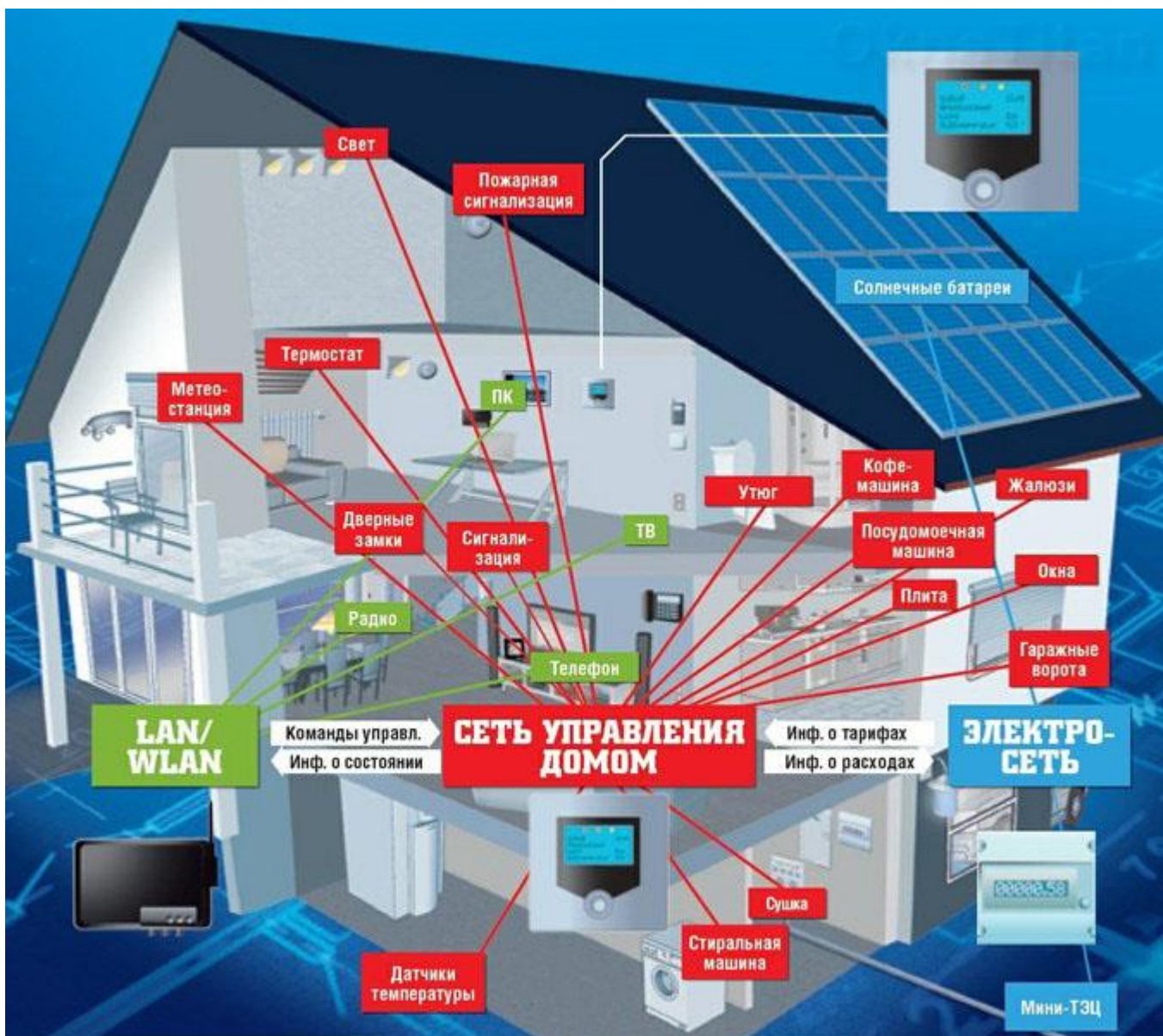


Рисунок 1.3 – Возможная схема подключения устройств «Умного дома»

Беспилотник является сложной и в большей степени законченной системой. Главная трудность по сравнению с предыдущими этапами заключается в беспроводной связи разных видов и управлении на больших расстояниях. Структура устройств беспилотника (Рисунок 1.4) соответствует принципу «ведущий-зависимый» [1].

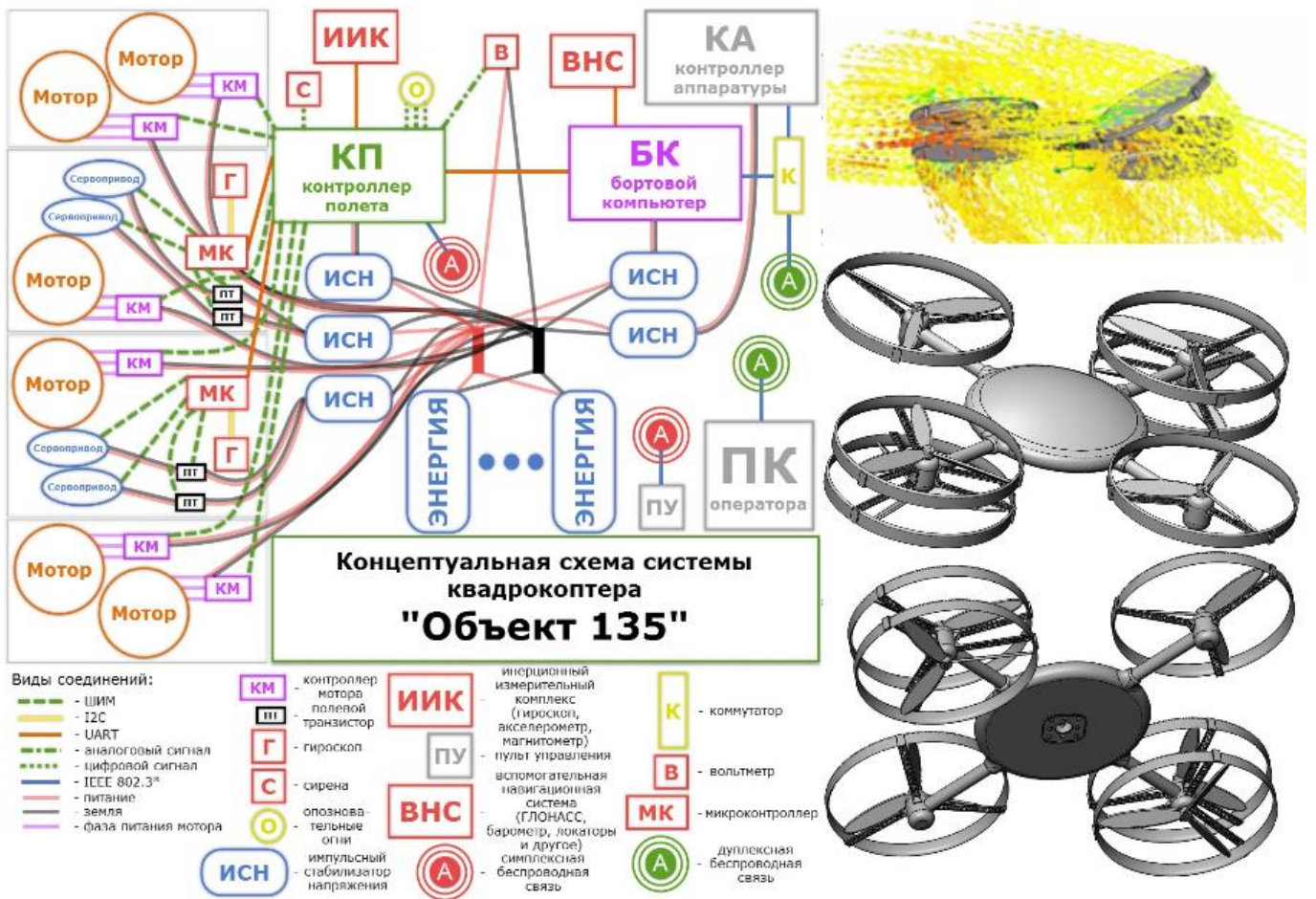


Рисунок 1.4 – Разработанная схема внутренней компоновки оборудования и протоколов взаимодействия, аэродинамика БПЛА и его внешний вид

Связь с беспилотником может происходить по беспроводным каналам разных типов (Рисунок 1.5). В зависимости от поставленных задач используется проводное, беспроводное высокочастотное (быстрое, но на близких расстояниях), беспроводное низкочастотное (медленное, но на дальние расстояния), сотовая связь и спутниковая.

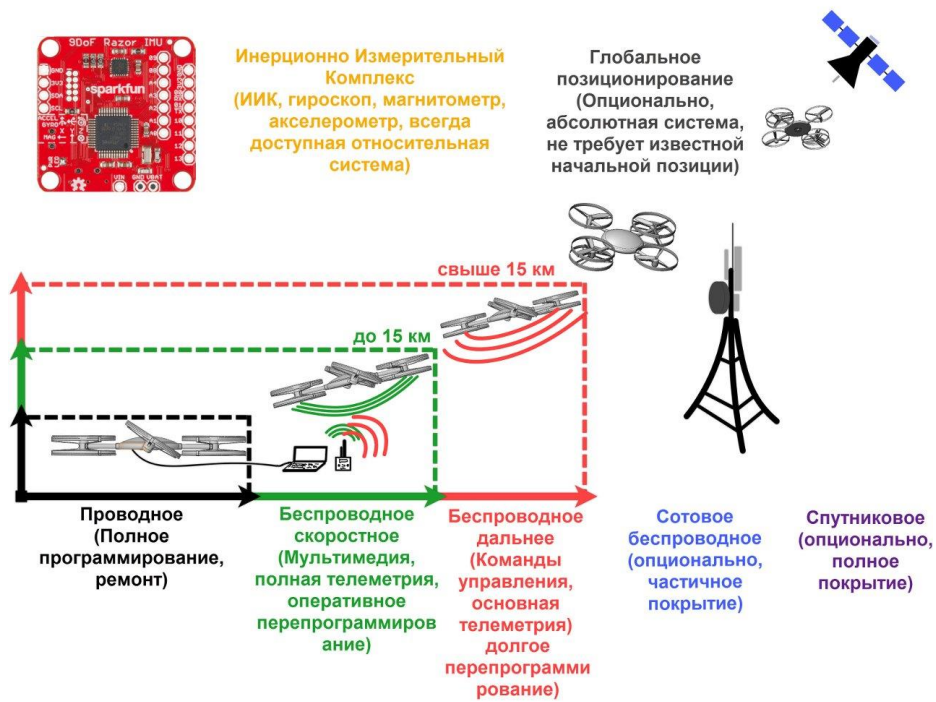


Рисунок 1.5 – Альтернативные каналы связи с беспилотником

Последним этапом идет интеграция в другие проекты, где это может быть необходимо (Рисунок 1.6). Например, как модуль для инфраструктуры РА, VOIP, защиты телеметрии игр и во многих других направлениях.



Рисунок 1.6 – Примеры интеграции протокола в другие АС

1.3 Требования к протоколу

«Протокол 125» должен обладать следующими характеристиками:

1. Поддержка TCP/UDP, транспортного уровня модели OSI. Для обеспечения совместимости с большим количеством оборудования.

2. Скорость шифрования должна быть сопоставима скорости UART. Для возможности использования данного протокола в электронном замке и беспилотнике, где скорость передачи трансивера, например, E31-TTL-500 на расстояние 4000 метров составляет 150 байт/с, а самая большая скорость UART в большинстве устройств «Умного дома» составляет 115 килобайт/с и часто полностью не используется. В случае с «Умным домом» всегда можно прибегнуть к помощи многоядерных процессоров.

3. Использование только симметричного шифрования. Быстрее, чем ассиметричное, например, в случае авторизации каждого пакета новым ключом. Скорость шифрования RSA с длиной ключа 512 составляет 111 байт/с, вместо 120 килобайт/с «Шифратора 125» (тест скорости проводился на одном и том же компьютере). Скорость RSA даже с длиной ключа 512 ниже скорости UART и даже ниже скорости трансивера E31-TTL-500 (большинство систем сегодня конфигурируются длиной ключа 1024, 2048 или 4096 и больше, например «GitLab»). К тому же, скорость генерации ключа RSA в тестируемой программе заняло более 4 минут, это показатель, скорее всего, можно ускорить, используя другие методы генерации ключа RSA, но, все-равно, он будет высок по сравнению с симметричными ключами. Скорость генерации случайного ключа «Шифратора 125» даже для 21 раунда составляет менее 15 секунд. А это сделает практически невозможным авторизацию каждого пакета. Симметричное шифрование достаточно для работы по схеме «ведущее-зависимое». Также, ассиметричное шифрование уязвимо перед алгоритмом Шора [2-9].

4. Задание необходимого объема данных, после отправки которого передается новый ключ, в том числе, возможность создания условий авторизации каждого пакета новым ключом. Позволяет достичь более высокого уровня безопасности и подобрать необходимую загрузку канала, чем авторизация пакета неизменным в рамках сессии или периодически меняющемся ключом без возможности задания необходимого интервала.

5. Имитозащита должна производиться не по хэшу с паролем, а по алгоритму шифрования. Прибавка в скорости вычисления хэша с паролем с целью экономии вычислительных ресурсов до расшифровки сообщения достигается более простым, а следовательно, менее надежным алгоритмом хэша с паролем, чем алгоритмом шифрования. Это создает большую вероятность возникновения угрозы навязывания ложных сообщений после нахождения парольной фразы для хэша с паролем.

6. Алгоритм шифрования должен быть лучше других решений защищен от атаки полным перебором и близок к получению всего множества исходных текстов при таком переборе на выходе.

7. Защита от атаки вызовом отказа в обслуживании.

8. Защита от коллизий хэша служебных полей. Минимизация риска принятия некорректного сообщения.

1.4 Выводы

Удалось определить область применения протокола и его этапы внедрения. Первым этапом является простой в реализации электронный замок. Вторым этапом – масштабируемое решение «Умного дома». Третьим – беспилотник с различными видами каналов удаленного управления. Четвертым – интеграция в другие решения.

Кроме этого, сформулированы 8 требований к протоколу, на основании которых будет выбран итоговый вариант.

2 ПРОТОКОЛ БЕЗОПАСНОГО ОБМЕНА ИНФОРМАЦИЕЙ

2.1 Анализ безопасности протоколов защищенного обмена информацией

Прежде, чем преступить к подведению итоговой таблицы сравнительных характеристик протоколов для выбора наиболее подходящего, необходимо изучить особенности всех рассматриваемых протоколов с точки зрения мер безопасности, которые в них используются.

2.1.1 РРТР

РРТР абсолютно не надежен. В 2012 году было показано, что сложность подбора ключа MSCHAP-v2 эквивалентна подбору ключа к шифрованию DES, и был представлен онлайн-сервис, который способен восстановить ключ за 23 часа [10].

2.1.2 GSM A5/1, A5/3

При помощи поточных шифров удастся достичь скорости шифрования при минимальных вычислительных ресурсах, но все это происходит за счет надежности шифра.

Также, в большинстве случаев, поточные шифры разрабатываются на физическом или канальном уровне по Модели OSI, а это требует дорогостоящей замены всего оборудования. Именно по этой причине только незначительное количество операторов перешло с A5/1 на новый шифр A5/3.

Сегодня существуют программы с открытым кодом, при помощи которых можно расшифровать данные GSM 2G (850/900,1850/1900МГц) практически на любом компьютере за минимальное время [11]. Сделать это может любой, купив приемник для ПК за 400 или 12000 рублей. Время для взлома минимальное – около часа.

Новый шифр A5/3 тоже взломан [12], но для этого требуется больше ресурсов. Организации, государственные службы и другие крупные структуры, в большинстве случаев, смогут позволить себе реализацию данной технологии.

2.1.3 IPsec/L2TP

Протокол IPsec состоит из множества вторичных протоколов (Рисунок 2.1). Для данной работы требуется рассмотреть «Протокол инкапсулирующей защиты содержимого (ESP)» и все алгоритмы, которые он использует для своей работы. Другие модули не связаны напрямую с шифрованием данных.



Рисунок 2.1 – Состав протокола IPsec

Из стандартов rfc1829 и rfc1851 можно сделать вывод, что на данный момент стандартизован для шифрования данных в IPsec только алгоритм DES, а алгоритм 3DES находится в экспериментальной стадии. Это достаточно слабые алгоритмы шифрования: длина блока 64 бит, длина ключа до 168 бит, а сам алгоритм полностью открыт и абсолютно статичен (без использования переменных раундов или таблиц замен). Также, при такой небольшой длине блока можно осуществить частотный анализ блоков и другие варианты атак, ускоряющие нахождения ключа шифра. В IKE [13] используется гораздо больше алгоритмов шифрования, но это отдельный протокол для обмена ключами, задействованный в IPsec, он не входит в протокол шифрования данных ESP.

Рассмотрим сформированный пакет протокола ESP [14]. Ниже показан пакет ESP для туннельного режима (Рисунок 2.2), потому что в поле «next» указано значение «IP». Причем, стоит обратить внимания на тот факт, что в туннельном режиме, например, при использовании L2TP VPN, значение «IP» будет в зашифрованной части пакет предопределенно, что позволяет производить перебор и фильтрацию значений по известным полям структуры пакета «IP».

Даже без использования туннельного режима L2TP, решения которое основывается на IPsec и больше всего распространено на практике, все равно, остается возможность использования данной уязвимости и создания угрозы расшифровки данных, потому что в большинстве случаев обмен в сети Интернет происходит при использовании протокола TCP, реже, UDP.

IPSec in ESP Tunnel Mode

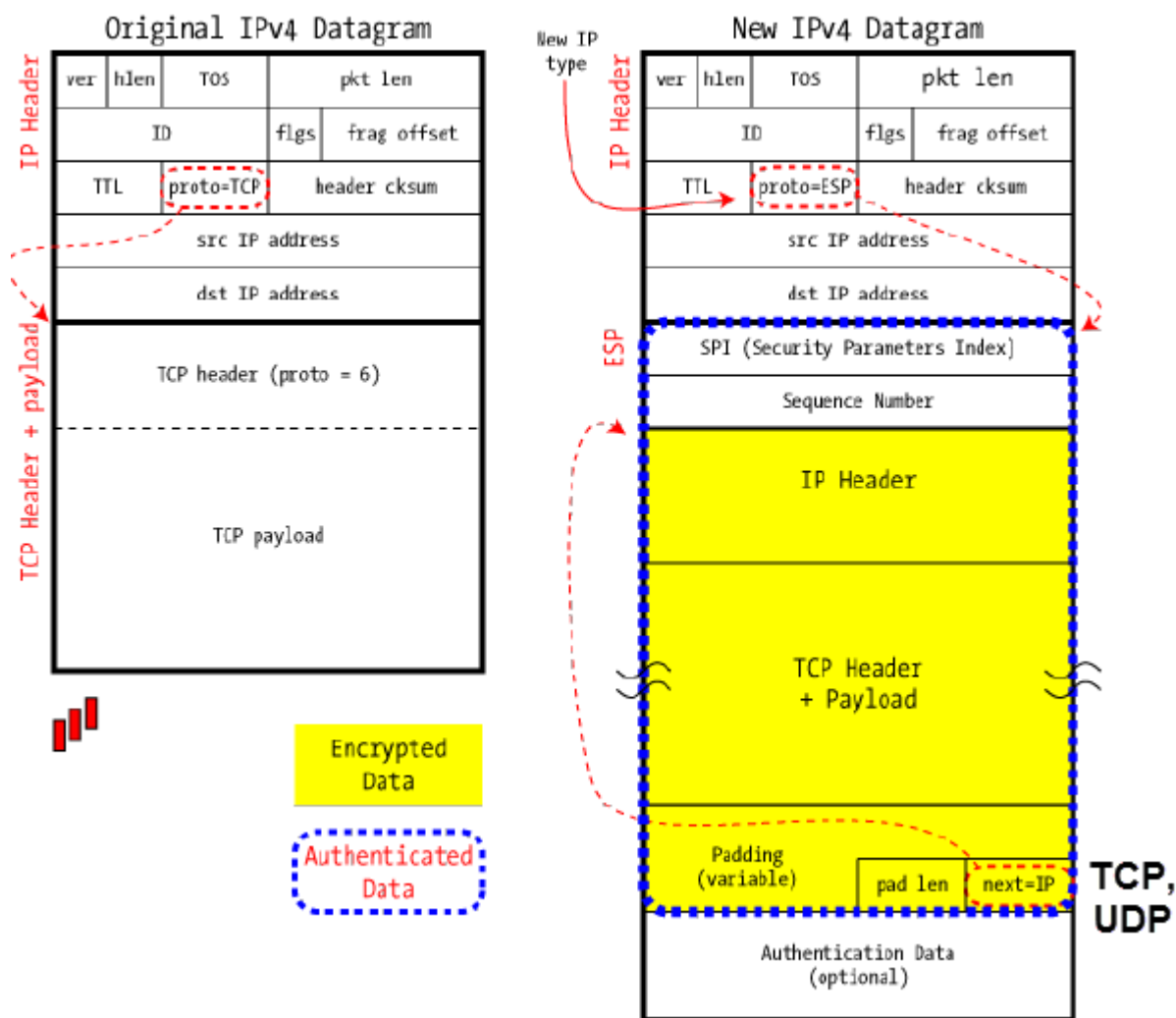


Рисунок 2.2 – Структура ESP

Проанализируем структуру пакета IP и сегментов наиболее распространенных транспортных протоколов TCP, UDP для того, чтобы убедиться в наличии полей, по которым можно осуществить перебор и фильтрацию данных с целью нахождения ключа шифрования.

Для IP можно использовать практически все поля для перебора, например, версий протокола всего две, а обычно используется IPv4. Также, существует стандартная

длина заголовка 20 байт для IP пакетов. Но особенно интересно поле длины пакета, которое включает заголовок и полезные данные, другими словами, это значение можно получить просто, измерив зашифрованный пакет IP за вычетом добавочных полей шифртекста из IPsec. Остальными полями легко можно отфильтровать полученные значения по их допустимой области.

К примеру, возможна такая последовательность расшифровки для протокола IP в L2TP VPN: вычисление длины пакета перебор ключей по полю длины пакета, далее, фильтрация по версии протокола, типу обслуживания, времени жизни пакета, допустимому диапазону IP адресов в случае, если после перебора получилось не одно корректное значение.

В случае отсутствия VPN при использовании IPsec, что является редким явлением, можно подобрать ключ по структуре протоколов TCP, UDP, которые используются в большинстве случаев (Рисунок 2.3).



Рисунок 2.3 – Структуры сегментов TCP и UDP

Алгоритм перебора UDP прост, необходимо сохранять пакеты с корректной длиной дейтаграммы и, затем, по необходимости, фильтровать по допустимым значениям других полей.

Случай с TCP, наверное, является самым ресурсоемким по сравнению с IP и UDP, так как гарантированный способ перебора подразумевает, помимо расшифровки сегмента по подбираемому ключу, еще и взятие контрольной суммы сегмента для сравнения с соответствующим полем. Фильтрацию, также, можно осуществить

по оставшимся полям. Учитывая не большую величину стандартного фрагмента сегмента TCP равную 1500 байт, это более ресурсоемкий, но вполне выполнимый алгоритм, особенно при слабом шифре. В случае размера пакета, отличающегося от 1500 байт, можно попытаться произвести расшифровку при помощи других известных диапазонов полей, например, портов служб. Либо осуществить повтор перебора для любой возможной длины пакета.

Уязвимость установления типа заголовка зашифрованного сообщения создает угрозу нарушения конфиденциальности данных. Злоумышленнику достаточно просто перехватить данные, расшифровку он сможет осуществить позднее.

ESP позволяет произвести, в том числе, защиту от навязывания и контроль целостности при помощи добавления аутентификатора. Для этого дадим ряд определений:

- поле «Индекс параметров безопасности» (Security Parameters Index – SPI) – это значение, которое в совокупности с адресом назначения и самим протоколом однозначно определяет ассоциацию безопасности (Security Association – SA) для данной дейтаграммы в виде 32-битного номера; номера с 1 по 255 зарезервированы IANA; SPI, равный нулю, означает, что SA не установлена; ассоциация безопасности – это набор параметров (версия алгоритмов шифрования и аутентификации, схема обмена ключами и т. п.), определяющих, каким образом будет обеспечиваться защита данных;
- поле «Последовательный номер» (Sequence number) – это монотонно возрастающий от 0 (при установлении SA) номер пакета. Он используется для возможности контроля получателем ситуации повторной пересылки пакетов;
- поле «Данные аутентификации» (Authentication data) содержит значение контроля целостности (Integrity check value – ICV), рассчитанное по всем данным, которые не изменяются в пути следования пакета или предсказуемы на момент достижения им получателя. Значение ICV рассчитывается в зависимости от алгоритма, определенного в SA, например, код аутентификации сообщения (Message Authentication Code – MAC) с ключом симметричного или асимметричного алгоритма, или хэш-функции.

Алгоритм выработки аутентфикатора (хэша) протокола IPsec в общем виде показан ниже (Рисунок 2.4). В ESP поля SPI, номер пакета и аутентфикатора не шифруются с целью обеспечения проверки хэша без расшифровки для защиты от атак на загрузку процессора расшифровкой [15]. Но это позволяет злоумышленнику восстановить исходный секретный ключ для хэша, потому что у него есть все необходимые данные, чтобы начать перебор.

На сколько будет эффективен данный метод, позволяющий имитировать пакеты, зависит от периода обмена новой ключевой информацией. Но обычно, обмен ключевой информацией происходит только в начале сессии.

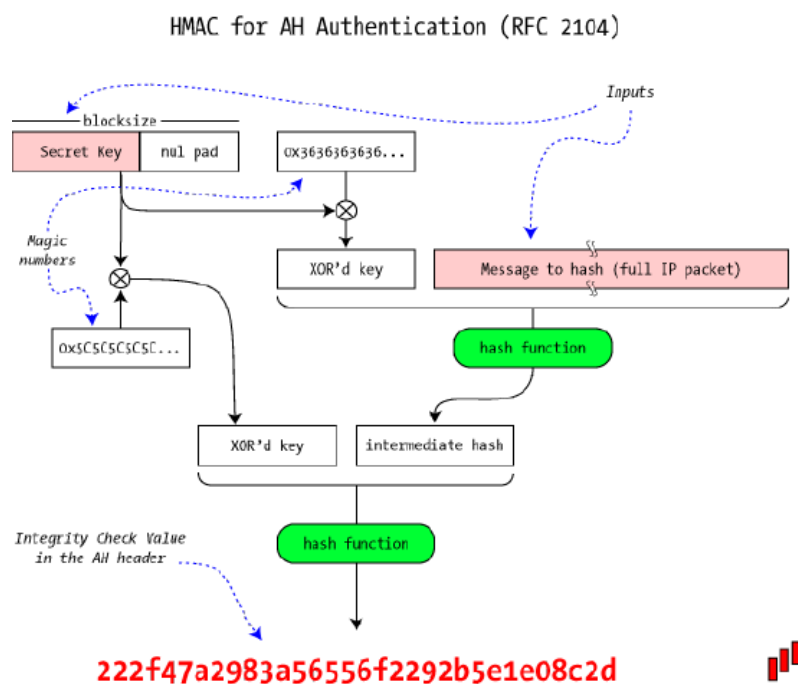
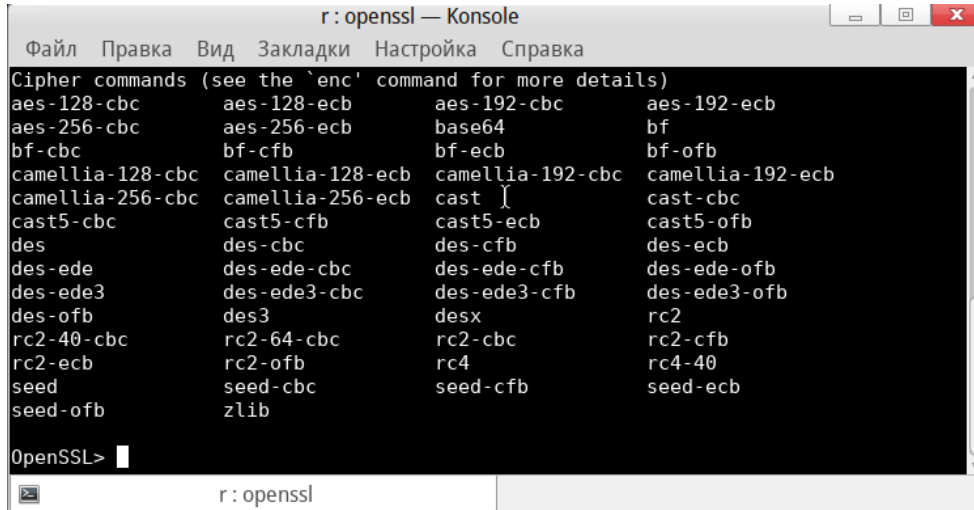


Рисунок 2.4 – Выработка аутентфикатора (хэша)

Также протокол не производит защиту от коллизий в зашифрованном поле типа следующего протокола, что редко, но может вызвать коллизию с корректными данными по другому протоколу. Легче реализовать атаку на доступность, потому что хэш всегда берется по всему сообщению, включая служебные поля, хоть и без расшифрования всех данных.

2.1.4 OpenVPN

Протоколы OpenVPN используют для шифрования и обмена ключами библиотеку OpenSSL, которая содержит следующие шифры (Рисунок 2.5): AES, Blowfish, camellia, CAST, CAST5, DES, 3DES, DESX, RC2, RC4, SEED.



```
r : openssl — Konsole
Файл  Правка  Вид  Закладки  Настройка  Справка
Cipher commands (see the `enc` command for more details)
aes-128-cbc      aes-128-ecb      aes-192-cbc      aes-192-ecb
aes-256-cbc      aes-256-ecb      base64            bf
bf-cbc          bf-cfb           bf-ecb           bf-ofb
camellia-128-cbc  camellia-128-ecb  camellia-192-cbc  camellia-192-ecb
camellia-256-cbc  camellia-256-ecb  cast              cast-cbc
cast5-cbc        cast5-cfb        cast5-ecb        cast5-ofb
des              des-cbc          des-cfb          des-ecb
des-ede         des-ede-cbc     des-ede-cfb     des-ede-ofb
des-ede3        des-ede3-cbc    des-ede3-cfb    des-ede3-ofb
des-ofb         des3            desx             rc2
rc2-40-cbc      rc2-64-cbc      rc2-cbc          rc2-cfb
rc2-ecb         rc2-ofb         rc4              rc4-40
seed            seed-cbc        seed-cfb         seed-ecb
seed-ofb        zlib

OpenSSL>
```

```
OpenSSL> version
OpenSSL 1.0.2m 2 Nov 2017
OpenSSL> ciphers
ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:SRP-DSS-AES-256-CBC-SHA:SRP-RSA-AES-256-CBC-SHA:SRP-AES-256-CBC-SHA:DH-DSS-AES256-GCM-SHA384:DHE-DSS-AES256-GCM-SHA384:DH-RSA-AES256-GCM-SHA384:DHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES256-SHA256:DHE-DSS-AES256-SHA256:DH-RSA-AES256-SHA256:DH-DSS-AES256-SHA256:DHE-RSA-AES256-SHA:DHE-DSS-AES256-SHA:DH-RSA-AES256-SHA:DH-DSS-AES256-SHA:DHE-RSA-CAMELLIA256-SHA:DHE-DSS-CAMELLIA256-SHA:DH-RSA-CAMELLIA256-SHA:DH-DSS-CAMELLIA256-SHA:SRP-DSS-AES-128-CBC-SHA:SRP-RSA-AES-128-CBC-SHA:SRP-AES-128-CBC-SHA:DH-DSS-AES128-GCM-SHA256:DHE-DSS-AES128-GCM-SHA256:DH-RSA-AES128-GCM-SHA256:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES128-SHA256:DHE-DSS-AES128-SHA256:DH-RSA-AES128-SHA256:DH-DSS-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA:DH-RSA-AES128-SHA:DH-DSS-AES128-SHA:DHE-RSA-SEED-SHA:DHE-DSS-SEED-SHA:DH-RSA-SEED-SHA:DH-DSS-SEED-SHA:DHE-RSA-CAMELLIA128-SHA:DHE-DSS-CAMELLIA128-SHA:DH-RSA-CAMELLIA128-SHA:DH-DSS-CAMELLIA128-SHA:ECDH-RSA-AES128-GCM-SHA256:ECDH-ECDSA-AES128-GCM-SHA256:ECDH-RSA-AES128-SHA256:ECDH-ECDSA-AES128-SHA256:ECDH-RSA-AES128-SHA:ECDH-ECDSA-AES128-SHA:AES128-GCM-SHA256:AES128-SHA:SEED-SHA:CAMELLIA128-SHA:PSK-AES128-CBC-SHA:ECDHE-RSA-RC4-SHA:ECDHE-ECDSA-RC4-SHA:ECDH-RSA-RC4-SHA:ECDH-ECDSA-RC4-SHA:RC4-SHA:RC4-MD5:PSK-RC4-SHA:ECDHE-RSA-DES-CBC3-SHA:ECDHE-ECDSA-DES-CBC3-SHA:SRP-DSS-3DES-EDE-CBC-SHA:SRP-RSA-3DES-EDE-CBC-SHA:SRP-3DES-EDE-CBC-SHA:EDH-RSA-DES-CBC3-SHA:EDH-DSS-DES-CBC3-SHA:DH-RSA-DES-CBC3-SHA:DH-DSS-DES-CBC3-SHA:ECDH-RSA-DES-CBC3-SHA:ECDH-ECDSA-DES-CBC3-SHA:DES-CBC3-SHA:PSK-3DES-EDE-CBC-SHA
```

Рисунок 2.5 – Шифры OpenSSL

Все рассмотренные шифры уязвимы к атаке перебора, большинство из них имеют широкоизвестные ускорения такого перебора при минимальных вычислительных мощностях до суток при помощи методов криптоанализа.

Остальные варианты, также, имеют возможность взлома не только перебором, но и криптоанализом, потому что у них полностью статичный алгоритм. Те, что имеют изменяемые блоки замен или раунды, обычно это, либо небольшие блоки замен, либо блоки шифра, что позволяет проводить криптоанализ, например, при накоплении большого количества пакетов. Так был взломан шифр RC4 в протоколе WEP.

Также, изменяемые блоки замен не выносятся в ключ, а поэтому во многих случаях являются общими не для двух, а для многих пользователей шифра, у которых отличаются только ключи к шифру, что значительно упрощает взлом таким внутренним пользователям.

При неимении исходных текстов, их можно получить для криптоанализа или перебора, изучив структуру протокола OpenVPN и поняв, в каком месте начинается инкапсуляция других структурированных протоколов. Также, необходимо выяснить тип передаваемой информации, это значительно ускорит взлом. OpenVPN в режиме туннеля может работать, либо на сетевом уровне «tun», либо на канальном уровне «tap».

В случае сетевого уровня, в большинстве случаев, используется инкапсуляция пакетов «IP», которая была рассмотрена ранее. Рассмотрим подробнее канальный уровень и структуру кадра Ethernet (Рисунок 2.6).



Рисунок 2.6 – Формат кадра Ethernet

Перебор можно осуществлять по контрольной сумме. Обычно длина данных в кадре равна 1500 байт, потому что это стандартный размер пакета IP сетевого уровня в широкораспространенном стеке TCP/IP. Он по умолчанию установлен на большинстве устройств. По остальным полям можно отфильтровать полученные данные, если это будет необходимо.

В случае несоответствия длине 1500 байт, можно пытаться произвести расшифровку по известному адресу или типу. Обычно тип – это IP или ARP пакеты. Либо произвести перебор для любой длины (типу) кадра.

Всегда можно высчитать с какого момента начинается заголовок IP, TCP или UDP пакета и расширить возможность фильтрации за счет полей из данных структур. Анализ полей для получения исходного ключа из структуры TCP, UDP был рассмотрен ранее.

Структура сообщения OpenVPN: Len (2 байта) | HMAC (20 байт) | IV (8 байт) | packet ID (4 байта) | timestamp (4 байта) | packet payload [16].

Len – длина всего пакета, кроме первых двух байт; используется только при работе через TCP. При работе через UDP длина пакета OpenVPN рассчитывается из длины UDP пакета, а поля Len в OpenVPN пакете нет.

HMAC – HMAC-SHA1 от всего, что идёт после HMAC (IV и зашифрованные данные)

IV – вектор инициализации для режима CBC.

Всё, что идёт после IV, передаётся зашифрованным.

Packet ID – номер пакета. Он включён сюда для защиты от повторов, и не используется для организации гарантированной доставки данных.

Timestamp – так же используется для защиты от повторов.

Packet payload – данные, которые несёт пакет. Это либо служебная информация OpenVPN, либо сам инкапсулированный трафик.

Из структуры OpenVPN можно сделать вывод, что HMAC не шифруется и создается по зашифрованному сообщению, аналогично IPsec, следовательно, возможно восстановить исходный ключ HMAC, тем более, этот алгоритм не так стоек по сравнению с шифрами. В результате этого появляется возможность имитировать сообщения.

Поле времени может быть использовано для получения ключа шифра, если известно приблизительное время отправления сообщения, что облегчает криптоанализ. Номер пакета тоже может быть использован, так как это порядковое число, оно не является случайным и очень хорошо выдает закономерности шифра, особенно если отследить начало сессии. Эти два поля шифруются.

2.1.5 HTTPS

Протокол HTTPS использует TLS, что является одним и тем же названием OpenSSL, другими словами, задействованы такие же алгоритмы шифрования и обмена ключами, что и в OpenVPN.

Поэтому, перейдем к анализу структуры HTTPS [17] (Рисунок 2.7). Она состоит из трех ключевых элементов: заголовков, тело передаваемой страницы, трейлер.

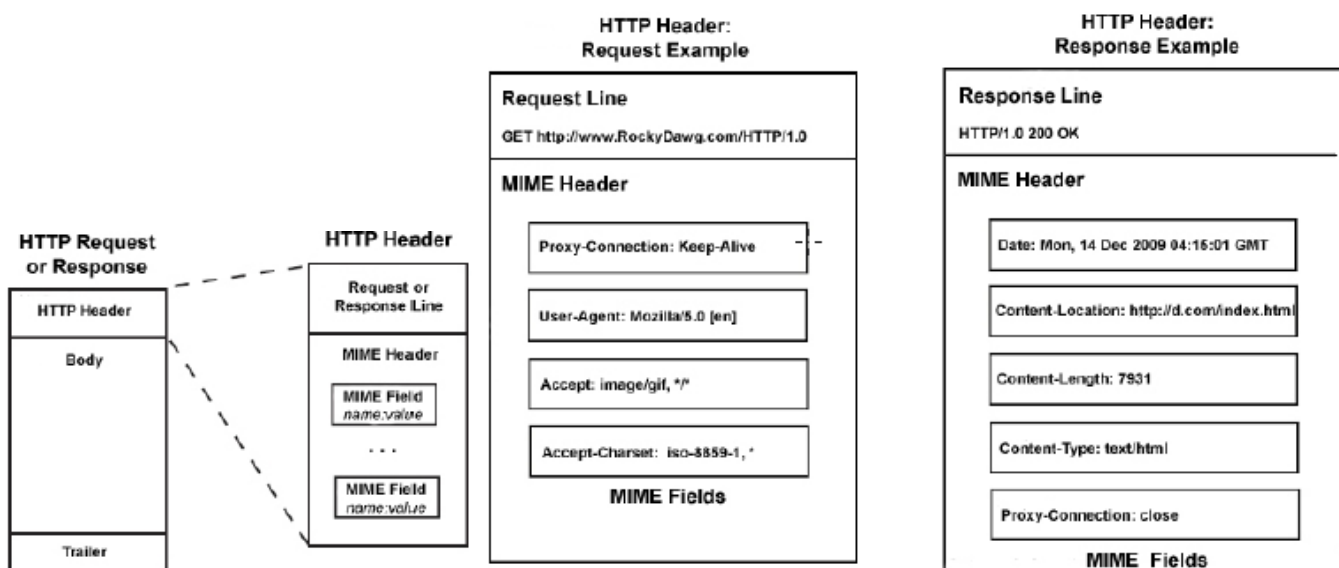


Рисунок 2.7 – Структура HTTPS

Трейлер передает поля заголовка в конце сообщения. Это позволяет передавать динамически созданный контент вместе с информацией, необходимой для получателя, чтобы убедиться, что он получил полное сообщение.

Самым интересным является поле заголовка (и его дубликат в трейлере), потому что оно содержит поля параметров соединения такие, как: длина контента, тип контента (text/html) и другие, по которым можно осуществить перебор и фильтрацию или облегчить криптоаналитику анализ закономерностей шифра.

Также, HTTPS шифрует, но не защищает от повторной передачи сообщений. Это необходимо реализовывать самостоятельно в открытом тексте тела сообщений HTTPS.

2.1.6 WPA2

WPA2 имеет большое количество уязвимостей. Уязвимости, вошедшие в состав утилиты KRACK:

- CVE-2017-13077: Reinstallation of the pairwise encryption key (PTK-TK) in the 4-way handshake;

- CVE-2017-13078: Reinstallation of the group key (GTK) in the 4-way handshake;
- CVE-2017-13079: Reinstallation of the integrity group key (IGTK) in the 4-way handshake;
- CVE-2017-13080: Reinstallation of the group key (GTK) in the group key handshake;
- CVE-2017-13081: Reinstallation of the integrity group key (IGTK) in the group key handshake;
- CVE-2017-13082: Accepting a retransmitted Fast BSS Transition (FT) Reassociation Request and reinstalling the pairwise encryption key (PTK-TK) while processing it;
- CVE-2017-13084: Reinstallation of the STK key in the PeerKey handshake;
- CVE-2017-13086: reinstallation of the Tunneled Direct-Link Setup (TDLS) PeerKey (TPK) key in the TDLS handshake;
- CVE-2017-13087: reinstallation of the group key (GTK) when processing a Wireless Network Management (WNM) Sleep Mode Response frame;
- CVE-2017-13088: reinstallation of the integrity group key (IGTK) when processing a Wireless Network Management (WNM) Sleep Mode Response frame.

Все эти уязвимости [18, 19] связаны в основном с процедурой установления сессии и основная причина в этом – открытость служебных полей и плохо реализованный протокол.

Трафик в сети шифруется не только от устройств, не подключенных к этой сети, но и друг от друга. Каждое устройство имеет свои ключи шифрования для обмена данными с точкой доступа. Существует несколько ключей шифрования:

- 1) Pairwise Transient Key (PTK). При помощи данного типа ключа шифруется личный трафик каждого клиента. Таким образом, обеспечивается защита сети «изнутри», чтобы один клиент, авторизованный в сети, не мог перехватить трафик другого.
- 2) Group Temporal Key (GTK). Данный ключ шифрует широковещательные данные.

WPA2 используется в качестве алгоритма шифрования CCMP. CCMP (Counter Mode with Cipher Block Chaining Message Authentication Code Protocol), протокол блочного шифрования с кодом аутентичности сообщения и режимом сцепления блоков и счётчика – протокол шифрования для сети WPA2, использующий алгоритм AES как основу для шифрования данных. В соответствии со стандартом FIPS-197 используется 128-битный ключ шифрования.

Возможность перебора может осуществляться при помощи получения четырех рукопожатий (handshake) [20], в которых передается Pre-Shared Key при подключении устройства к сети.

Также стоит обратить внимание, что для того, чтобы перехватить handshake злоумышленнику совсем не обязательно ждать того момента, как к сети будет подключено новое устройство. Есть возможность посылки в сеть реассоционных пакетов, которые будут прерывать сетевые соединения и инициировать новый обмен ключами в сети между клиентами и точкой доступа. В таком случае, для того, чтобы захватить требуемые пакеты, необходимо, чтобы к сети был подключен хотя бы один клиент.

2.1.7 SSH

Каждый бинарный пакет имеет следующий формат (Рисунок 2.8) [21].

| | | | | |
|---------------|----------------|----------|----------------|---------|
| uint32 | byte | byte[n1] | byte[n2] | byte[m] |
| packet_length | padding_length | payload | random padding | mac |

Рисунок 2.8 – Формат SSH

Packet_length – длина пакета в байтах, не включая MAC или не включая самого поля длины пакета.

Padding_length – длина перегрузки в байтах.

Payload – полезное содержимое пакета. Если было согласовано сжатие, то это поле сжато. Первоначально сжатие должно быть в положение «нет» (т.е. по умолчанию сжатия нет). Формула расчета: «n1 = packet_length-padding_length-1».

Padding – перегрузка произвольной длины, являющаяся случайным набором байтов. Она должна быть, такой чтобы общая длина конкатенации (`packet_length || padding_length || payload || padding`) была кратна 8 или размеру блока шифра, в зависимости от того, что из них больше. Размер перегрузки должен быть по крайней мере 4 байта. Максимальный размер перегрузки 255 байт. Значение `n2 = padding_length`.

MAC – код аутентификации сообщения. Если аутентификация сообщения согласована, это поле содержит байты MAC. Первоначально, флаг MAC-алгоритма должен быть в положении «нет». Значение `m = mac_length`. Отметим, что поле длины пакета также шифруется.

Для шифрования используются шифры, которые показаны ниже (Рисунок 2.9).

| | | |
|--------------|-------------|--------------------------------|
| 3des-cbc | REQUIRED | three-key 3DES in CBC mode |
| blowfish-cbc | RECOMMENDED | Blowfish in CBC mode |
| twofish-cbc | RECOMMENDED | Twofish in CBC mode |
| arcfour | OPTIONAL | the ARCFOUR stream cipher |
| idea-cbc | OPTIONAL | IDEA in CBC mode |
| cast128-cbc | OPTIONAL | CAST-128 in CBC mode |
| none | OPTIONAL | no encryption; NOT RECOMMENDED |

Рисунок 2.9 – Шифры в SSH

Из данных шифров многие были рассмотрены ранее, но и те, что являются новыми, по своим характеристикам уступают ранее рассмотренным. В целом SSH наследует большинство уязвимостей шифров из предыдущих протоколов. Это и фильтрация по длине и по mac, при включении режима mac. Он оставляет много возможностей для перебора и анализа криптоаналитику.

2.2 Выбор подходящего протокола

2.2.1 Сравнительный анализ соответствия требованиям

После подробного рассмотрения методов обеспечения безопасности в различных широкораспространенных протоколах можно составить сводную таблицу их соответствия требованиям, которые ранее были сформулированы в первой главе (Таблица 2.1).

Таблица 2.1 – Сравнение протоколов на соответствие требованиям

| Протокол | TCP/UDP | Скорость UART | Только симметричный шифр | Авторизация новым ключом по объему данных | Имитозащита по шифру | Защита от отказа в обслуживании | Защита от коллизий служебных полей | Лучшая защита от атаки перебором |
|----------------|----------------------|-----------------|--------------------------|---|----------------------|---------------------------------|------------------------------------|---|
| WEP | | | | | | | | Взлом перехватом 200000 пакетов |
| PPTP | | | | | | | | Подбор ключа за 23 часа |
| GSM A5/1, A5/3 | | | | | | | | Известны способы взлома |
| IPsec | Сетевой уровень | Достаточна | Диффи-Хеллман | Отсутствует | По хэшу | Да | Нет | DES, 3DES |
| OpenVPN | Транспортный уровень | Достаточна | Возможно | Отсутствует | По хэшу | Да | Нет | Могут быть использованы передовые шифры |
| HTTPS | Прикладной | Достаточна | Сертификат | Отсутствует | Отсутствует | Нет | Нет | Могут быть использованы передовые шифры |
| WPA2/PSK | Канальный | Достаточна | Да | Отсутствует | Отсутствует | Нет | Нет | Использованы передовые шифры |
| SSH | Прикладной | Достаточна | Диффи-Хеллман | Отсутствует | По хэшу | Да | Нет | Могут быть использованы передовые шифры |
| Протокол 125 | Транспортный уровень | 120 килобайт /с | Да | Да | Да | Да | Да | Да |

По результатам сравнения протокола можно сделать вывод, что все они обладают достаточной скоростью передачи данных, только некоторые из них не поддерживают протоколы TCP/UDP и не имеют защиты от отказа в обслуживании. Большинство использует для проведения процедуры авторизации и установления сессии ассиметричное шифрование, а имитозащиту осуществляют по нешифрованному, но запароленному хэшу.

У всех отсутствует возможность задания интервала замены ключа симметричного шифрования и авторизации. Также, не предпринимаются действия по дополнительным, не ресурсоемким мерам защиты от коллизии служебных полей. Часть протоколов была полностью взломана и созданы онлайн ресурсы или доступные методы для быстрого повторения взлома.

Ни один из протоколов не соответствует заявленным требованиям, но наиболее подходящим является протокол OpenVPN, у него наилучшее соответствие требованиям и есть открытый исходный код, который можно доработать для полного соответствия требованиям. Разработка нового «Протокола 125» тоже даст полное соответствие требованиям. Окончательное решение об использовании протокола OpenVPN

или разработки нового можно будет принять после рассмотрения существующих шифраторов.

2.2.2 Подбор модуля шифратора

Рассмотрим ключевые возможности популярных шифров, которые используются в различных протоколах обмена информацией и для шифрования данных. А также, добавим к сравнению не широко распространенный, но перспективный алгоритм с переменной фрагментацией подблоков шифрования «Шифратор 125» (Таблица 2.2).

Таблица 2.2 – Сравнение возможностей шифраторов

| Шифр | Блок, бит | Ключ, бит | Возможность изменения таблиц замен | Количество раундов | Переменная фрагментация блоков замен и сложения с ключом |
|------------------------------|-------------|--------------------|------------------------------------|--------------------|--|
| IDEA | 64 | 128 | нет | 8,5 | нет |
| SEED | 128 | 128 | нет | 16 | нет |
| DES, 3DES, DESX | 64 | 64, 112, 168, 184 | нет | 16, 48 | нет |
| Camellia | 128 | 128, 192, 256 | нет | 18, 24 | нет |
| ГОСТ Р 34.12-2015 (Кузнечик) | 128 | 256 | нет | 10 | нет |
| Twofish | 128 | 128, 192, 256 | нет | 16 | нет |
| AES | 128 | 128, 192, 256 | нет | 10, 12, 14 | нет |
| RC2,4 (arcfour),5,6 | 32, 64, 128 | 0-2040 | нет | 1-255 | нет |
| ГОСТ 28147-89 (Магма) | 64 | 256 | да, 4 бит | 16, 32 | нет |
| Blowfish | 64 | 32-448 | да, 8 бит | 16 | нет |
| CAST5, 6 | 128 | 128, 160, 192, 224 | да, 8 бит | 48 | нет |
| Шифратор 125 | 240 | 0-5040< | да, 0-240 бит | 0-21< | да |

Сразу можно сделать вывод (Таблица 2.2), что предоставить лучшую защиту от взлома перебором, с последующей фильтрацией или отбором по уменьшенному множеству при помощи различных закономерностей, уязвимостей шифров, может только «Шифратор 125», благодаря большему количеству свободных входных параметров

при помощи больших замен, переменной фрагментации подблоков и других изменяемых параметров алгоритма, которые не могут предоставить другие шифры.

Высокой степенью защиты обладают алгоритмы, позволяющие менять блоки замен, например, ГОСТ 28147-89 (Магма), Blowfish, CAST, но в таких реализациях, как правило блоки замен не выносятся в отдельный ключ, а используются внутри организации или структуры без изменения (под секретностью), что позволяет внутренним пользователям осуществить более простой взлом шифра. Также, существующие реализации не обладают большими блоками замен (4, 8 битные размерности), что сегодня и, особенно, в будущем сможет серьезно сказаться на их безопасности.

Все рассмотренные шифры, даже используемые в последней и разрабатываемой версиях TLS имеют много уязвимостей, связанных не только со статическим алгоритмом и отсутствием большого количества свободных входных параметров алгоритма, но и с неправильной реализацией протокола обмена информацией, который использует TLS.

Например, известны атаки на новые вариации шифров SSLv3 со сцепкой CBC [22, 23], которая была введена для защиты от атак частотного анализа и других закономерностей большого количества блоков, но привела к другим атакам при помощи анализа ответа сервера и изменения битов (оракул), которые позволяют менее, чем за час получить, например, данные файлов cookie сессии в открытом виде при помощи отправки запросов на сервер.

Криптоаналитики предлагают следующее решение: после расшифрования сразу же проводить аутентификацию сообщения [24], что реализовано в «Протоколе 125». Помимо этого, в нем происходит постоянная замена ключевой информации (по заранее определенному объему данных). Шифр не только шифрует, но и проверяет целостность по зашифрованному хэшу, совместно с аутентификацией по отдельному служебному полю.

Данный протокол находится на сеансовом уровне, что позволяет его использовать на любом устройстве, где есть поддержка стека протоколов TCP/IP, а это есть практически на любых компьютерах и мобильных устройствах.

Протокол защищен от атаки на доступность лучше, чем другие рассмотренные решения, потому что прежде чем расшифровать все сообщение, сначала расшифровываются только служебные поля и только после корректного хэша по служебным полям и аутентификации данных, расшифровывается оставшееся сообщение.

В протоколе практически невозможно реализовать преднамеренное разъединение сессии в отличие от WPA2 и осуществить подбор исходных ключей по информации, полученной при инициализации сессии.

Протокол, в целом, имеет дополнительную защиту от коллизий. Во-первых, потому что размещен над транспортным уровнем, где на более мелких сегментах проверяется открытый хэш. Во-вторых, протокол проверяет свой собственный зашифрованный служебный хэш, а после еще и зашифрованный аутентификатор. Но и это не все, после этого он проверяет тип сообщения, которое тоже имеет защиту от коллизий и в случае нарушения, отфильтрует его. Для данных существует отдельный хэш только по данным.

В итоге, зашифрованный хэш проверяется два раза. По служебному полю и по данным, в случае их наличия. Это позволяет лучше защитить от атаки отказа в обслуживании, как было сказано ранее, и значительно уменьшить вероятность принятия коллизии. Скорость алгоритма, при этом, меняется меньше всего, потому что для хеширования используется простое побитовое сложение блоков 240 бит, это стало возможным благодаря тому, что хэш зашифрован.

Данный протокол имеет свой независимый режим с уведомлениями и без уведомлений, который можно комбинировать с режимами TCP, UDP транспортного уровня. Также, в протоколе есть встроенное тестирование случайных последовательностей и шифртекстов (один из тестов был доработан), которые признаны лучшими среди большого количества различных тестов [25-28].

«Шифратор 125», на котором основан «Протокол 125», позволяет за счет минимальной избыточности, зависимой от параметра объема данных на один ключ и применением режима сцепки в шифровании, добиться максимальной надежности шифра близкой к абсолютной надежности без необходимости дальнейшего увеличения размера исходного блока данных, ключа шифрования и количества раундов.

По результатам сравнения, можно сказать, что «Протокол 125», который опирается на «Шифратор 125», позволяет добиться значительных результатов повышения безопасности конфиденциальности, целостности, доступности обмена информацией по сравнению с существующими широкораспространенными аналогами из разных сфер применения.

Вариант разработки нового протокола предпочтительнее доработки OpenVPN, потому что позволяет добиться лучших результатов особенно при использовании нового «Шифратора 125» и адаптация протокола непосредственно только под этот шифратор даст дополнительные преимущества. А доработку OpenVPN всегда можно использовать для создания «быстрого» протокола безопасного обмена информацией, если в этом будет необходимость.

2.3 Описание «Шифратора 125»

2.3.1 Внесенные изменения в алгоритм шифра

В работе [29] впервые в открытой печати представлен алгоритм шифрования с динамическим изменением размеров криптографических примитивов в различных раундах. Иными словами, предлагается проводить зашифрование текста, применяя замены по таблицам разных размеров в различных раундах.

В [29] проведены первые исследования, тестирование результатов зашифрования. В настоящей работе производится анализ модификации алгоритма [30, 31]. Данная модификация адаптирована для использования в «Протоколе 125» с целью достижения лучших характеристик безопасности канала обмена информацией, а именно: в алгоритме производится нормировка и сцепка блоков; используется раундовое преобразование над блоками размером 240 бит; длина общего ключа (ОК) алгоритма может быть от 0 до $240 \cdot (\text{количество раундов})$ бит, другими словами, ОК может быть уникальным для каждого раунда; замена осуществляется по неприводимым многочленам [32, 33] в зависимости от длины подблока; динамическое изменение криптографических примитивов (подблоков) производится на блоки не одинаковой длины в пределах одного раунда, а произвольной, обеспечивая любые возможные комбина-

ции; также предлагается производить линейный сдвиг после замены не на постоянную величину, а на случайную; аналогично предыдущему улучшению происходит сдвиг общего ключа случайно для каждого раунда. Ключевые отличия и итоговая схема шифрования показаны ниже (Рисунок 2.10).

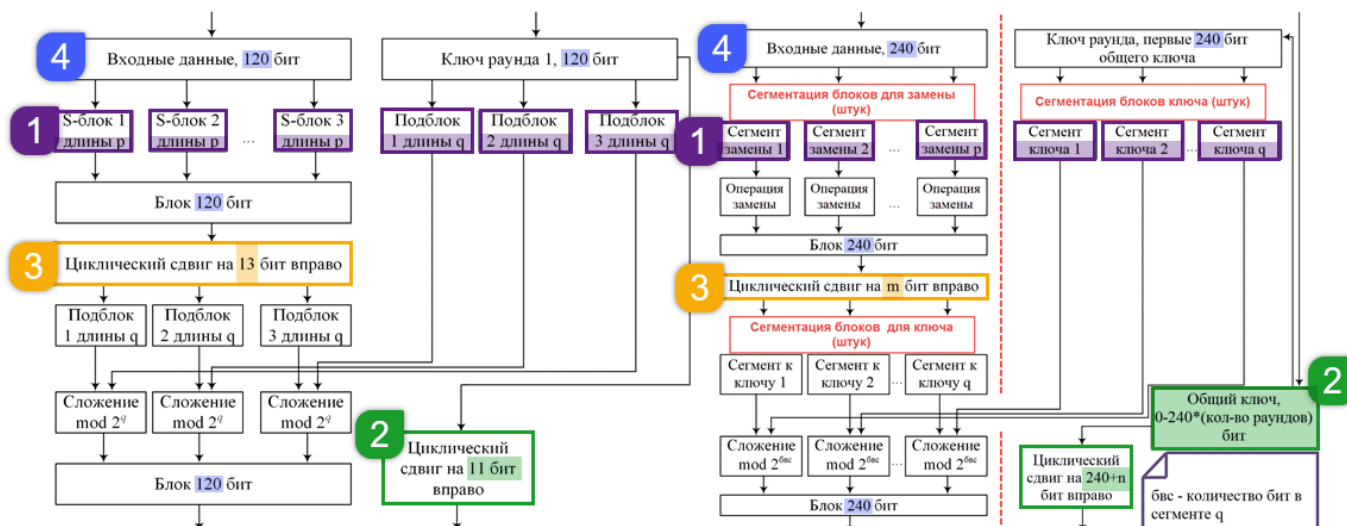


Рисунок 2.10 – Сравнение схемы предыдущего алгоритма шифрования с новым

Можно выделить следующие пять ключевых отличий, первые четыре из них пронумерованы под соответствующими номерами (Рисунок 2.10):

- 1) подблоки различной длины относительно друг друга;
- 2) возможность создания уникального ключа для каждого раунда;
- 3) сдвиг на случайное число;
- 4) увеличения размер блока с 120 бит до 240 бит;
- 5) полный отказ от таблиц замен в пользу неприводимых многочленов.

Скорость шифра после изменений 1, 2, 3, 5 осталась прежней, при этом, произошло увеличение количества свободных входных параметров и незначительное увеличение длины конфигурационного ключа.

Помимо изменения схемы работы алгоритма шифрования, в «Шифратор 125» для его исправной работы была добавлена нормировка. В том числе, алгоритм был усилен режимом сцепки блоков с вектором инициализации.

2.3.2 Принципы работы измененного алгоритма

Как уже было сказано ранее, практически все симметричные шифры, существующие на сегодняшний момент, работают с небольшим количеством входных параметров: есть возможность изменения ключа и, в лучшем случае, таблиц замен и количества раундов шифрования [34, 35].

Рост надежности подобных шифров можно обеспечить только расширением длины блока шифрования с одновременным увеличением размера ключа. Таким способом, при неизменном количестве входных параметров, можно добиться только увеличения множества перебора одного блока в отдельности, но итоговое сообщение, как правило состоит из огромного множества таких блоков.

При этом, если у нас имеются возможности для осуществления перебора и, хотя бы одна пара исходного и шифртекста (определенных, например, по известной структуре протокола), то мы можем просто перебрать все варианты ключа, до тех пор, пока не получим нужный исходный текст. Чтобы избежать подобных ситуаций осуществляется засекречивание алгоритма шифрования, либо полностью, либо, например, только таблиц замен. Таблицы замен, в таком случае, выступают в качестве дополнительных входных параметров алгоритма шифрования и практически полностью упрощают взлом для тех, кому они известны. Даже для легальных пользователей, которые смогут получить ключи шифрования других пользователей осуществив, затем, простой перебор. Но даже, если мы не знаем таблицы замен, но обладаем необходимыми мощностями для перебора, то, теоретически, можем перебрать все множество таблиц замен и отфильтровать их по известным блокам исходного и шифртекста, которые связаны между собой порядком следования, потому что таблицы замен имеют малую размерность 4 или 8-битную.

Все это оставляет гипотетическую возможность для осуществления взлома алгоритма шифрования. Для того, чтобы этого избежать, оставив при этом полностью открытым сам алгоритм шифрования, необходимо менять последовательность не только ключа, но и всех остальных параметров алгоритма.

Все это удалось реализовать в «Шифраторе 125» [30, 31] для ОС Windows и Linux, который позволяет изменять случайным образом такие параметры, как общий

ключ, сдвиг блока и ключа, две последовательные фрагментаций блока для замены и сложения с ключом раунда, способ и вид самой замены. Скорость шифрования на данный момент составляет 120 кбайт/с (примерно, 1 Мбит/с). В будущем, возможно увеличить данный показатель при помощи низкоуровневого или аппаратного программирования, распараллеливания вычисления блоков на графических процессорах.

Рассмотрим подробнее «Шифратор 125». Так как он является блочным, он требует нормировки (Рисунок 2.11). Нормировка в нем производится при помощи нормировочного блока, в начало которого добавляется число равное количеству лишних случайных битов, которые были добавлены в сообщение в результате обеспечения его кратности числу 240 (размер блока) с учетом размера числа о количестве лишних нормировочных бит. Далее следуют, при наличии, лишние случайные биты и первые биты полезных данных.

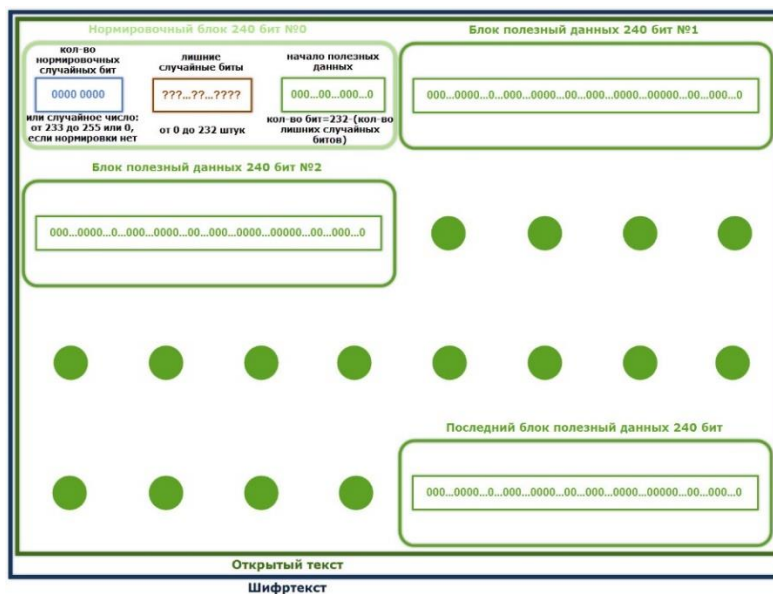


Рисунок 2.11 – Схема нормировки

Благодаря тому, что лишние нормировочные биты – случайны, они вносят шум в исходное сообщение и для злоумышленника являются непредсказуемыми, он не сможет произвести фильтрацию по заранее известным значениям лишних битов множества сообщений при переборе ключей. При этом, добавляя эти лишние случайные биты в начало сообщения, а не в конец, мы добиваемся непредсказуемого смещения начала исходного сообщения в 232 варианта, это сможет повысить надежность шифра, если длина исходного сообщения злоумышленнику не известна.

В том числе, при попытке расшифровки злоумышленник не сможет фильтровать по незадействованным значениям параметра о количестве лишних нормировочных бит в диапазоне от 233 до 255, потому что они тоже являются корректными и, как и ноль, означают отсутствие лишних нормировочных бит. Одно из этих чисел задается случайным образом в ситуации отсутствия нормировки.

Процесс шифрования раунда (Рисунок 2.12) осуществляется над блоками размером 240 бит. Длина общего ключа (ОК) алгоритма может быть от 0 до $240 \cdot (\text{количество раундов})$ бит, другими словами, ОК может быть уникальным для каждого раунда.

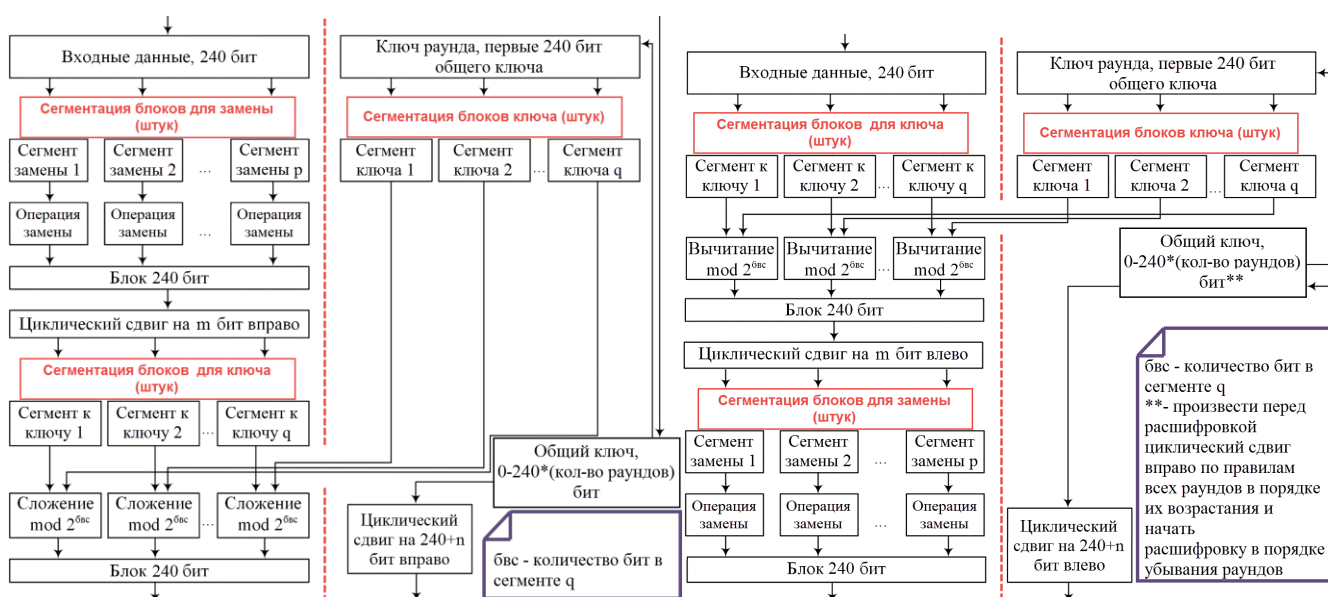


Рисунок 2.12 – Схема шифрования и расшифрования «Шифратора 125»

Первым шагом раунда, происходит фрагментация блока 240 бит, которая разбивает его на подблоки произвольной длины (в битах) и произвольного количества (в штуках), в сумме дающих 240 бит. Затем, каждый полученный подблок отдельно заменяется по таблице замен или по неприводимому многочлену [32, 33] в зависимости от его длины.

После этого этапа, подблоки объединяются и происходит циклический сдвиг вправо на заданное количество бит. Далее, блоки второй раз уникально фрагментируются на подблоки произвольной длины и произвольного количества для того, чтобы произвести сложение. В ОК берутся первые 240 бит и фрагментируются аналогичным

образом. Происходит сложение по модулю $2^{\text{кол-во бит в сегменте}}$ каждого в отдельности подблока ключа раунда (КР) с подблоком исходного блока.

Теперь, подблоки вновь объединяются в один блок 240 бит. Происходит циклический сдвиг ключа на $240+n$ бит вправо, где n – заданное дополнительное количество бит сдвига (по умолчанию равно нулю).

Замена блока по неприводимому многочлену [32, 33] позволит перетасовать комбинации идущих последовательно блоков, создавая тем самым цепочки текста из одних и тех же блоков, но разной последовательности, а, следовательно, и содержания. Предварительная переменная фрагментация блока обеспечит значительный прирост вариативности таких комбинаций перетасовок цепочек, количества различных текстов на выходе (Рисунок 2.13).

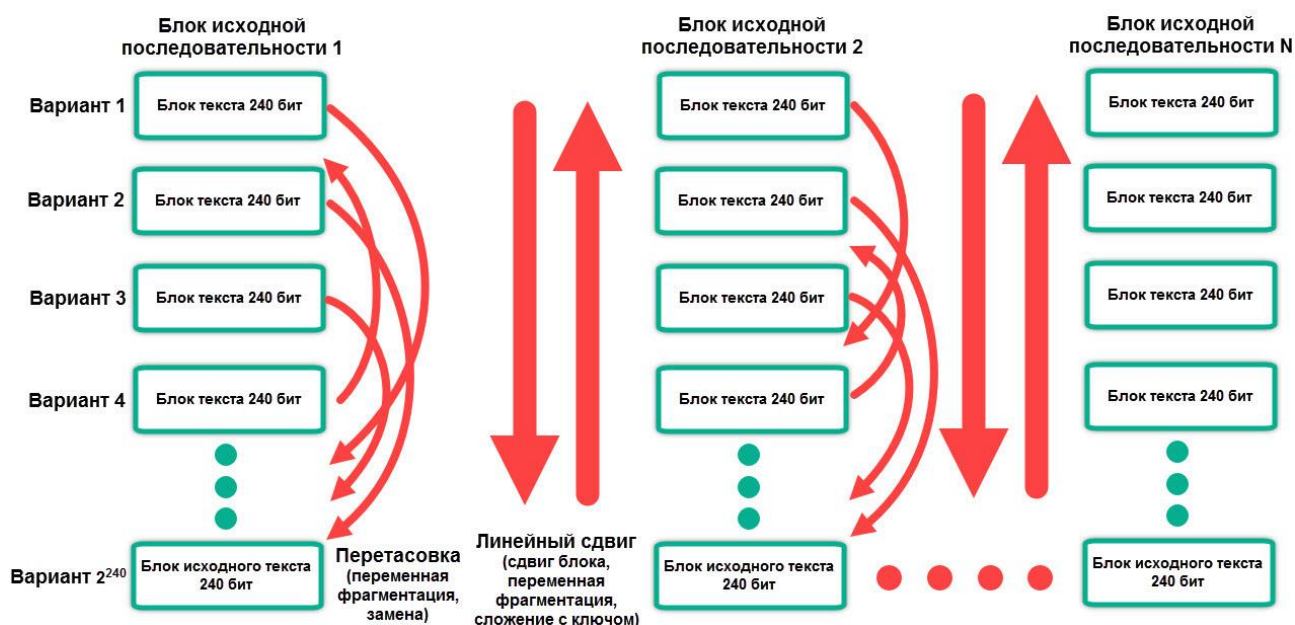


Рисунок 2.13 – Смещение комбинаций цепочек последовательностей при расшифровке

Последующий сдвиг блока и переменная фрагментация перед сложением с ключом обеспечит линейный сдвиг комбинаций блоков относительно друг друга, что в совокупности с переменной фрагментацией перед заменой обеспечит достаточное количество уникальных комбинаций последовательно идущих блоков текстов, чтобы обеспечить невозможность после атаки перебора, определить какой из множества текстов является истинным.

Злоумышленник будет получать множество текстов, которые выполняют все условия фильтрации. Например, при фильтрации по структуре протокола останется множество комбинации с разными значениями поля данных, а возможно и все множество комбинаций поля данных.

Чем больше у злоумышленника последовательно связанных пар исходных и зашифрованных текстов, тем проще осуществить взлом шифра, но их получить достаточно трудно, а длина пакета протокола обычно не большая.

Тем более, есть возможность увеличения количества раундов, использование временных ключей «Протокола 125» на определенный объем данных и сцепки с вектором инициализации (Рисунок 2.14).

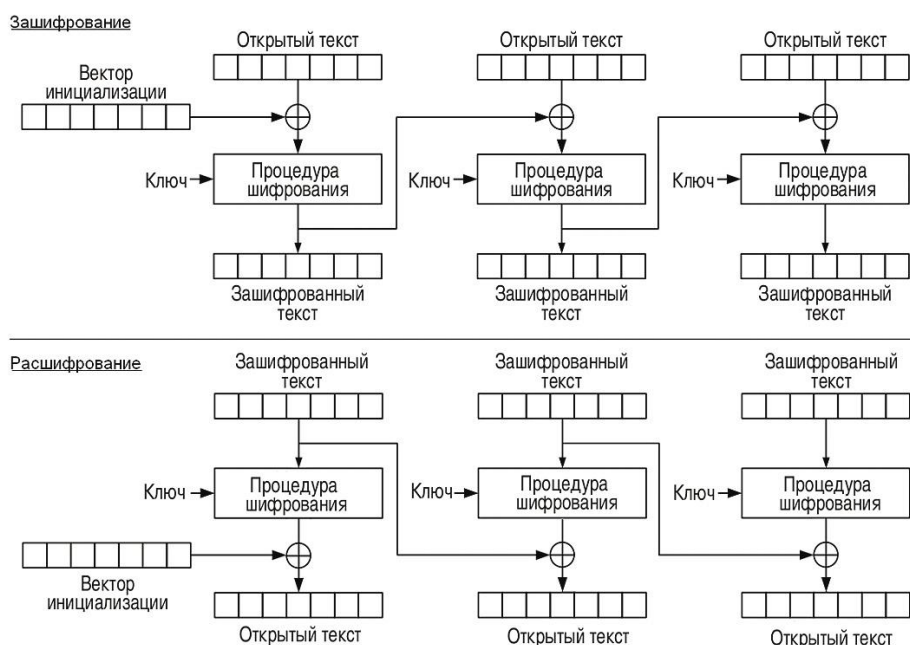


Рисунок 2.14 – CBC

Также, если этого будет недостаточно, можно всегда увеличить длину блока или использовать устройства большей вычислительной размерности (например, трюичные компьютеры [36]).

2.4 Описание «Протокола 125»

Если предлагать защищенный обмен информацией, то нужно говорить о правилах, по которым будет передаваться информация. Обычно таким автоматизированным набором правил является протокол.

Что лежит в основе безопасного протокола? Конечно, это шифратор. Поэтому определяясь с основными особенностями протокола безопасного обмена информацией, также необходимо определяться и с основными особенностями шифратора, потому что это две взаимосвязанные вещи. На основе «Шифратора 125» возможна реализация безопасного протокола передачи данных «Протокол 125» [30, 31].

Какие можно выделить параметры повышения безопасности передачи данных для «Протокола 125»? Сделаем перечень:

- длина блока или ключа шифрования;
- количество раундов шифрования;
- количество временных ключей на количество переданных зашифрованных данных.

Но не всегда возможно использовать эти параметры за пределами большими для повышения надежности шифртекста протокола безопасного обмена информацией, потому что существуют следующие ограничения:

- скорость передачи информации;
- избыточность передаваемых или хранимых данных.

Благодаря одной из главных особенностей «Шифратора 125», а именно, генерации конфигурационного ключа, можно подобрать приемлемые характеристики длины блока и ключа, количества раундов и временных ключей на определенный объем данных таким образом, что при практически максимальной скорости шифрования и минимальной избыточности шифртекста, будет получаться надежный шифр для последовательностей любой длины без необходимости дальнейшего увеличения количества раундов, длины ключа или блоков.

Определимся с основными функциями протокола, которые могут понадобиться для решения задачи безопасного обмена информацией в различных ситуациях (Рисунок 2.15).



Рисунок 2.15 – Основные функции протокола

Разрабатываемый протокол находится на сессионном уровне, в связи с этим, такую функцию, как коррекция ошибок сочтено не использовать, потому что ее применение в небольших блоках, после фрагментации, на транспортном уровне, полностью удовлетворяет основным требованиям качества связи, в том числе, данная функция не нуждается в шифровании.

Не имеет смысла использовать дополнительную адресацию и распараллеливание нескольких потоков данных. Это, также, реализовано на транспортном уровне при помощи портов. В крайнем случае, при необходимости, эту возможность можно дополнительно реализовать на уровне разработки программного кода и инкапсулировать в протокол безопасного обмена данными. Сообщения передают большой объем данных за один цикл обмена информацией и поэтому могут отправляться последовательно без нумерации.

Аналогично можно сказать и по поводу предварительного сжатия данных. «Шифратор 125» по результатам тестов, рассмотренных далее, хаотизирует большинство распространенных типов данных, например, текст в кодировке UNICODE или бинарные файлы различных программ. Для команд рекомендуется использовать слу-

чайные последовательности, не содержащие большое количество упорядоченных нулей или единиц. Но если результаты тестов после шифрования показывают плохие результаты для каких-либо специфических данных, то можно всегда дополнительно разработать эффективный алгоритм сжатия, оптимизированный непосредственно под этот тип информации.

Для выбора корректного алгоритма обработки информации на принимающей стороне, необходимо добавить в структуру протокола поле тип данных. Данное поле возможно защитить от коллизии при помощи создания для каждого типа пакета уникальной длины заголовка служебных данных.

Поле идентификатора позволяет выбрать необходимый симметричный ключ для расшифровки, из базы данных, затем, произвести расшифровку. В случае шифрования данных и служебных полей используется режим сцепки блоков «Шифратора 125». В этом случае, в качестве начальной сцепки (вектора инициализации) выступает поле идентификатора, которое не шифруется.

Функция хэширования здесь дублирует функцию хэша на транспортном уровне. Для защиты от нарушения доступности и повышения устойчивости от возникновения коллизий в протоколе используется два хэша: служебный хэш и хэш данных.

Хэш данных получается по всем отправляемым данным. Объем данных, по которому берется хэш, значительно больше, чем в сегментах транспортного уровня, что позволяет проверять целостность более укрупненно и дополнительно защитить от принятия сообщений с коллизией по хэшу. Но самое главное, данный хэш шифруется, в связи с чем, он позволяет дополнительно определить наличие попыток подмены отдельных участков сообщения, например, поля нового временного общего ключа, нарушение которого может вызвать потерю сессии или других полей, которые могут вызвать критический сбой конфигурации системы, выведению ее из строя.

Служебный хэш позволяет определить корректность заголовка и всех служебных полей до расшифровки полезных данных. Это усиливает защиту от коллизий и позволяет осуществить проверку поля тип данных по длине служебного пакета, а также, усилить защиту от вызова отказа в обслуживании.

Риск возникновения непреднамеренной коллизии остается, но он значительно ниже, чем при использовании проверки хэша только на транспортном уровне. Хэш в «Протоколе 125», несмотря на это, решает свою главную задачу, а именно сводит практически полностью к нулю вероятность проведения атаки на подмену участков сообщений.

Риск возникновения случайной коллизии в ходе проведения данной атаки остается, но он вряд ли приведет к быстрому результату, скорее всего, это произойдет один раз за значительный период времени, и каждая новая попытка в разы увеличит интервал времени появления следующей такой коллизии, что сделает данный способ практически не целесообразным. Например, вместо гарантированной возможности разъединения сессии посылкой всего одного сообщения, потребуется ждать неопределенное количество времени, за которое, скорее всего, произойдет запланированное отключение сессии легальными пользователями.

Возникновение коллизии редко, но она может появиться. Чаще всего это приводит к получению не корректных диапазонов данных, которые будут отфильтрованы, как ошибочные в программном приложении или протоколе, которые обрабатывают и интерпретирует полученные данные. Но возможна ситуация, когда коллизия создаст сообщение, в котором будут допустимые параметры. Данная ситуация возникает еще реже, но может привести к критичным последствиям для системы и вывести ее из строя, поэтому ее нужно обязательно учитывать.

Как полностью защититься от коллизии данных? Во-первых, нужно понимать, что не все виды информации нуждаются в защите от коллизий, потому что она происходит достаточно редко. Например, мультимедия потоки данных, телеметрия или некоторые команды, которые меняются с достаточно высокой частотой могут не нуждаться в дополнительной проверке на наличие коллизий. Но остальные команды или данные, которые редко обновляются и на их показателях работают другие критичные алгоритмы, которые могут успеть сделать непоправимые ошибочные решения из-за неправильных входных данных, нуждаются в такой проверке.

Как можно осуществить такую проверку? Решать данную задачу необходимо не на уровне протокола, а на уровне программы. Например, достаточно эффективным

способом будет последовательная отправка нескольких таких сообщений, например, 3-5 штук с одной и той же информацией. Принимающая сторона, когда получит сообщение с критически важной информацией 3-5 раз, произведет побитовое сравнение всех этих данных и если, хотя бы одно сообщение отличается от других, то будет отправлен запрос на повторную передачу данного сообщения еще 3-5 раз, до тех пор, пока данные не будут полностью совпадать.

Рассмотрим подробнее ситуацию, когда возможна коллизия хэша в служебных полях протокола и к каким последствиям это может привести. В отличие от данных, коллизия в служебных полях должен корректно обрабатывать именно протокол.

Если коллизия произойдет в хэше, то сообщение будет отфильтровано, это вызовет повторную передачу сообщения у отправляющей стороны. Если после определенного количества попыток уведомление не приходит, то пакет заново формируется, чтобы исключить возможные ошибки при формировании пакета и отправляется снова несколько раз. После этого, если не приходит уведомление, считается, что потеряна сессия и происходит отправка сообщения о запросе новой сессии. Если после определенного количества запросов сессии ответы не приходят, считается недоступной принимающая сторона и соединение потеряно.

Если коллизия произошла в аутентификаторе возвращается ошибка аутентификации, на которую протокол совершает определенное количество попыток повторной аутентификации и, если это не помогает, то происходят серьезные помехи в канале связи и соединение прекращается.

В случае если коллизия коснулась новых временных общих ключей, то при следующей попытке отправки нового сообщения расшифровка на принимающей стороне даст некорректные данные и сообщение будет отфильтровано без отправки уведомлений. Потом, как описывалось ранее, после определенного количества попыток безуспешного завершения отправки данного пакета будет сформирован повторно пакет, затем, потеряна сессия и при отправке запроса новой сессии будут заменены некорректные временные общие ключи, и сессия восстановится.

Коллизия наиболее опасна на этапе установления сессии. В случае коллизии в предсохраненной имитовставке, на принимающей стороне сообщение отфильтруется

и вызовет повторную отправку данного пакета на отправляющей стороне, после определенного количества попыток, если это не поможет будет потеря соединения. Здесь все происходит нормально.

Но самое опасное, если коллизия случится при отправке новой предсохраненной имитовставки. В случае передачи данного поля с ошибкой, установить новое соединение и сессию больше не удастся без ручной замены предсохраненной имитовставки.

Для защиты от подобной ситуации, необходимо сохранять старую предсохраненную имитовставку до успешного установления новой сессии. После успешного установления новой сессии производить замену старой предсохраненной имитовставки на предсохраненную имитовставку использованную при инициализации данной сессии. Для обеспечения безопасности и неповторимости старую предсохраненную имитовставку перед повторным использованием необходимо циклически сдвинуть на определенное количество нечетных бит. Если произойдет третье использование одной и той же старой предсохраненной имитовставки, то процедуру необходимо повторить, используя тот же самый нечетный сдвиг, что был использован ранее.

Если коллизия коснется сессионных общих ключей приема или передачи в режиме без уведомления, то это может привести к такой ситуации, что при отправке данных без уведомления они никогда не дойдут до адресата и определить это не удастся.

Для того, чтобы этого избежать, при инициализации сессии, после отправки сессионных общих ключей приема и передачи в режиме без уведомления, необходимо послать проверочные пакеты, зашифрованные этими ключами с уведомлением, по одному разу для ключа приема и ключа передачи.

Если произойдет сбой, то будет отправлен запрос на новую сессию. Другими словами, без предварительной проверки корректности передачи сессионных общих ключей для корректной работы в режиме без уведомления, сессия не установится.

В тестовое сообщение необходимо поместить значение синхронизированного таймера и при получении данного сообщения провести полную процедуру проверки

имитовставки. Если таймер будет неправильно отсинхронизирован в результате коллизии, то сообщение будет постоянно отфильтровываться, что вызовет повторную инициализацию сессии.

Аналогично необходимо поступать, если требуется в пределах одной сессии дополнительно отсинхронизировать таймеры и заменить сессионные общие ключи приема и передачи для режима без уведомлений. После данного сообщения необходимо убедиться в отсутствии коллизии, передав тестовые сообщения режима без уведомления с ожиданием уведомления об их доставке.

Для дополнительного подтверждения владения корректным ключом наряду с хэшем и проверкой достоверности идентификатора служебного сообщения необходимо добавить в структуру протокола поле аутентификатора. Оно проверяется сразу после проверки хэша. В отличие от хэша данная процедура практически не ресурсоемка и позволяет уменьшить вероятность возникновения коллизии аутентификации и значительно уменьшить вероятность вызова коллизии подменной идентификатора на заведомо недействительный идентификатор. Это значительно увеличивает время, за которое такая коллизия может появиться, что в совокупности с процедурой проверки хэша, сделает данную атаку практически не целесообразной.

Имитовставка в «Протоколе 125» осуществляется, в случае отправки сообщения с уведомлением, при помощи генерации нового временного общего ключа на каждую пару сообщение/уведомление, а в случае отправки сообщения без уведомления, при помощи добавления поля со значением синхронизированного таймера и дополнительным случайным числом в лишние битах блока для нарушения закономерностей таймера.

Шифрование производится при помощи «Шифратора 125». Технология «VPN» в первом варианте «Протокола 125» реализована не будет, но стоит описать, как это будет реализовано в будущем.

Так как «Протокол 125» предполагается использовать только с шифрованием данных, то в нем получится избежать двойного шифрования при использовании VPN, потому что исходные данные пользователя не будут повторно шифроваться.

При помощи отдельного конфигурационного ключа общего для всех участников VPN соединения будет дополнительно зашифрована следующая информация: открытый идентификатор сети (аналогично, как у пользователя), параметры транспортного и сетевого уровней.

Вся сессия VPN будет установлена подобно пользовательской сессии «Протокола 125», но только между узлами в реальной сети. Другими словами, в роли пользователей выступают конкретные узлы VPN соединения с уникальным конфигурационным ключом, который общий для всех участников данной сети. Пакеты от пользователей будут помещаться в поле данных подобной сессии VPN.

Поле данных будет содержать следующую информацию: открытый идентификатор пользователя (зашифрованный конфигурационным ключом VPN сети), шифротекст пользователя (зашифрованный только конфигурационным ключом пользователя), параметры транспортного и сетевого уровней (зашифрованные конфигурационным ключом VPN сети).

По приходу данных на другой узел VPN пакет будет расшифрован для принятия пакета или его переадресации (продвижения) для этого потребуются параметры сетевого и транспортного уровней и открытый идентификатор пользователя зашифрованного сообщения сеансового уровня «Протокола 125», но нет необходимости в расшифровке служебных полей и данных пользователя, в том числе, знать ключ расшифровки может только конечных адресат.

Другими словами, будет полностью восстановлен сетевой пакет сети внутри VPN. В случае многопользовательской системы понадобится сервер-координатор-маршрутизатор данного VPN соединения. В реальной сети пакеты будут адресованы на него. В случае соединения точка-точка, в реальной сети будут указаны известные данные конечного узла-адресата VPN соединения.

Подробно ознакомиться с процедурой инициализации сессии «Протокола 125» можно в «Приложении А». Теперь рассмотрим способы разрешения коллизий отправки сообщений, приоритеты типов сообщений и правила обработки оставшихся некорректных ситуаций в «Протоколе 125».

В «Протоколе 125» две передающие стороны могут находиться в следующих состояниях: ожидание сообщения данных, ожидание уведомления, вещание сообщения данных с уведомлением, вещание сообщения данных без уведомления, вещание сообщения замены временных ключей, вещание сообщения временных ключей без уведомления, вещание сообщения замены параметров без уведомления, вещание проверки передачи сессионного общего ключа передачи и приема для режима без уведомления.

После вещания сообщения ключей, параметров, проверок, данных передающая сторона переходит в режим ожидания уведомления. Но, если в этот момент ей поступает на вход аналогичное сообщение ключей, параметров, проверок или данных (некорректные уведомления просто отфильтровываются), то передающая сторона поступает следующим образом:

- если приоритет сообщения выше встречного сообщения, то передающая сторона вещает это сообщение без дополнительной случайной задержки, используя лишь стандартную задержку повторной посылки сообщения в случае не получения уведомления;
- если приоритет ниже или равен встречному сообщению, то передающая сторона добавляет к интервалу повторной посылки пакета случайное число в пределах заданного диапазона случайной задержки.

Аналогично поступает и удаленная сторона сеанса. Таких ситуаций с уведомлениями быть не может. В остальных спорных или некорректных ситуациях сообщения отбрасываются и по необходимости передаются повторно со стандартной задержкой повторной посылки сообщения в случае неполучения уведомления. Если определенное количество попыток отправить сообщение не приводят к успеху, сессия считается потерянной и отправляется запрос на новую сессию. Если произошла, например, коллизия, то установится новая сессия. Но если и попытки установить новую сессию не приведут к успеху, то считается, что соединение потеряно.

Приоритеты иницирующих сообщений, в порядке уменьшения приоритета:

- сообщение замены временного общего ключа;
- сообщения замены параметров режима без уведомлений;

- сообщение проверки передачи сессионного общего ключа приема или передачи для режима без уведомлений;
- сообщения данных;
- сообщение временного общего ключа для сообщения с данными в режиме без уведомления;
- сообщения данных в режиме без уведомления.

2.5 Выводы

Были получены результаты анализа безопасности широкораспространенных протоколов защищенного обмена информацией. Составлена сводная таблица по соответствию протоколов требованиям, определенным в первой главе. Также, представлена сводная таблица сравнения характеристик шифраторов широкораспространенных протоколов.

После сравнения, было определено, что ни один из протоколов не соответствует полностью всем требованиям. Наиболее подходящим является протокол OpenVPN. Но и он не соответствует всем требованиям. В связи с этим было решено разрабатывать новый шифратор и протокол. Для них было дано подробное описание принципов их работы.

3 КОНФИГУРИРОВАНИЕ ПРОТОКОЛА, РЕЛИЗАЦИЯ И ТЕСТИРОВАНИЕ ШИФРАТОРА И ГЕНЕРАТОРА КЛЮЧЕЙ

3.1 Реализация «Шифратора 125» и генератора ключей

3.1.1 Кроссплатформенная программа

На основе вышеописанного алгоритма разработана программа-шифратор (Рисунок 3.1). Конфигурационный ключ (КК) программы состоит из ОК и конфигурационных настроек шифратора.

Одной из главных функций программы является генератор КК. Он позволяет случайным образом для каждого раунда генерировать ОК, сдвиг блока и ключа, задавать все параметры двух последовательных фрагментаций блока для замены подблоков и сложения с ключом раунда (КР). После каждой генерации КК производится тесты на качество шифрования по заданным критериям оценки, в результате которых можно принимать решение о повторной генерации ключа или о добавлении, уменьшении количества раундов.

Сохраненный КК можно использовать в программе или для подачи на вход библиотеке шифрования данного шифратора, программе-терминалу в случае автоматизации процесса шифрования, например, для связи с удаленной автоматической системой по беспроводному каналу или для шифрования в «Протоколе 125» [30, 31].

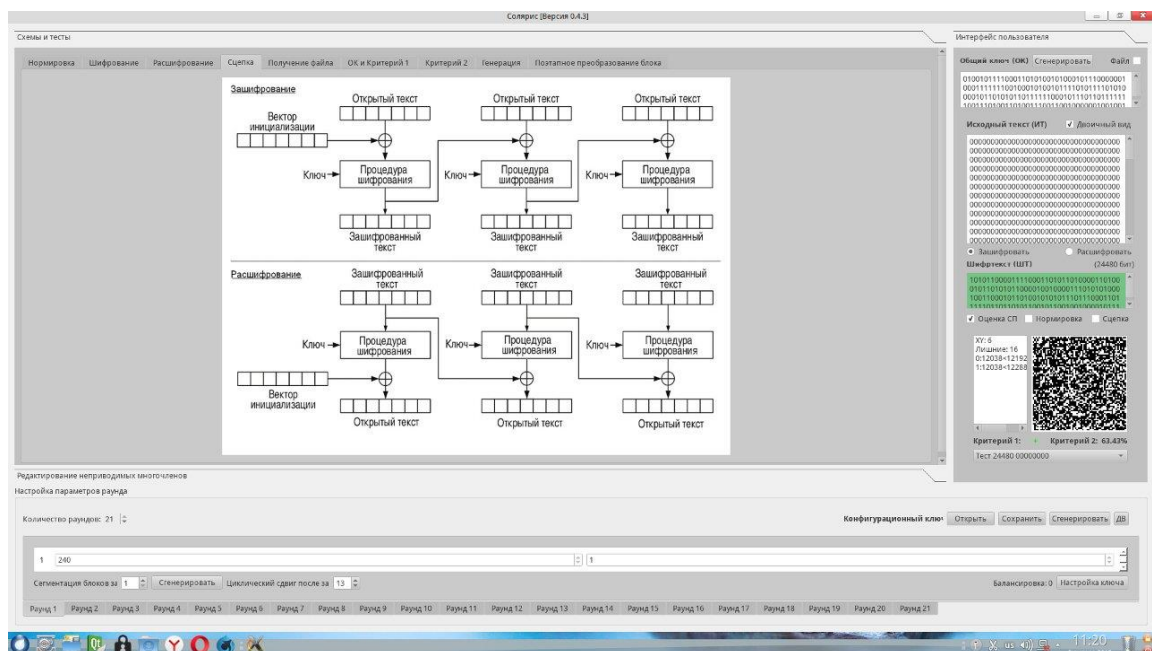


Рисунок 3.1 – Интерфейс программы «Шифратора 125»

КК программы состоит из ОК и конфигурационных настроек алгоритма шифрования. Одной из главных функций программы является генератор КК и ОК. Он позволяет случайным образом для каждого раунда генерировать ОК, сдвиг блока и ключа, задавать все параметры двух последовательных фрагментаций блока для замены подблоков и сложения с КР. После каждой генерации КК производятся тесты на качество шифрования по заданным критериям оценки.

Дополнительной функцией программы будет поэтапная проверка шифрования блока, когда на вход подается блок 240 бит и по заданной конфигурации происходит поэтапное преобразование блока с отображением всех промежуточных шагов. Также, есть и другие вспомогательные функции.

Данное программное обеспечение позволит исследовать конфигурации симметричных алгоритмов без переменной фрагментации блока на качество работы, сохранять КК для подачи на вход библиотеке «Шифратора 125» или программе-терминалу.

3.1.2 Тестирование скорости работы

Программа на данный момент позволяет шифровать текстовые сообщения и любые файлы со средней скоростью 120 килобайт/с (Рисунок 3.2, Рисунок 3.3). В будущем, необходимо произвести оптимизацию программы для увеличения показателя скорости преобразования при помощи низкоуровневого или аппаратного программирования, распараллеливания и графических процессоров.

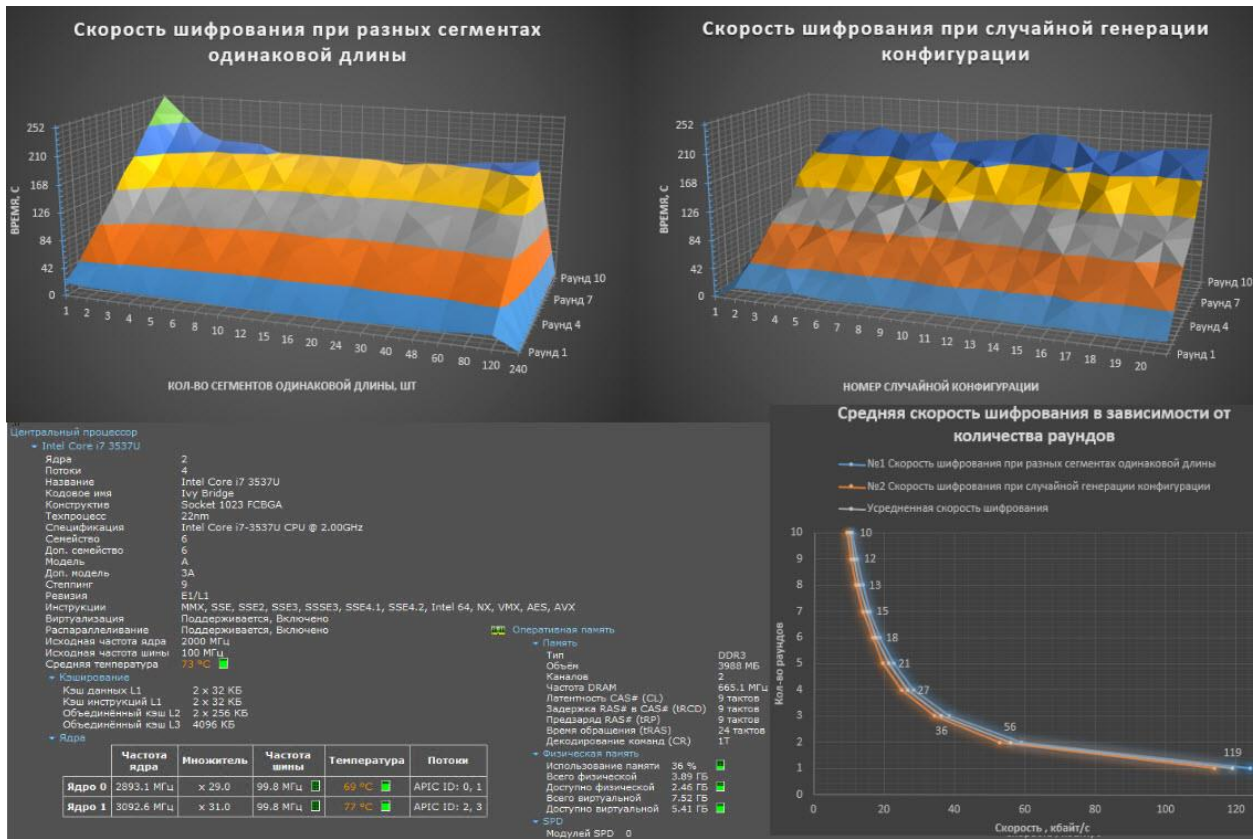


Рисунок 3.2 – Графические результаты анализа скорости и условия эксперимента

| №1 Скорость шифрования при разных сегментах одинаковой длины | | | | | | | | | | | | | | | | | | | |
|--|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|-------|
| Кол-во сегментов одинаковой длины, шт | | | | | | | | | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 8 | 10 | 12 | 15 | 16 | 20 | 24 | 30 | 40 | 48 | 60 | 80 | 120 | 240 |
| 20,530 | 16,973 | 16,895 | 16,115 | 15,909 | 15,521 | 15,787 | 14,593 | 14,369 | 15,005 | 14,185 | 15,351 | 14,383 | 14,756 | 16,532 | 15,790 | 15,567 | 18,425 | 19,127 | 1,264 |
| 38,310 | 38,369 | 36,937 | 35,505 | 33,272 | 32,860 | 32,780 | 31,102 | 31,385 | 33,446 | 32,382 | 31,623 | 30,742 | 32,524 | 32,762 | 32,780 | 33,439 | 36,210 | 37,738 | 1,990 |
| 64,518 | 61,767 | 54,351 | 53,154 | 52,477 | 51,505 | 50,218 | 48,977 | 47,454 | 49,410 | 49,034 | 47,444 | 46,198 | 48,897 | 49,821 | 49,936 | 50,396 | 54,148 | 57,739 | 2,626 |
| 91,775 | 80,821 | 73,964 | 71,230 | 69,470 | 71,980 | 66,314 | 66,750 | 64,333 | 68,372 | 67,008 | 65,495 | 62,534 | 65,672 | 67,794 | 66,945 | 67,607 | 72,896 | 73,724 | 3,182 |
| 117,540 | 100,863 | 92,546 | 89,465 | 88,929 | 87,847 | 84,032 | 81,441 | 80,758 | 83,352 | 84,849 | 80,604 | 81,066 | 81,089 | 83,630 | 85,301 | 85,845 | 89,660 | 92,089 | 4,121 |
| 148,329 | 126,273 | 112,019 | 111,874 | 107,833 | 104,993 | 100,013 | 101,528 | 98,775 | 98,596 | 99,068 | 99,204 | 96,330 | 97,159 | 104,461 | 100,587 | 101,399 | 108,764 | 110,469 | 4,531 |
| 169,132 | 141,572 | 133,826 | 127,565 | 126,560 | 122,942 | 120,818 | 115,141 | 113,623 | 117,606 | 115,297 | 116,709 | 114,735 | 114,813 | 115,882 | 117,094 | 118,186 | 130,423 | 128,940 | 5,328 |
| 200,703 | 167,051 | 153,854 | 147,008 | 143,002 | 145,157 | 133,575 | 135,162 | 133,614 | 132,866 | 135,515 | 131,777 | 131,253 | 130,501 | 133,342 | 135,652 | 137,722 | 142,975 | 155,201 | 5,820 |
| 227,774 | 190,602 | 176,809 | 164,103 | 166,459 | 160,799 | 159,915 | 148,215 | 147,101 | 148,705 | 149,734 | 146,597 | 147,722 | 145,443 | 151,693 | 153,429 | 157,332 | 170,065 | 168,375 | 6,485 |
| 251,785 | 221,511 | 192,947 | 182,054 | 181,933 | 179,907 | 167,390 | 167,042 | 169,633 | 167,789 | 169,442 | 166,344 | 162,716 | 166,979 | 167,462 | 172,776 | 175,989 | 178,751 | 184,777 | 7,164 |

| №2 Скорость шифрования при случайной генерации конфигурации | | | | | | | | | | | | | | | | | | | |
|---|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| Номер случайной конфигурации | | | | | | | | | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 16,598 | 17,983 | 16,525 | 19,358 | 16,605 | 15,567 | 16,318 | 16,790 | 16,093 | 17,781 | 16,865 | 17,080 | 17,313 | 16,099 | 16,032 | 17,370 | 15,590 | 15,932 | 16,097 | 16,526 |
| 36,189 | 35,722 | 39,844 | 33,171 | 33,762 | 36,162 | 37,580 | 35,312 | 35,575 | 39,634 | 39,879 | 35,708 | 36,694 | 34,123 | 33,180 | 39,393 | 36,078 | 33,342 | 34,613 | 36,383 |
| 53,378 | 55,994 | 59,613 | 57,518 | 56,993 | 56,715 | 52,181 | 52,383 | 55,331 | 59,117 | 52,718 | 54,612 | 51,818 | 56,384 | 51,268 | 55,028 | 58,327 | 57,252 | 59,224 | 55,998 |
| 78,270 | 75,028 | 70,696 | 80,048 | 71,319 | 73,231 | 74,220 | 73,415 | 70,696 | 69,679 | 75,305 | 73,019 | 75,109 | 76,269 | 84,266 | 80,480 | 84,673 | 85,553 | 74,328 | 80,649 |
| 96,057 | 88,950 | 91,853 | 90,412 | 97,334 | 106,265 | 90,150 | 99,049 | 88,993 | 93,167 | 95,347 | 96,482 | 98,588 | 107,496 | 92,934 | 97,155 | 94,111 | 100,050 | 108,032 | 94,522 |
| 106,844 | 114,268 | 115,012 | 119,631 | 109,896 | 111,674 | 119,193 | 111,046 | 130,996 | 109,903 | 112,857 | 118,117 | 109,525 | 118,150 | 110,841 | 111,757 | 109,354 | 109,294 | 109,480 | 105,317 |
| 142,044 | 136,040 | 128,372 | 135,623 | 131,942 | 148,951 | 130,559 | 144,350 | 142,307 | 131,210 | 133,507 | 133,966 | 129,132 | 132,580 | 131,734 | 151,916 | 134,033 | 133,679 | 125,252 | 130,700 |
| 158,766 | 161,025 | 156,864 | 160,766 | 151,441 | 165,378 | 159,245 | 156,587 | 156,963 | 175,726 | 150,854 | 149,302 | 151,967 | 169,635 | 157,041 | 169,688 | 162,013 | 155,655 | 149,418 | 160,266 |
| 172,175 | 186,758 | 183,367 | 181,290 | 176,263 | 187,190 | 190,095 | 168,561 | 171,120 | 175,744 | 169,948 | 169,553 | 174,253 | 164,053 | 180,551 | 172,736 | 186,530 | 174,076 | 184,707 | 181,789 |
| 187,945 | 195,291 | 196,122 | 205,056 | 198,377 | 198,042 | 204,155 | 187,495 | 193,058 | 196,636 | 197,037 | 210,397 | 211,432 | 202,496 | 189,629 | 192,008 | 194,503 | 199,528 | 201,838 | 203,826 |

| №1 | | | №2 | | |
|--------------------|------------------|---------------------------|--------------------|------------------|---------------------------|
| Кол-во раундов, шт | Среднее время, с | Средняя скорость, кбайт/с | Кол-во раундов, шт | Среднее время, с | Средняя скорость, кбайт/с |
| 1 | 15 | 124 | 1 | 17 | 114 |
| 2 | 32 | 59 | 2 | 36 | 53 |
| 3 | 50 | 38 | 3 | 56 | 34 |
| 4 | 67 | 28 | 4 | 76 | 25 |
| 5 | 84 | 23 | 5 | 96 | 20 |
| 6 | 102 | 19 | 6 | 113 | 17 |
| 7 | 118 | 16 | 7 | 135 | 14 |
| 8 | 137 | 14 | 8 | 159 | 12 |
| 9 | 154 | 12 | 9 | 178 | 11 |
| 10 | 172 | 11 | 10 | 198 | 10 |

| Размер файла (байт) | Кол-во раундов, шт | Средняя скорость, кбайт/с |
|---------------------|--------------------|---------------------------|
| 1905152 | 1 | 119 |
| | 2 | 56 |
| | 3 | 36 |
| | 4 | 27 |
| | 5 | 21 |
| | 6 | 18 |
| | 7 | 15 |
| | 8 | 13 |
| | 9 | 12 |
| | 10 | 12 |

Параметры случайной генерации конфигурации:
 - вероятность появления блоков от 1 до 20 штук 70%;
 - вероятность появления блоков от 20 до 40 штук 25%;
 - вероятность появления блоков от 40 до 81 штук 5%;
 - блоки только из сегментов больших, чем 1 бит, до 80 штук.

Рисунок 3.3 – Подробные данные анализа скорости шифрования

Результат расчета скорости на BeagleBone Black (BeaglePocket) произведенной в Китае с 1GHz 512 MB составляет свыше 5 килобайт/с (свыше 43 килобит/с). Случайная генерация ключа в среднем выдает скорость 12 килобайт/с (около 97 килобит/с, стандартная скорость на большинстве используемого оборудования UART от 9.6 до 115.2 килобит/с). Увеличить скорость можно при помощи использования неприводимых многочленов меньшей степени.

Ниже показана скорость «Шифратора 125» для процессора ноутбука и микрокомпьютера (Рисунок 3.4). А также их примерная стоимость.

Intel i7 3537U 2 ядра
Скорость "Шифратора 125"
120 кбайт/с
8100 Р



BeagleBone Black
Скорость "Шифратора 125"
от 12 до 5 кбайт/с
4000 Р

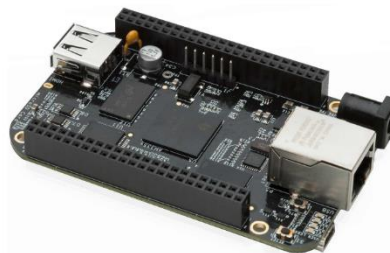


Рисунок 3.4 – Скорость шифрования на разных процессорах.

В ходе исследования, также, была установлена особенность данного алгоритма шифрования – зависимость скорости шифрования от КК. Например, неприводимый многочлен из 240 бит и 1 раунд шифрования дадут на ноутбуке скорость шифрования 120 килобайт/с, а на BeagleBoneBlack (PocketBeagle) 5 килобайт/с тогда, как средняя скорость при случайно сгенерированном КК будет, примерно, в два раза больше.

Это говорит о том, что злоумышленник может определить факт использования подблоков небольшого размера при скорости шифрования большей 120 килобайт/с или 5 килобайт/с для ноутбука и платы соответственно. В таком случае, рекомендуется создать дополнительный раунд, чтобы скорость шифрования значительно не превышала приведенные выше показатели. Подбирать необходимую скорость можно при помощи фрагментации блоков от небольших к большим.

3.1.3 Примеры возможного использования на конкретном оборудовании

Опираясь на схему электронного замка, рассмотренного в первой главе, ниже показана ее реализация на конкретном оборудовании (Рисунок 3.5).

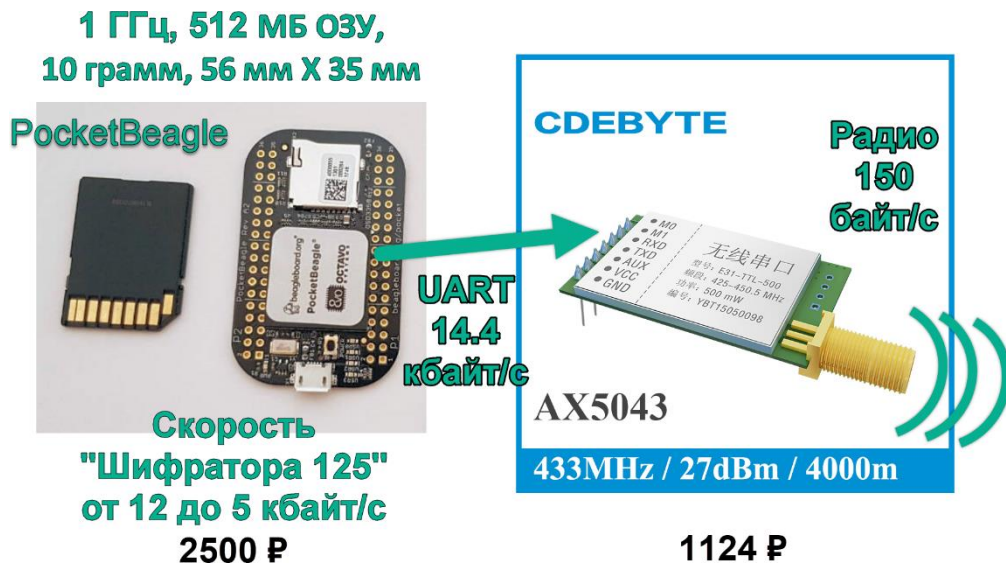


Рисунок 3.5 – Схема электронного замка на конкретном оборудовании

В системах «Умного дома» или беспилотника для решения задачи агрегирования информации с различных блоков устройств, находящихся в одном корпусе можно использовать схему ниже (Рисунок 3.6), где на вход PocketBeagle произведенной в Китае поступает информация для шифрования. Затем идет BeagleBoneBlack Industrial произведенный в Китае с улучшенным диапазоном рабочих температур и более стойкий к резким их перепадам. Он выполняет окончательное агрегирование информации и передачу данных по сети на большие расстояния (при использовании переходника, информацию можно передавать по оптоволокну).

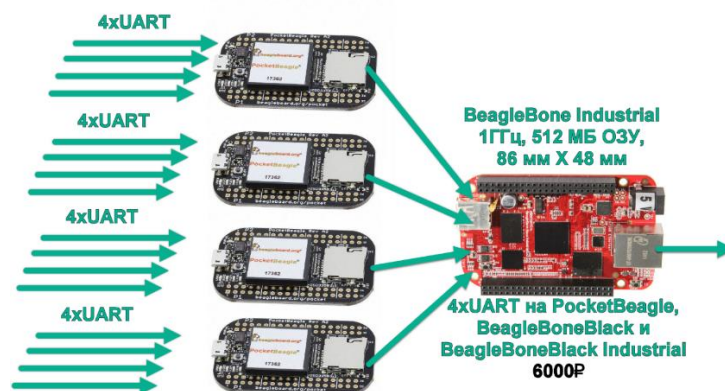


Рисунок 3.6 – Схема агрегирования информационных потоков

Кроме этого, возможно использование принципа подключения «Корпус безопасности» (Рисунок 3.7). Здесь основная идея состоит в том, чтобы для устройств, которым это необходимо, разрабатывать корпус, который помимо функций защиты (экранирования, зашумления сигналов извне и от самого устройства, дополнительной теплоизоляции, влагозащиты, стеклоомывателей и т.п.) добавить, переводя исходную информацию в унифицированный стандарт конфигурирования, шифрование только необходимой и важной информации, а остальную информацию передавать по технологии NAT. Например, для камеры видеонаблюдения – видеопоток можно передавать через NAT, а конфигурационную информацию преобразовывать из любого протокола в стандарт ONVIF и дополнительно шифровать.



Рисунок 3.7 – Концепция корпуса безопасности

3.1.4 Нагрузочное тестирование и расчет энергопотребления

Для проверки работоспособности шифратора было проведено нагрузочное тестирование, а также, осуществлен расчет энергопотребления в процессе шифрования. Ниже можно увидеть результаты тестирования энергопотребления и трехдневный тест непрерывного шифрования (Таблица 3.1). Конфигурация алгоритма шифрования: 3 раунда, один раунд с заменой по неприводимому многочлену 240 бит, остальные – случайно сгенерированы. Размер файла шифрования 13,6 гигабайт. Аккумулятор 20000 mAh, Li-ion. Во время шифрования процессор был загружен вычислениями

на 100%. Температура окружающей среды 25°C. Корректность шифрования проверялась по объему зашифрованных данных и количеству прошедшего времени с учетом известной скорости шифрования алгоритма и только в случае совпадения результатов проверка непрерывного шифрования признавалась успешной.

Таблица 3.1 – Сравнение возможностей шифраторов

| Вид теста | Время | Результаты |
|--|-----------------------|---|
| Разряд аккумулятора и непрерывное шифрование | 19.05.2018 [16:29] | Успешный запуск устройства и шифратора |
| Разряд аккумулятора и непрерывное шифрование | 19.05.2018 [20:17] | Проверка. Непрерывное шифрование 4 часа. Потребление 2000 mAh (500 mAh/ч) |
| Разряд аккумулятора и непрерывное шифрование | 20.05.2018 [12:07] | Проверка. Непрерывное шифрование 20 часов. Потребление 11660 mAh (728 mAh/ч) |
| Непрерывное шифрование | 20.05.2018 [22:00] | Проверка. Непрерывное шифрование 1 день 6 часов |
| Непрерывное шифрование | 21.05.2018 [20:23] | Проверка. Непрерывное шифрование 2 дня 2 часа |
| Непрерывное шифрование | 22.05.2018 [02:00] | Проверка. Непрерывное шифрование 2 дня 8 часа |
| Непрерывное шифрование | 22.05.2018 [16:30] | Проверка. Непрерывное шифрование 3 дня |

Плата, на которой проводился непрерывный тест шифрования эксплуатируется более трех лет без сбоев. Один год из трех лет плата работала практически в круглосуточном режиме осуществляя периодические операции чтения/записи без серьезных проблем. Также, ранее, был проведен месячный тест непрерывной работы платы с шифрованием файлов небольшой длины два раза в день в случайное время.

Все тесты «Шифратора 125» были проведены на оборудовании Beagle-BoneBlack (есть компактный аналог BeaglePocket) произведенной в Китае с 1GHz процессором и 512MB ОЗУ.

3.2 Тесты ОК и шифртекста «Шифратора 125»

3.2.1 Первый критерий качества. Посимвольная проверка.

Авторы книги [25] предлагают оценочный метод посимвольной проверки, который считают самым сильным из всех рассмотренных в своей работе (подборка тестов Д. Кнута, система оценки статистических свойств «DIEHARD», CRYPT-S, руководство НИСТ и другие [26-28]).

Данный метод заключается в проверке равномерности распределения символов в исследуемой последовательности на основе анализа частот появления каждого символа.

Пусть $\varepsilon = \varepsilon_1, \varepsilon_2, \dots, \varepsilon_n$ – последовательность блоков m -разрядных двоичных чисел длины n . Для выполнения критерия, все значения v_i (количество повторений) должны лежать в интервале по формуле (3.1).

$$\left[\frac{n-2.58\sqrt{n(2^m-1)}}{2^m}, \frac{n+2.58\sqrt{n(2^m-1)}}{2^m} \right] \quad (3.1)$$

Например, для последовательности $\varepsilon = 3 \ 5 \ 4 \ 2 \ 1 \ 4 \ 6 \ 1$, $n=8$, $m=3$, все значения повторений $v_0=0$, $v_1=2$, $v_2=1$, $v_3=1$, $v_4=2$, $v_5=1$, $v_6=1$, $v_7=0$ принадлежат интервалу (3.2).

$$\left[\frac{8-2.58\sqrt{8(2^3-1)}}{2^3} = -1.41; \frac{8+2.58\sqrt{8(2^3-1)}}{2^3} = 3.41 \right] \quad (3.2)$$

В первом критерии тест используется для анализа бинарных последовательностей по одному биту в отдельности, поэтому итоговая формула расчета диапазона значений примет более простой вид (3.3).

$$\left[\frac{n-2.58\sqrt{n}}{2}, \frac{n+2.58\sqrt{n}}{2} \right] \quad (3.3)$$

Тест только по каждому отдельному биту уязвим для комбинаций-чередований нулей и единиц. Например, 11110000, 10101010 и т.п. Поэтому в программе он используется одновременно с графическим способом распределения на плоскости, который адаптивно анализирует координаты с учетом их расположения относительно друг друга.

3.2.2 Второй критерий качества. Распределение на плоскости.

Авторы книги [25] предлагают графический метод распределения на плоскости. Данный тест осуществляется следующим образом. На поле размером $(2^R-1)(2^R-1)$, где R – разрядность чисел исследуемой последовательности, наносятся точки с координатами $(\varepsilon_i; \varepsilon_{i+1})$, ε_i – элементы исследуемой последовательности ε , $i=1, \dots, (n-1)$. n – длина последовательности. Например, для последовательности $\varepsilon=2\ 3\ 5\ 4\ 3$ получим точки $(2;3)$, $(3;5)$, $(5;4)$, $(4;3)$.

Далее, авторами книги [25] предлагается проводить визуальную оценку полученных результатов. Если между элементами последовательности отсутствуют зависимости, то точки на поле расположены хаотично (Рисунок 3.8, слева). Если на координатной плоскости присутствуют зависимости – последовательность не является случайной (Рисунок 3.8, по центру). Для последовательностей большой длины хорошим результатом является черный квадрат (Рисунок 3.8, справа).

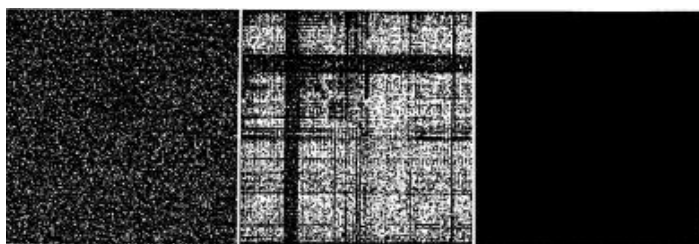


Рисунок 3.8 – Распределение координат из последовательности на плоскости

Предложенный графический метод был улучшен и преобразован в оценочный способ для выполнения оценки бинарной последовательности при помощи нормировки системы координат по длине тестируемого сообщения. В зависимости от длины сообщения, выбирается интервал изменения координат XU общий для координаты X

и для координаты Y . Таким образом, чтобы множество координат (X, Y) , которое задает бинарная последовательность слева-направо со сдвигом на 1 координату, вмещало все получаемые таким образом координаты не менее одного раза. Для этого XU подбирается таким образом, чтобы выполнялось условие (3.4).

$$(2^{XY-1})^2 < \frac{n}{XY} \leq (2^{XY})^2 \quad (3.4)$$

Где n – количество бит в сообщении. В идеале, должно выполняться равенство (3.5).

$$\frac{n}{XY} = (2^{XY})^2 \quad (3.5)$$

Оно говорит о том, что плоскость с интервалом изменения координат XU может вместить в себя каждую координату ровно один раз, другими словами, получится полностью черный квадрат или мера хаотичности будет равна 100%. Если же, выполняется условие (3.6).

$$\frac{n}{XY} < (2^{XY})^2 \quad (3.6)$$

Тогда лишние координаты (которые всегда будут свободны) определяются по формуле (3.7).

$$Q = (2^{XY})^2 - \frac{n}{XY} \quad (3.7)$$

Мера хаотичности будет равна значению (3.8), где b – количество черных (заполненных) пикселей, а p – количество всех пикселей изображения.

$$W = \frac{b}{p} \cdot 100 \quad (3.8)$$

Например, для последовательности $\varepsilon=0\ 1\ 1\ 0$ получим точки $(0;1)$, $(1;1)$, $(1;0)$, $(0;0)$, $XY=1$, $Q=0$, $W=100\%$ (полностью черный квадрат). В результате экспериментов определено, что для получения достоверных результатов длина последовательности должна быть не менее 100 бит, а лучше не менее 10000 бит.

Лишние координаты Q должны быть одинаковыми для разных тестов, чтобы их можно было сравнивать между собой и, лучше всего, минимальными, т.е. равными нулю. Вероятность попадания координат более длиной бинарной последовательности в уникальную позицию меньше, чем вероятность попадания координат меньшей по длине бинарной последовательности на графике той же размерности, поэтому после вычитания Q , во втором случае, значения получаются завышенными от расчетных. Во избежание этого нужно проводит эксперименты при одинаковых значениях Q , а лучше, при $Q=0$. Для программы шифрования, из-за нормировки, минимальное значение параметра Q равно 16. Достигается оно при длине сообщения: 960, 5040, 24480 и т.п.

Для того, чтобы уточнить теоретические границы на практике, были проведены дополнительные эксперименты (Таблица 3.2).

Таблица 3.2 – Расчет практических границ второго критерия

| № | Кол-во бит в ОК или ШТ ($Q = 16$) | | | № | Максимальная практическая граница, | | |
|---|-------------------------------------|-------------|-------------|--|------------------------------------|--------------|--------------|
| | 960 | 5040 | 24480 | | 9 | 63,90 | 62,70 |
| - | | | | 10 | 63,46 | 63,19 | 65,00 |
| Минимальная практическая граница, % | | | | | | | |
| 1 | 2,50 | 0,50 | 0,12 | 11 | 64,14 | 65,97 | 63,75 |
| 2 | 2,08 | 0,40 | 0,07 | 12 | 63,80 | 62,90 | 71,25 |
| 3 | 1,67 | 0,30 | 0,10 | 13 | 62,16 | 62,50 | 64,17 |
| Средняя мин. практическая граница, % | | | | 14 | 63,72 | 63,00 | 64,58 |
| - | 2,08 | 0,40 | 0,10 | 15 | 63,68 | 62,90 | 67,50 |
| Максимальная практическая граница, % | | | | 16 | 63,21 | 62,20 | 69,58 |
| 1 | 63,45 | 62,80 | 63,33 | 17 | 62,92 | 63,69 | 65,00 |
| 2 | 62,87 | 63,89 | 61,70 | 18 | 63,50 | 64,09 | 63,75 |
| 3 | 64,26 | 63,99 | 67,08 | 19 | 64,51 | 62,30 | 68,75 |
| 4 | 63,16 | 65,18 | 62,08 | 20 | 63,95 | 63,39 | 64,17 |
| 5 | 63,11 | 64,88 | 65,42 | Мин. максимальная практическая граница, | | | |
| 6 | 63,50 | 62,70 | 64,58 | - | 62,16 | 62,20 | 61,70 |
| 7 | 63,43 | 63,99 | 67,08 | Средняя максимальная практическая | | | |
| 8 | 63,43 | 63,49 | 65,00 | - | 63,54 | 63,49 | 65,33 |

По достижению максимальной практической границы в диапазоне от 62,15% и больше, можно утверждать, что бинарная последовательность успешно выполнила второй критерий, но чем больше данный показатель, тем меньше повторений координат на плоскости. Данные границы были получены при подаче на вход второму критерию последовательностей, о которых было заранее известно, что они являются псевдослучайными последовательностями.

Может показаться, что после использования второго теста, в первом нет необходимости, но это не так. Учитывая, что чаще всего только 65% от последовательности занимает уникальную координату, то не все множество координат представлено в ней. Иначе говоря, при показаниях второго теста ниже 100%, всегда будет происходить случайное смещение количество нулей и единиц бинарной последовательности влево или вправо от равного их значения и, чем дальше от 100%, тем данные флуктуации будут сильнее. Поэтому, необходимо компенсировать этот недостаток первым критерием.

3.2.3 Обоснование необходимости внедрение третьего критерия.

Благодаря первым двум тестам можно утверждать, что около 65% бинарной последовательности занимает уникальную координату на плоскости. Но что можно сказать об оставшихся 35%? Это множество координат, которые повторяются и значительно уменьшить этот процент практически невозможно.

Как быть в такой ситуации? Необходимо принять факт наличия повторений координат. В принципе, это не является критичным, но только, если эти повторения распределены по всему множеству координат случайным образом.

Например, проведем мысленный эксперимент, если сгенерировать последовательность, которая успешно выполнит первые два критерия и покажет результат в 65% хаотичности, то, заменив все повторяющиеся координаты на значение, допустим, 00001111 для восьмибитных координат, то результаты теста не изменятся, а закономерность последовательности недопустимо увеличится.

Это произойдет, потому что, оба теста не определяют каким образом происходит повторение оставшихся 35% координат: они расположены закономерно относительно друг друга или равномерно распределены по всему множеству?

Возможно ли решить данную проблему? Да, но для этого необходимо вернуться к общей формуле первого критерия частотного анализа (3.1). Теперь ее не нужно упрощать, вместо этого, приравнять интервал изменения координаты XU из второго теста к значению разрядности числа частотного анализа m по формуле (3.1). Тогда получим формулу (3.9).

$$\left[\frac{k - 2.58\sqrt{k(2^{XY} - 1)}}{2^{XY}}, \frac{k + 2.58\sqrt{k(2^{XY} - 1)}}{2^{XY}} \right] \quad (3.9)$$

Где размер текста тоже необходимо изменить, уменьшив его, в соответствии с разрядностью координаты, по формуле (3.10).

$$k = \frac{n}{XY} \quad (3.10)$$

Благодаря этому тесту можно убедиться в том, что 35% повторенных координат распределились по множеству случайно, а не закономерно. Также, в отличии от первого критерия, необходимо обеспечить возможность более точной интерпретации результатов третьего критерия с целью определения степени наличия закономерностей.

Достаточно будет использовать два оценочных показателя для третьего критерия: количество координат, по которым зафиксирован выход за пределы допустимого интервала и максимальное значение выхода за допустимый интервал.

3.2.4 Выявленные закономерности

По результатам оценки качества шифртекста при различных КК (Рисунок 3.9), можно сделать следующие выводы:

| Настройки шифра | Критерий 1 | Критерий 2 | Настройки шифра | Критерий 1 | Критерий 2 |
|---|------------|--|--|------------|--|
| Исходный текст | ✗ |  4.93% | - с ключом; - без замены; - 1 раунд. | ✗ |  18.80% |
| - без ключа; - малые блоки; - 1 раунд. | ✗ |  27.11% | - без ключа; - средние блоки; - 1 раунд. | ✗ |  42.62% |
| - без ключа; - большие блоки; - 1 раунд. | ✓ |  63.11% | - с ключом; - случайная конфигурация; - 1 раунд. | ✓ |  63.68% |
| - с ключом; - случайная конфигурация; - 3 раунда. | ✓ |  63.33% | - с ключом; - случайная конфигурация; - 5 раундов. | ✓ |  63.75% |

Рисунок 3.9 – Оценка качества шифртекста при разных конфигурациях

- если произвести замену хотя бы одним блоком замены по неприводимому многочлену от 120 бит и более, то это приведет к увеличению меры хаотичности и случайности шифртекста с наименьшим количеством раундов. Например, можно использовать подобные блоки замен в первых раундах. Чем больше степень неприводимого многочлена для замены, тем лучше (максимально – 240 бит);
- ключ вносит около 10% хаотичности, малые блоки 20%, средние 40%, а крупные 60%;
- случайная конфигурация очень часто приводит к необходимому результату при любом количестве раундов. Большое количество раундов может быть использовано для увеличения вариативности КК (при использовании временных ключей вместо ОК) и хаотизации последовательностей со значительными закономерностями, а меньшее для повышения скорости.


3.2.5 Расчет строго лавинного критерия и криптоанализ

Для оценки возможности проведения линейного криптоанализа был рассчитан строгий лавинный критерий (СЛК) [37] для «Шифратора 125», так как он является показателем хорошей конфузии и диффузии [38]. Исходное сообщение одинаково для всех ключей «Текст для тестирования СЛК шт» в таблице UNICODE. Проводились

240 замеров для каждого ключа. В каждом замере изменялся один бит блока, а полученный шифртекст сравнивался с обычным шифртекстом сообщения. Для исследования использовался второй блок с данными, первый нормировочный блок не задействуется.

В итоге, были получены результаты, из которой видно, что «Шифратор 125» успешно выполнил СЛК для диффузии и конфузии (Таблица 3.3), потому что эти значения практически полностью совпадают с поведением псевдослучайной последовательности (ПСП). ПСП использовался вместо текста, каждый замер использовалась новая ПСП и вычиталась из первой ПСП, затем все характеристики рассчитывались, как и при расчете СЛК шифра. Конфузия определялся для ключа раунда, но учитывая конфигурационный ключ, ключ раунда может использоваться только, как имитозащита в протоколе на временных ключах, либо необходимо добавлять второй раунд, если требуются хорошие показатели конфузии ключа раунда.

Таблица 3.3 – Результаты расчета строго лавинного критерия

| | Максимальное отклонение от ½ среди всех битов блока | Кол-во бит выполняющих строгий лавинный критерий с точностью 0.1 | Кол-во бит выполняющих строгий лавинный критерий с точностью 0.01 | Лавинный эффект | Частотный анализ |
|--------------------------------------|---|--|---|-----------------|---|
| Эталон, для расчета использованы ПСП | 0.10 | 239 | 53 | 0.50 | |
| Диффузия, 1 раунд, небольшие блоки | 0.50 | 0 | 0 | 0.02 |  K1:- K2: 26% |
| Диффузия, 1 раунд, большой блок | 0.09 | 240 | 33 | 0.50 |  K1:+ K2: 63% |
| Диффузия, 3 раунда | 0.09 | 240 | 51 | 0.50 |  K1:+ K2: 64% |
| Конфузия, 1 раунд | 0.50 | 0 | 0 | 0.49 | |
| Конфузия, 2 раунда | 0.09 | 240 | 44 | 0.50 | |

Рассмотрим возможности применения других известных атак на шифраторы [39-46].

Атака полным перебором: наибольшее количество свободных входных параметров «Шифратора 125» среди других широко распространенных решений (ранее

была представлена таблица со сравнительными характеристиками) делает его защищенным от данной лучше, чем другие решения.

Статистический криптоанализ: режим сцепки блоков с вектором инициализации из ПСП, успешное выполнение статистических тестов [47] (например, частотного анализа), результаты были представлены ранее.

Атака на основе связанных ключей: ключи всегда являются случайными последовательностями (принимаются по результатам выполнения статистических тестов, например, частотного анализа).

Дифференциальный и интегральный криптоанализ: не имеет смысла, потому что S-блоки переменной длины, они могут быть большими и шифр не в режиме простой замены (кодовой электронной книги), так как использована сцепка блоков, все это сводит на нет возможность взлома шифра по каждому раунду в отдельности, зная структуру небольших блоков замен.

3.3 Конфигурирование протокола

3.3.1 Классификация каналов безопасного обмена информацией

Для того, чтобы выделить оптимальные параметры протокола, необходимо понять какие ситуации могут возникнуть при передаче информации по безопасному каналу связи. Данную классификацию можно произвести относительно двух характеристик: количество ключей или тип передаваемой информации.

Если рассматривать первую характеристику количества ключей, то можно определить два способа работы протокола: один ключ, много пользователей и много пользователей, для каждого свой ключ. Такое возникает, когда пользователи разделяют общую среду передачи или общую сеть, например, в случае WiFi сети или VPN режима.

Как было сказано ранее, относительно VPN, два этих способа работы в протоколе будут осуществлены одновременно. Другими словами, начальный вариант работы протокола рассчитан для ситуации много пользователей, для каждого свой ключ.

Но для обеспечения возможности использования режима VPN и WiFi, в протоколе будет дополнительно задействован режим, в котором добавляется один дополнительный ключ одинаковый в рамках конкретной сети VPN или WiFi. Этот ключ будет шифровать дополнительные служебные поля конкретной общей сети, а служебные поля пользователей и их данные будут зашифрованы ключом пользователя. При этом, не будет двойного шифрования данных ключом сети и ключом пользователя, что значительно экономит время, в отличие от существующих распространенных решений VPN. Идентификатор пользователя в служебных полях сети превратится в идентификатор конкретной сети.

Но для определения параметров оптимальной конфигурации протокола, больший интерес представляет следующая классификация, по типу передаваемых данных:

- передача команд изменения конфигурации, критичных обновлений;
- передача управляющих команд и телеметрии;
- передача большого потока данных в реальном времени, например, видео и звука, обновлений системы.

3.3.2 Определение параметров протокола и его правильная конфигурация

Ранее уже были определены ключевые параметры повышения безопасности передачи данных для «Протокола 125»: длина блока или ключа шифрования, количество раундов шифрования, количество временных ключей на количество переданных зашифрованных данных. Данные параметры подбираются таким образом, чтобы для передаваемых данных получались шифртексты с высокими показателями.

Рассмотрим первый параметр протокола. Длина блока – этот параметр один из немногих фиксированных параметров. Учитывая наличие режима сцепки шифра и посылку временных ключей на определенный объем информации, аутентификацию и хэш, конфигурационного ключа в протоколе его менять нет необходимости. Блок из 240 бит для таблицы UNICODE позволяет зашифровать 15 символов, иначе говоря, он работает «на уровне» слов и небольших словосочетаний. Повторения в словах достаточно редки, тем более, словосочетаний, учитывая непредсказуемость смещения

их начала, в отличие от символов. А благодаря сцепке по предыдущему шифртексту, проблему повторений можно исключить практически полностью.

Сцепка обеспечивает защиту от повторений в рамках одного временного ключа, смена временного ключа меняет все последовательности сцепок на новые, аутентификация и хэш каждого сообщения не дает возможности использовать какие-то его части, конфигурационный ключ делает неизвестным все входные параметры «Шифратора 125».

Количество раундов шифрования может изменяться для обеспечения большего множества уникальных конфигураций ключа, выполнения тестов на больших объемах данных.

Максимальная длина общего ключа определяется количеством раундов, а минимальная дополненным до кратности 240 бит введенным общим ключом. Дополнение происходит копированием необходимого количества бит из начала общего ключа.

Параметр объема данных в сообщении для одного временного общего ключа создает незначительную избыточность, но и повышает безопасность шифра и делает уникальным каждую пересылку данных в сообщении, даже, если они повторяются.

Следующие параметры связаны непосредственно с работой самого протокола: задержка повторной посылки пакета, диапазон случайной добавки к задержке повторной посылки пакета, количество попыток до потери сессии, количество попыток до потери соединения, количество шагов таймера до повторной посылки параметров режима без уведомлений, режим передачи с уведомлением или без, количество попыток до признания ошибки аутентификации, количество попыток до повторного создания пакета из исходных данных.

Количество раундов выбирается для повышения надежности в случае избыточной пропускной способности канала и отсутствии требований по энергоэффективности, либо, когда шифртекст не проходит результаты тестирования.

В случае передачи команд изменения конфигурации, необходимо одну команду слать в отдельном сообщении, так как для каждого сообщения есть новый временный ключ. Это обеспечивает наибольшую защиту целостности и конфиденциальности.

Для дополнительной защиты от коллизий рекомендуется слать минимум 3-5 сообщений с одинаковой командой подряд. Шифртекст должен давать лучшие показатели. Раунды можно увеличивать, так как не требуется большая скорость обмена информацией.

В случае передачи команд управления или данных телеметрии, скорость играет большую роль, но, все-равно, можно, в случае необходимости, увеличить количество раундов. Так как данные команд управления не влияют на конфигурацию и постоянно обновляются, то рекомендуется не защищать данные сообщения от возможности возникновения коллизий. Только, если параметр долго не обновляется и за это время оператор или программа может принять неправильное решение без возможности последующей его корректировки. В таком случае, можно слать несколько сообщений, чтобы максимально снизить вероятность появления коллизий. Параметры телеметрии и управляющие команды можно отправлять в одном сообщении в соответствии с установленным параметром максимальной нагрузки данных на один временный ключ.

В случае отправки потоковых данных большого объема, например, видео или аудио, рекомендуется использовать только один раунд шифрования и режим без уведомлений. В «Приложении Б» можно ознакомиться с генератором конфигурационных ключей необходимых для работы «Протокола 125».

Удалось определить параметры протокола: количество раундов, режим сцепки шифра, объем данных в сообщении для одного временного общего ключа, задержка повторной отправки пакета, диапазон случайной добавки к задержке повторной отправки пакета, количество попыток до потери сессии, количество попыток до потери соединения, количество шагов таймера до повторной отправки параметров режима без уведомлений.

В том числе, даны рекомендации по правильному выбору и конфигурированию параметров протокола при различных типах передаваемой в канале информации, в том числе, рекомендации по дополнительным мерам защиты и контроля надежности шифртекста. Также, существует перспективная модель обмена ключами по квантовому каналу связи, подробнее можно ознакомиться в «Приложении В».

3.4 Выводы

Протестирована кроссплатформенная реализация «Шифратора 125». Тестирование проводилось, как на качество шифрования, так и на качество работы в конкретных аппаратных платформах. Получены приемлемые результаты тестирования.

В том числе, сформулирована методика конфигурирования протокола в зависимости от вида канала защищенного обмена информацией, предложена модель обмена ключами по квантовому каналу связи и инструкция по работе с генератором ключей шифратора.

ЗАКЛЮЧЕНИЕ

По результатам поделанной работы была определена целевая область использования протокола, описаны основные для его применения автоматизированные системы, организованные по принципу «ведущее-зависимое» устройство, на основании которых разработаны этапы внедрения и апробации протокола, сформулированы 8 критериев, которым должен соответствовать протокол.

Проведен анализ безопасности широкораспространенных протоколов и их шифраторов, составлены 2 подробные сравнительные таблицы по протоколам и шифраторам для определения степени их соответствия сформулированным требованиям, обоснована необходимость разработки нового протокола и шифратора, для более быстрой альтернативы можно использовать OpenVPN, предварительно его доработав.

Дано описание окончательному варианту протокола и шифратора, в алгоритме шифрования удалось расширить количество свободных входных параметров, особенно, при помощи подблоков не постоянной, а переменной длины, добавить нормировку, сцепку блоков с вектором инициализации.

Реализован шифратор и генератор ключей для протокола с возможностью запуска на Windows, Linux и BeagleBoneBlack (PocketBeagle) производство Китай, исследована скорость шифратора на ноутбуке и BeagleBoneBlack (PocketBeagle), она составляет около 120 килобайт/с и 5 килобайт/с, приведены примеры использования на конкретном оборудовании.

Энергопотребление платы около 500 mAh/ч, успешно пройдены нагрузочные тесты на плате BeagleBoneBlack (PocketBeagle) произведенной в Китае: 3 дня непрерывного шифрования, 1 месяц непрерывной работы с периодическим шифрованием, 1 год непрерывной работы и 3 года эксплуатации.

В генератор ключей включено 2 теста (один из них был доработан) и предложено внедрение 3 теста, тесты относительно других вариантов имеют лучшие показатели определения случайности последовательности и уменьшают недостатки друг друга, на основании тестирования шифратора по внедренным и отдельным тестам,

выявлены закономерности и сформулированы рекомендации интерпретации их результатов. Шифр устойчив к криптоанализу.

Определены оптимальные параметры протокола и разработана методика их правильной конфигурации в соответствии с различными существующими типами каналов обмена защищаемой информацией. Приведена схема инициализации сессии протокола и инструкция по выбору параметров шифратора и генератора ключей.

Показатели алгоритма могут быть улучшены при помощи многоядерный процессоров, увеличения производительности вычислительных устройств, качества сред передачи данных, увеличения блока с исходными данными, а также, при помощи использования устройств большей вычислительной размерности (например, троичные компьютеры) [36].

В будущем необходимо внедрить в генератор ключей предложенный третий критерий тестирования ОК и шифртекста, а также, алгоритм подбора конфигурации для ограничения скорости работы шифра при помощи дополнительного раунда с необходимой фрагментация от маленькой до крупной, автоматический тест на строго лавинный критерий диффузии и конфузии.

Реализовать возможность получения неприводимых многочленов не из заранее известных таблиц, а случайным образом, осуществляя поиск неприводимого многочлена отталкиваясь от случайной бинарной последовательности, тестируя ее алгоритмом Берлекемпа и, в случае неудачи, инкрементируя значение случайной последовательности, продолжать поиск до достижения успеха. Генерацию ключа можно проводить один раз, до начала использования алгоритма.

Также, создать кроссплатформенную библиотеку шифратора, разработать программную реализацию «Протокола 125», осуществить ранее рассмотренные этапы внедрения и протестировать протокол на автоматизированных системах «ведущее-зависимое» устройство, рассмотренных в них.

Рассмотреть возможность внедрения в алгоритм дополнительного сложения с ключом перед фрагментацией блоков для замены с целью достижения высокой конфузии за 1 раунд. Для этого ключ раунда должен быть в 2 раза больше (480 бит).

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Митращук, В.В. Разработка беспилотного летательного аппарата по типу квадрокоптера с обеспечением передвижения параллельно земле платформы со сменными модулями приборов / В.В. Митращук // Актуальные проблемы авиации и космонавтики в 2 т. – Красноярск: Сиб. гос. аэрокосмич. ун-т., 2016. – Т. 1. – С. 299-301.
2. Голубчиков, Д.М. Квантовая криптография: принципы, протоколы, системы / Д.М. Голубчиков, К.Е. Румянцев // Таганрогский технологический институт Южного федерального университета. – Таганрог. – 37 с.
3. Пригожин, И. Квантовые компьютеры и квантовые вычисления / И. Пригожин, В.А. Садовничий // Международный научный журнал. – Москва, 2000. – №1. – 116 с.
4. Китаев, А. Классические и квантовые вычисления / А. Китаев, А. Шень, М. Вялый. – Москва: МЦНМО, 1999. – 193 с.
5. Белокуров, В.В. Квантовая телепортация – обыкновенное чудо / В.В. Белокуров, О.Д. Тимофеевская, О.А. Хрусталева. – Ижевск: РХД, 2000. – 256 с.
6. Манин, Ю.И. Вычислимое и невычислимое / Ю.И. Манин. – Москва: Сов. Радио, 1980. – 128 с.
7. Нейман, И. Математические основы квантовой механики / И. Нейман. – Москва: Издательство «Наука», 1964. – 366 с.
8. Садовничий, В.А. Квантовые вычисления: за и против / В.А. Садовничий. – Ижевск: Издательский дом «Удмуртский университет», 1999. – 212 с.
9. Садовничий, В.А. Квантовый компьютер и квантовые вычисления / В.А. Садовничий. – Ижевск: Ижевская республиканская типография, 1999. – 288 с.
10. Divide and Conquer: Cracking MS-CHAPv2 with a 100% success rate [Электронный ресурс] // URL: <https://www.cloudcracker.com/blog/2012/07/29/cracking-ms-char-v2/> (дата обращения: 20.10.2017).
11. Dunkelman, O. A Practical-Time Attack on the A5/3 Cryptosystem Used in Third Generation GSM Telephony / O. Dunkelman, N. Keller, A. Shamir // Faculty of Mathematics and Computer Science. – Israel: Weizmann Institute of Science. – 23 с.

12. Kraken: к взлому сетей GSM готов! [Электронный ресурс] // URL: <https://haker.ru/2010/07/22/52768/> (дата обращения: 15.10.2017).
13. Шнитман, В.З. Протокол обмена ключами версии 2 (IKEv2) / В.З. Шнитман // Российская Академия Наук. – Москва: Институт системного программирования, 2007. – 78 с.
14. An Illustrated Guide to IPsec [Электронный ресурс] // URL: <http://unixwiz.net/techtips/iguide-ipsec.html#ip> (дата обращения: 23.10.2017).
15. Методическое пособие IPSec [Электронный ресурс] // URL: <http://dfe.petrus.ru/koi/posob/security/index.html> (дата обращения: 23.10.2017).
16. Препарируем OpenVPN. Часть 1. Статические ключи [Электронный ресурс] // URL: <https://habrahabr.ru/post/313052/> (дата обращения: 19.11.2017).
17. HTTP Headers [Электронный ресурс] // URL: <https://docs.trafficserver.apache.org/en/7.1.x/developer-guide/plugins/http-headers/> (дата обращения: 15.11.2017).
18. Опубликована подробная информация о проблемах WPA2 [Электронный ресурс] // URL: <https://haker.ru/2017/10/16/wpa2-krack-2/> (дата обращения: 15.11.2017).
19. Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2 [Электронный ресурс] // URL: <https://papers.mathyvanhoef.com/ccs2017.pdf> (дата обращения: 15.11.2017).
20. Теоретические основы защиты WPA\WPA2 [Электронный ресурс] // URL: http://mgupi-it.ru/Tech/wireless/wifisecurity_part2.html (дата обращения: 13.12.2017).
21. SecureShell 2 [Электронный ресурс] // URL: http://unix1.jinr.ru/faq_guide/sec/ssh2/ (дата обращения: 13.12.2017).
22. Противодействие атакам на протокол TLS [Электронный ресурс] // URL: <http://www.cryptopro.ru/sites/default/files/docs/TLSvsBEAST.pdf/> (дата обращения: 13.12.2017).
23. Легкий способ дешифрования закрытой XML-информации [Электронный ресурс] // URL: <https://haker.ru/2013/10/01/light-xml-decrypted/> (дата обращения: 10.12.2017).

24. Padding Oracle Attack: криптография по-прежнему пугает [Электронный ресурс] // URL: <https://habrahabr.ru/post/338072/> (дата обращения: 19.09.2017).
25. Иванов, М.А. Теория, применение и оценка качества генераторов псевдослучайных последовательностей / М.А. Иванов, И.В. Чугунков. – Москва: КУДИЦ-ОБРАЗ, 2003. – 240 с.
26. Кнут, Д. Искусство программирования для ЭВМ в 3 томах / Д. Кнут. – Москва: Мир, 1998. – Т.2.
27. DIEHARD Statistical Tests [Электронный ресурс] // URL: <http://stat.fsu.edu/> (дата обращения: 27.09.2017).
28. A Statistical Test Suite for the Validation of Random and Pseudorandom Number Generators. NIST Special Publication 800-22 [Электронный ресурс] // URL: <http://stat.fsu.edu/> (дата обращения: 27.09.2017).
29. Жданов, О.Н. Алгоритм шифрования с переменной фрагментацией блока / О.Н. Жданов, А.В. Соколов // Проблемы и достижения в науке и технике. – Омск: Инновационный Центр Развития Образования и Науки, 2015. – № 2. – С. 153-159.
30. Митрашук, В.В. Протокол безопасного обмена данными на основе алгоритма шифрования с переменной фрагментацией блока // Молодежь. Общество. Современная наука, техника и инновации. – Красноярск: Сиб. гос. аэрокосмич. ун-т., 2017. – С. 299-301.
31. Шифратор 125 [Электронный ресурс] // URL: <https://github.com/malfis/Shifr> (дата обращения: 23.05.2018).
32. Лидл, Р. Конечные поля в 2-х томах, пер. с англ. / Р. Лидл, Г. Ниддеррайтер. – Москва: Мир, 1988. – Т.1. – 430 с.
33. Seroussi, G. Table of Low-Weight Binary Irreducible Polynomials / Seroussi, G. // Computer Systems Laboratory HPL-98-135. – HEWLETT PACKARD, 1998. – 16 с.
34. Gustafson, H. A computer package for measuring strength of encryption algorithms / H. Gustafson // Journal of Computers and Security, volume 13. – 1994. – № 8. – С. 687-697.
35. Menezes, A. Handbook of Applied Cryptography / A. Menezes, P. Oorshot, S. Vanstone. – CRC Press, 1997.

36. Zhdanov, O.N. Block symmetric cryptographic algorithm based on principles of variable block length and many-values logic / O.N. Zhdanov, A.V. Sokolov // Far East Journal of Electronics and Communications Volume 16. – India: Pushpa Publishing House, 2016. – № 3. – 573-589 с.
37. Лавинный эффект [Электронный ресурс] // URL: http://wp.wiki-wiki.ru/wp/index.php/Лавинный_эффект (дата обращения: 11.05.2018).
38. Shannon C.E. Vol 28 // Communication Theory of Secrecy Systems. – 1949. – С. 656-715.
39. Блочные шифры и их криптоанализ [Электронный ресурс] // URL: http://cryptowiki.net/index.php?title=Блочные_шифры_и_их_криптоанализ/ (дата обращения: 15.05.2018).
40. A Tutorial on Linear and Differential Cryptanalysis [Электронный ресурс] // URL: http://www.engr.mun.ca/~howard/PAPERS/ldc_tutorial.pdf (дата обращения: 20.05.2018).
41. Пилиди, В.С. Криптография. Вводные главы / В.С. Пилиди. – Ростов-на-Дону: ЮФУ, 2009. – 110 с.
42. Баричев, С.Г. Основы современной криптографии / С.Г. Баричев, В.В. Гончаров, Р.Е. Серов. – Москва: ДИАЛОГ-МИФИ, 2011. – 175 с.
43. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си / Б. Шнайер. – Москва: Триумф, 2002.
44. Алферов, А.П. Основы криптографии / А.П. Алферов, А.Ю. Зубов, А.С. Кузьмин, А.В. Черемушкин. – Москва: Гелиос АРВ, 2002. – 480 с.
45. Габидулин, Э. М. Защита информации: учебное пособие / Э.М. Габидулин, А.С. Кшевецкий, А.И. Колыбельников. – Москва: МФТИ, 2011. – 262 с.
46. Фомичев, В.М. Методы дискретной математики в криптологии / В.М. Фомичев. – Москва: Диалог-МИФИ, 2010. – 424 с.
47. Захарова, К.О. Исследование статистических параметров алгоритма шифрования с переменной фрагментацией блока / К.О. Захарова // Материалы XXI Меж-

дунар. науч.-практ. конф., посвящ. памяти генерального конструктора ракетно-космических систем академика М. Ф. Решетнева в 2 ч. – Красноярск: СибГУ им. М. Ф. Решетнева, 2017. – Ч. 2. – С. 400-401.

ПРИЛОЖЕНИЯ

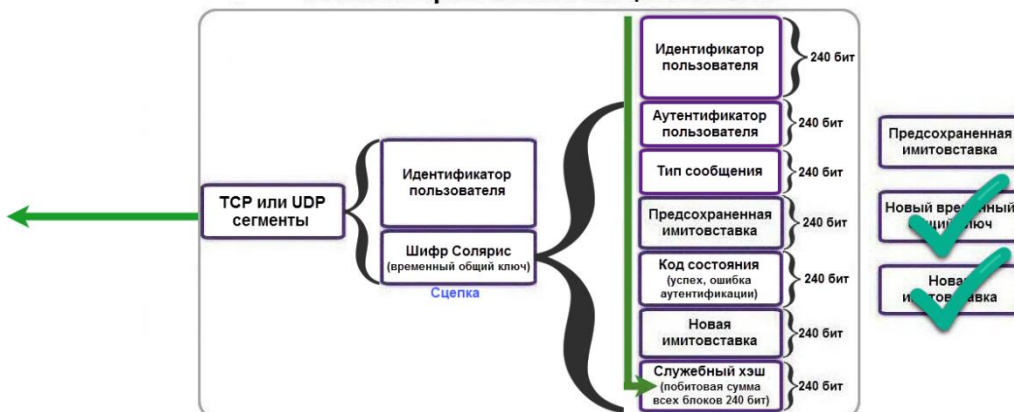
Инициализация сессии "Протокола 125" для режимов передачи с уведомлением и без

Сообщение замены временного общего ключа

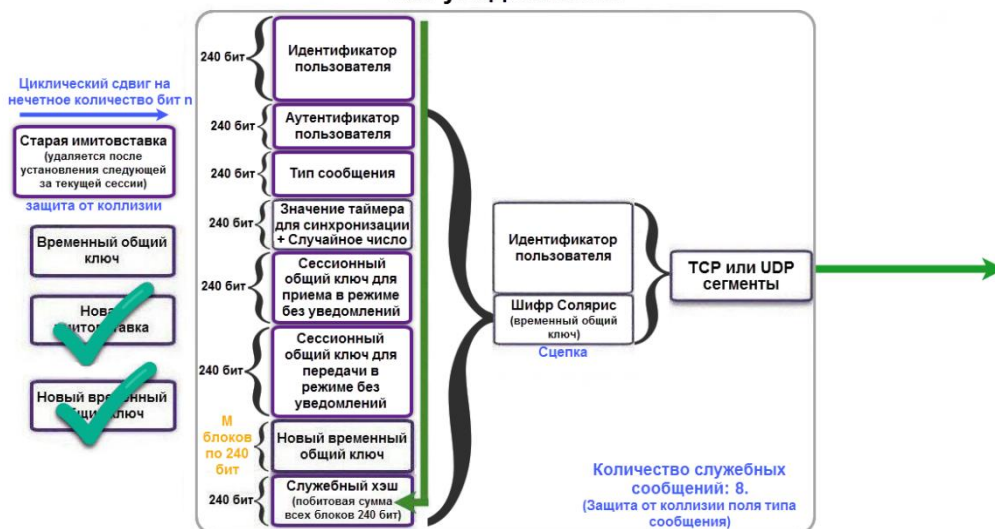


Рекомендуется информацию в любом служебном блоке из 240 бит кодировать не порядковыми числами, а случайными, размером 240 бит.

Сообщение уведомления о корректности доставки замены временного общего ключа

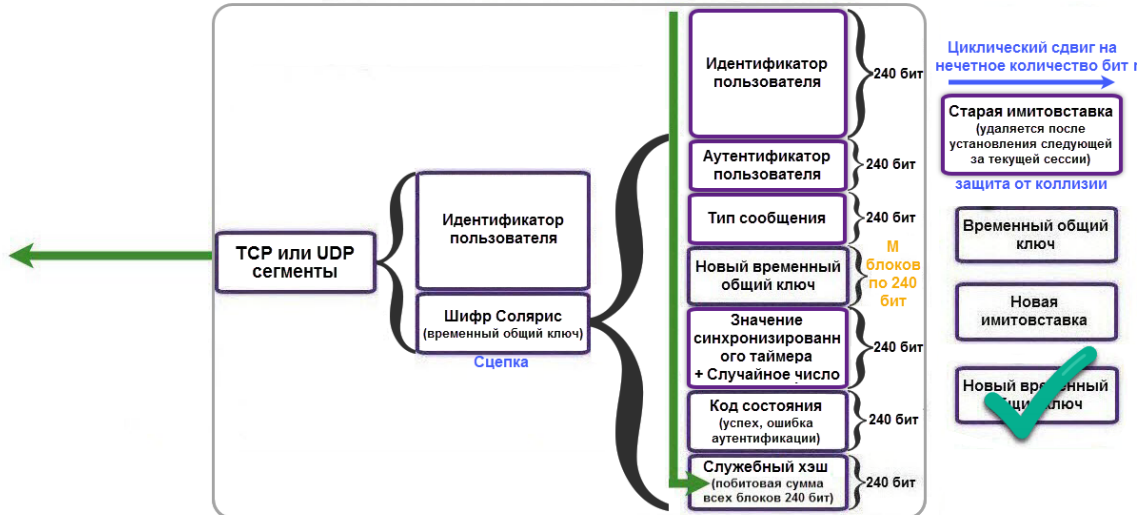


Сообщение замены параметров режима без уведомлений

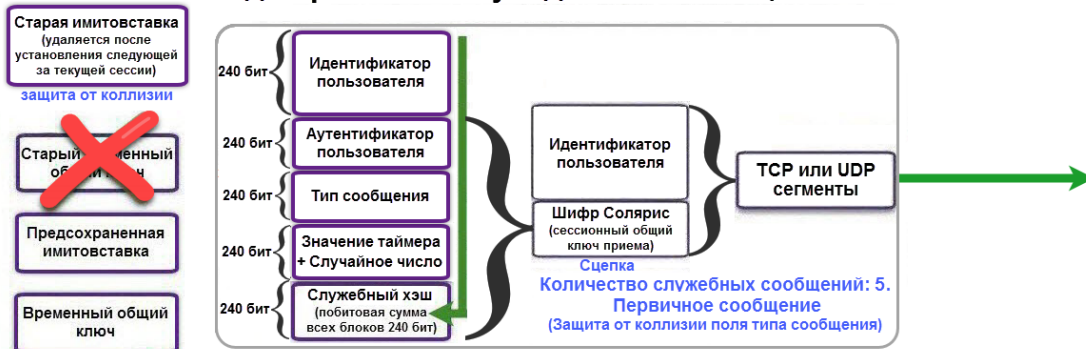


Рекомендуется использовать отдельный конфигурационный ключ для работы в режиме без уведомлений, потому что в нем постоянно шифруется таймер и длительное время не меняется сессионный общий ключ.

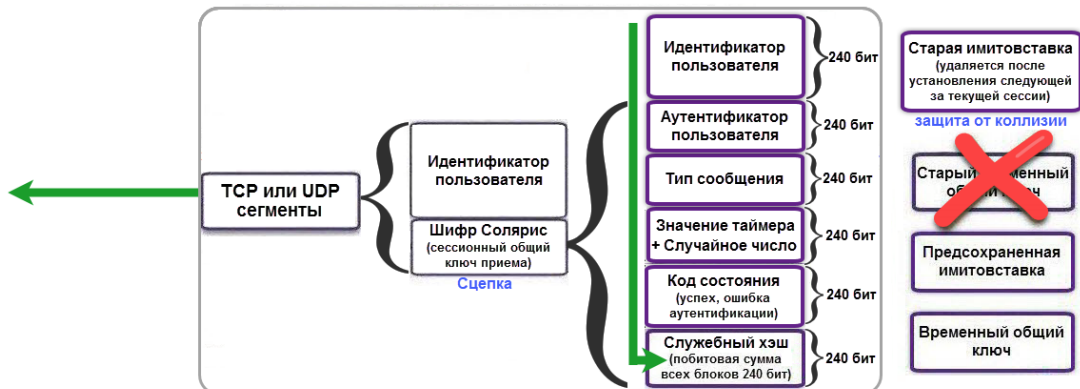
Сообщение уведомления о корректности доставки замены параметров режима без уведомлений



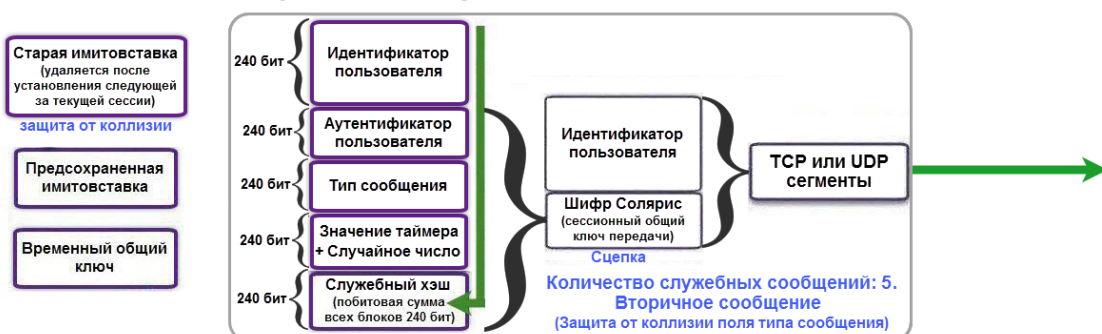
Проверка передачи сессионного общего ключа приема для режима без уведомления. Защита от коллизии



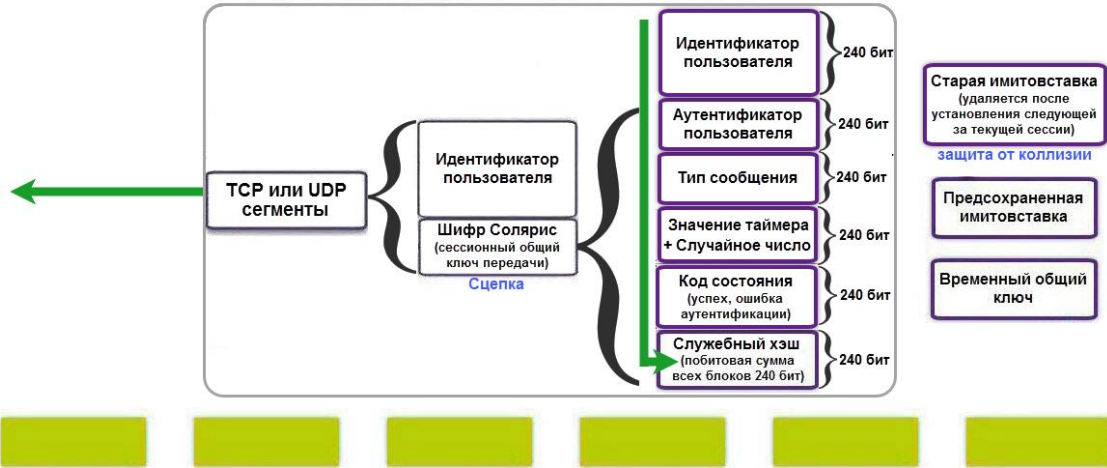
Уведомление проверки передачи сессионного общего ключа приема для режима без уведомления. Защита от коллизии



Проверка передачи сессионного общего ключа передачи для режима без уведомления. Защита от коллизии

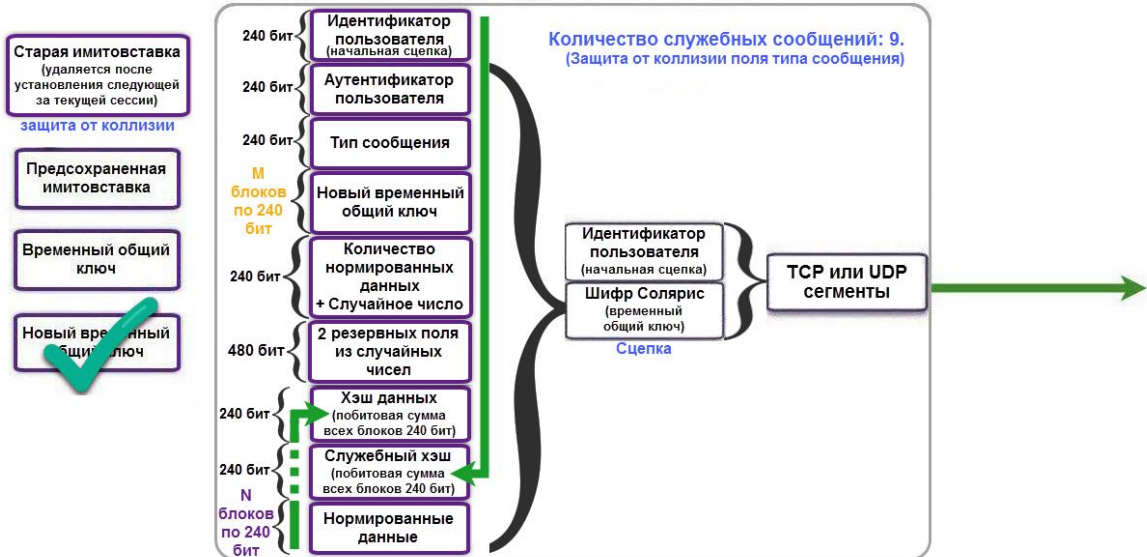


**Уведомление проверки передачи сессионного общего ключа
передачи для режима без уведомления. Защита от коллизии**



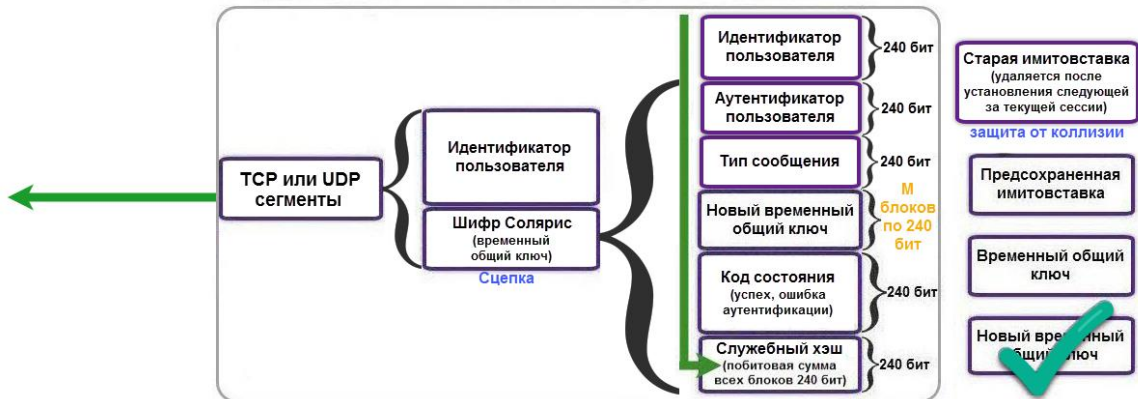
Завершение инициализации сессии. Сессия действительная до потери текущих временных общих ключей в оперативной памяти или до возникновения случайной коллизии по хэшу.

Сообщение с данными №1

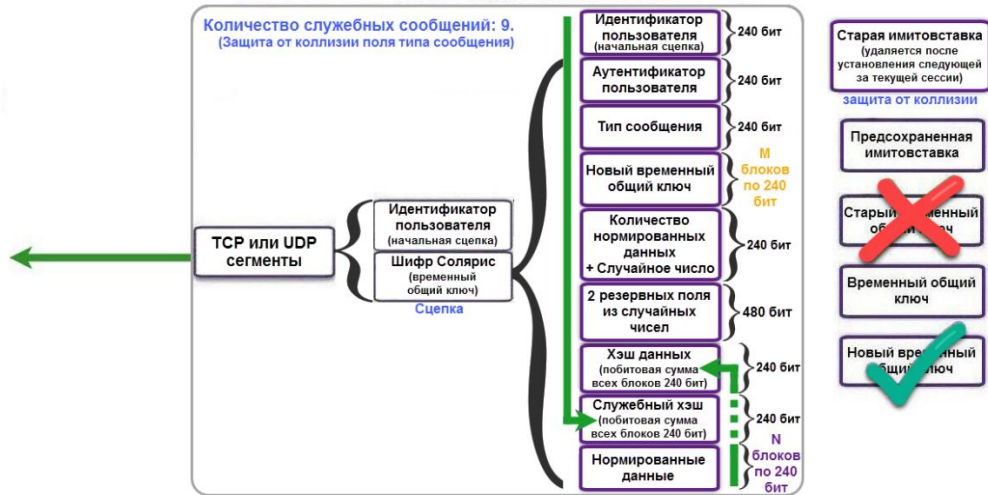


Если после шифрования данных зашифрованные пакеты не проходят тесты, то рекомендуется использовать предварительное сжатие данных

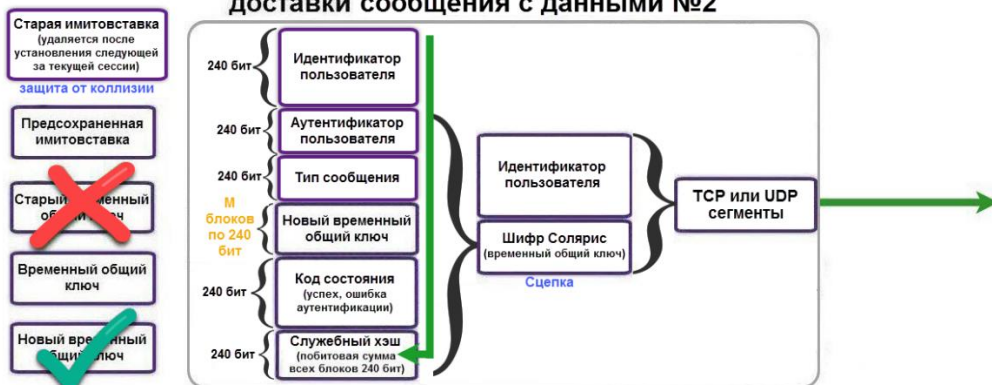
Сообщение уведомления о корректности доставки сообщения с данными №1



Сообщение с данными №2

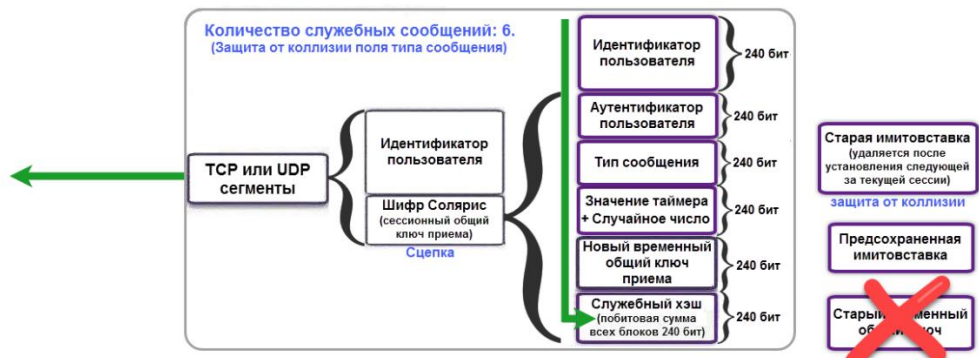


Сообщение уведомления о корректности доставки сообщения с данными №2

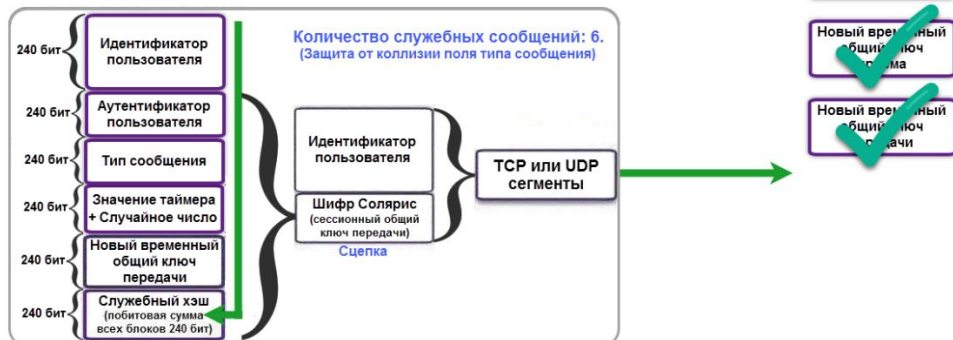


Сообщение без уведомления может приниматься и передаваться одновременно

Сообщение временного общего ключа приема для сообщения с данными №3 в режиме без уведомления

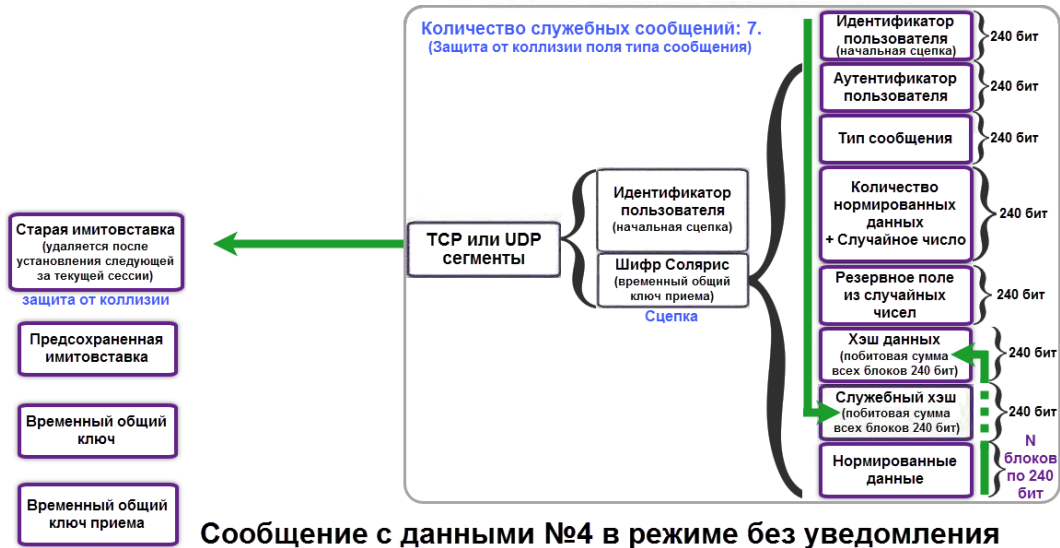


Сообщение временного общего ключа передачи для сообщения с данными №4 в режиме без уведомления

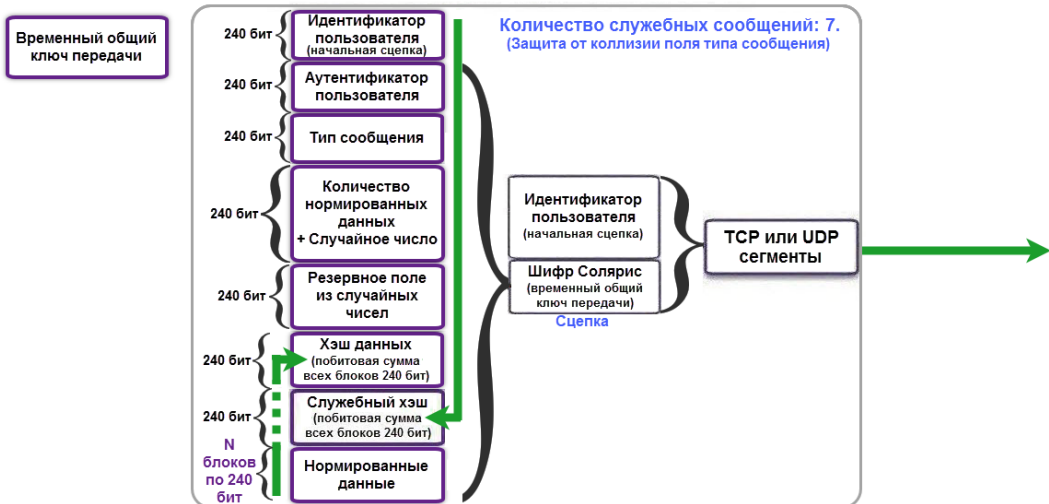


Сообщение без уведомления может приниматься и передаваться одновременно

Сообщение с данными №3 в режиме без уведомления



Сообщение с данными №4 в режиме без уведомления

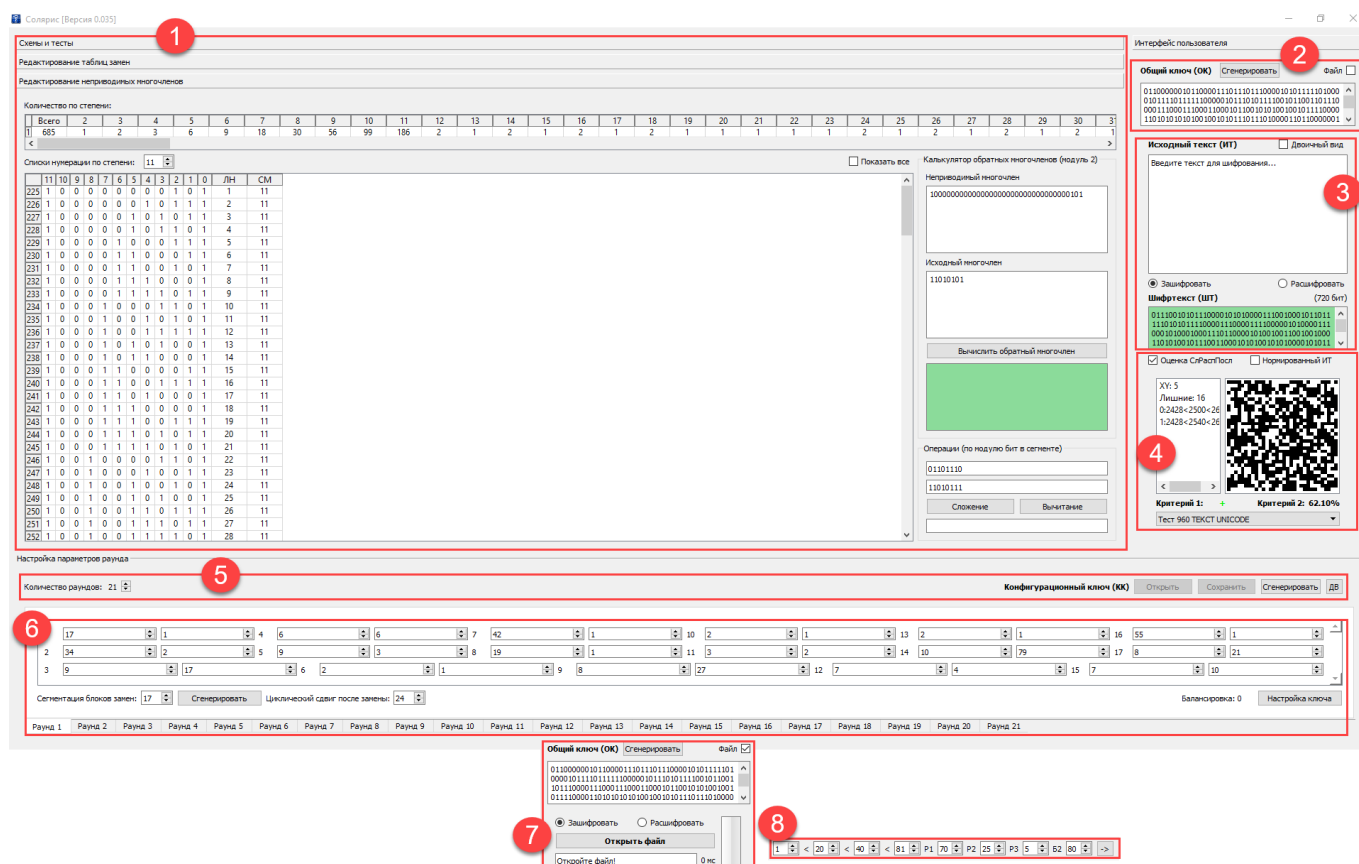


Остальные сообщения с данными и уведомления.

- Старая имитовставка (удаляется после установления следующей за текущей сессии) защита от коллизии
- Предсохраненная имитовставка
- Временный общий ключ
- ~~Временный общий ключ приема~~
- ~~Временный общий ключ передачи~~
- ✓

Параметры шифратора и генератора ключей Протокола 125.

В области один «Шифратора 125» можно увидеть справочную информацию и схемы его работы, также здесь находится интерфейс управления неприводимыми многочленами, по которым осуществляется замена. Кроме этого, размещены интерфейсы для работы отдельных функций алгоритма преобразования для того, чтобы можно было осуществить ручную поэтапную проверку корректности шифрования блоков.



В области два находится общий ключ, максимальная длина которого определяется по количеству раундов. В автоматическом режиме генерируется максимально возможный по длине ключ. Также, в области четыре после генерации отображаются результаты тестирования данного ключа по двух мощным тестам, взаимодополняющим и исключающим недостатки друг друга. Если задать ключ не кратный 240 бит он будет копироваться до тех пор, пока не станет размером кратным 240 бит, если

остаточное количество бит меньше введенного ключа берутся первые биты, необходимые для обеспечения итоговой кратности длины ключа. Также в области два можно выбрать режимы шифрования файлов из области семь.

В области три находится окно ввода коротких текстов для оперативного получения шифртекстов и результатов их тестов. Также, здесь можно осуществить расшифрование.

В области четыре, есть функция просмотра того, как выглядит нормированный исходный текст и параметр отключения режима тестирования сгенерированных последовательностей или шифртекстов. В том числе здесь отображаются подробные результаты тестирования сгенерированных последовательностей и шифртекстов.

В области пять можно включить режим сцепки блоков, указать количество раундов шифра сохранить, открыть или сгенерировать конфигурационный ключ. При нажатии кнопки ДВ открывается область 8.

В области шесть происходит конфигурация раундов. Если нажата кнопка сгенерировать из области пять, то здесь будут предустановлен случайный конфигурационный ключ, который можно изменить. Существует два окна конфигурации раундов. Изначально открывается конфигуратор этапа замены блоков. Под каждой цифрой (номер подблока) находится два окна. Первое окно – это параметр количества бит в подблоке. Все параметры должны давать значение балансировки ноль, что означает корректный для работы конфигурационный ключ. Второе окно – это локальный номер (ЛН) неприводимого многочлена, который можно определить в области один, где указать в поле списки нумерации по степени значения количества бит в подблоке.

Ниже идет сегментация блоков замен, что определяет количество подблоков для конфигурации. Кнопка сгенерировать для случайной генерации этапа замены одного раунда, параметр циклического сдвига после замены, проверка балансировки и переключения в режим конфигурации этапа сложения с ключом, где все тоже самое, за исключением отсутствия второго окна выбора неприводимого многочлена из базы данных. Также сдвиг там не после сложения с ключом для получившегося блока, а сдвиг общего ключа, который может задать уникальное смещение внутри общего ключа, это специфический параметр необходимый для обеспечения универсальности

алгоритма. Рекомендуется его, либо не задавать, либо задавать таким образом, чтобы не происходило наложение ключей. Но можно сделать и не предсказуемый сдвиг для каждого раунда.

В области семь можно открыть файл для шифрования или расшифрования. В области восемь увидеть расширенные параметры генератора конфигурационного ключа. На рисунке указаны следующие значения: $1 < 20 < 40 < 81$ P1 70 P2 25 P3 5 B2 80. Их можно интерпретировать так: вероятность количества подблоков в диапазоне от 1 до 20 равна 70%, в диапазоне от 20 до 40 равна 25%, в диапазоне от 40 до 81 равная 5%.

Последний параметр B2 определяет значение степени неприводимого многочлена, до которого будет проверяться отсутствие сегментов замены в количестве большем, чем количество неприводимых многочленов заданной степени в базе данных. Это связано с тем, что чаще всего генерируется большое количество бит в одном подблоке, а остальные уже изменить невозможно, потому что будет нарушение балансировки. Решить эту проблему можно уменьшением самого большого числа на случайное значение и новое перераспределение значений блоков. При новой итерации выбирается подблок, аналогично, с наибольшим числом количеством сегментов в качестве нового донора.

Рекомендуется после работы генератора конфигурационных ключей вносить изменения в некоторые конфигурации раундов вручную для обеспечения наилучшего результата случайности конфигурационного ключа.

Передача симметричного ключа по квантовому каналу связи.

Для создания защищенной передачи данных на основе разрабатываемого алгоритма симметричного шифрования необходимо рассмотреть возможности обмена ключами: от передачи при личной встрече, до автоматизированной многопользовательской системы. А также, определить роль алгоритма шифрования в подобных системах.

Квантовые вычисления благодаря нелинейному росту своей мощности, возможно, смогут не только находить в любом числе, являющимся произведением двух простых, эти исходные два множителя быстрее того, как это сообщение будет зашифровано, но и делать подобное с любым математическим алгоритмом, имеющим строгое условие выполнения и экспоненциальный рост сложности, например, проверка равенства двух чисел после выполнения определенных преобразований. Все вышесказанное теоретически может свести на нет любой ассиметричный способ шифрования, остается возможность использования только симметричного или квантового.

По квантовому каналу можно передавать конфиденциальную информацию с гарантией того, что она будет, либо безопасно передана, либо потеряна. Но потеря может произойти либо случайно, либо преднамеренно, после перехвата злоумышленником. Какое из этих двух событий произошло мы точно не узнаем, а сможем лишь строить вероятностные предположения, исходя из анализа стандартных потерь в канале передачи. Также нужно учитывать, что квантовый канал всегда будет медленным, и чем больше расстояние, тем медленнее. При дальности 150 км скорость передачи может составить 10 бит/с, а на 50 км – примерно 10 кбит/с.

В случае симметричного шифрования, можно добиться такой надежности шифра, что, теоретически, при полном переборе всех возможных ключей будет получаться огромное множество логически связанных сообщений, определить какое из них истинное ни злоумышленник, ни компьютер не сможет. Даже сообщения, имеющие известный формат пакета протокола передачи данных, после перебора всего множества и фильтрации данных по известной структуре, все равно, в поле данных будут

оставлять достаточное множество вариаций, если у пакета небольшая длина (а большая длина для пакетов протокола обычно и не требуется). Также, есть возможность использования временных ключей и отправлять их не только в начале сессии, но и через определенное количество пакетов, в том числе, для каждого нового пакета.

Угроза расшифровки может появиться при накоплении большого количества пар открытого и зашифрованного текста, но получить такие пары злоумышленник сможет только на вычислительных машинах, которые будут производить шифрование, а этого можно избежать, тем более, в случае заражения компьютера защита канала связи теряет смысл даже при передаче всей информации по квантовому каналу. Но и в таком случае возможно создать такой алгоритм симметричного шифрования, который после нескольких раундов будет давать непредсказуемые результаты, иначе говоря, закономерности будут практически отсутствовать, даже нейронная сеть не сможет их определить в достаточном качестве для восстановления исходной информации.

При использовании квантового канала и «Протокола 125», нам не имеет смысла определять был ли ключ передан или прослушан злоумышленником. Ключ, который сгенерирован передается по квантовому каналу, затем сторона-получатель шифрует им хэш от текста и посылает по не защищенному каналу связи стороне-отправителю текст и шифр хэша текста, затем сторона-отправитель вычисляет хэш от текста, шифрует его и сверяет с полученным шифром хэша текста. Если шифры хэша не совпали, генерируется новый ключ и отправляется. Так происходит до тех пор, пока шифры хэша не совпадут, либо не будет превышен лимит повторной передачи ключа, который укажет, что канал передачи недоступен или неисправен.

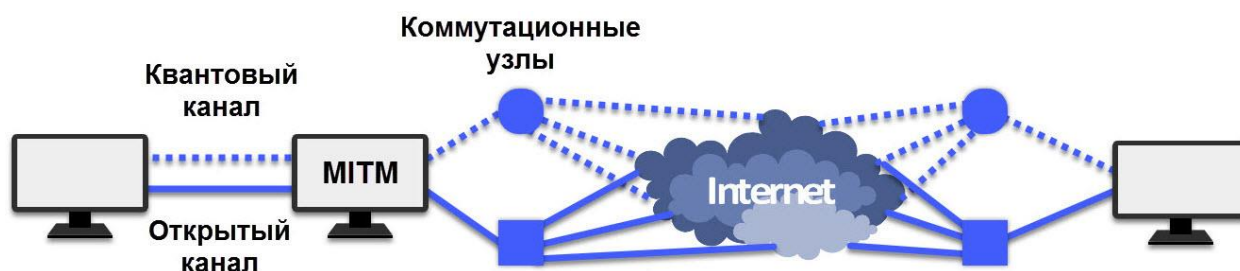
Это для случая поляризации фотона, а в случае EPR эффекта симметричного атома, можно будет сгенерировать им квантовый симметричный ключ размером с ключ от «Шифратора 125» и произвести операцию XOR этих ключей, передать результат по обыкновенному каналу связи. Дальше осуществить проверку корректности отправки по ранее описанному способу.

Также, возможно организовать многопользовательскую систему передачи по квантовому каналу, в которой коммутацию можно будет осуществлять физическим

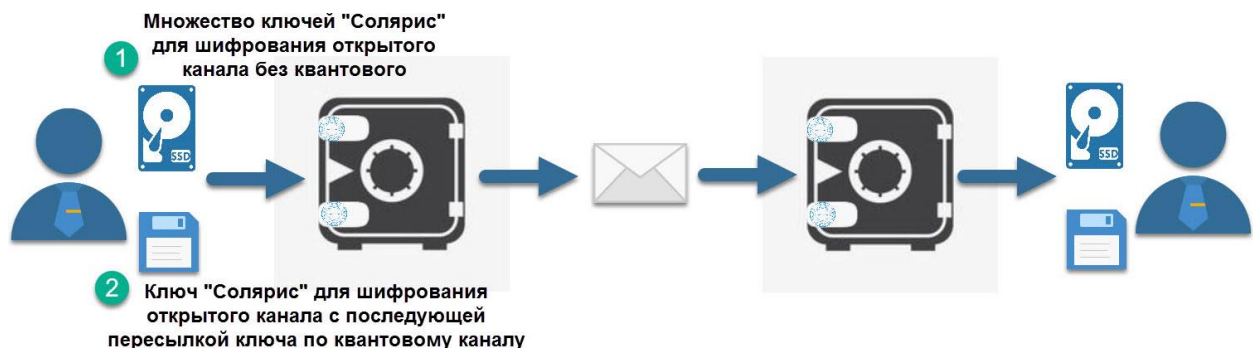
переключением каналов. Таким образом, по запросу пользователя, ему будет на некоторое время проложен маршрут до необходимого ему ресурса. Это может быть сервер сайта, на который будет отправлен ключ шифрования при регистрации пользователя или замены уже существующего ключа на новый. Практическая применимость данной многопользовательской системы будет расти с повышением расстояния квантовой передачи данных по безразрывному кабелю связи, уменьшения потерь на стыках при физической коммутации каналов.

Отдельно стоит рассмотреть уязвимость вышеописанной системы, которая заключается в возможности осуществления атаки MITM, если злоумышленник производит одновременный перехват квантового и открытого канала, по которому пересылается дополнительная информация для завершения процедуры обмена ключами.

Например, он может расположиться сразу за контролируемой зоной, в местах, где проложены кабели квантового и открытого канала. Даже, если они будут находиться на большом расстоянии друг от друга, это усложнит задачу атаки, но не исключит ее.



Первый способ борьбы с данной атакой технологически самый простой, но требует большого временного ресурса для его осуществления и практически не реализуем в многопользовательской системе. Заключается он в полном отказе от использования квантового канала, генерации большого количества симметричных ключей и сохранения их на носитель информации большого объема.

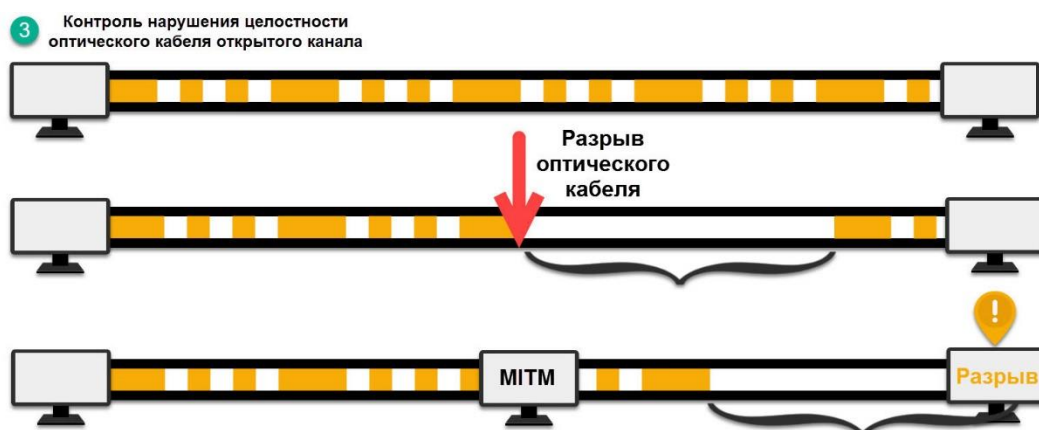


Далее, необходимо передать носитель информации лично, либо поместить его в контейнер, нарушение целостности которого будет легко замечено (опечатанный сейф). После отправки транспортной службой получателю, он сможет проверить целостность контейнера, вскрыть его и извлекать новый ключ каждый раз, когда будет оговорена необходимость замены ключа. Выбор ключей может осуществляться в порядке их следования, либо по номеру в общем списке, выбранный ключ вычеркивается и больше не используется.

Второй способ заключается в обмене только одним ключом, аналогично, при личной встрече, либо при помощи транспортной службы. После получения ключа появляется возможность защитить открытый канал симметричным ключом алгоритма. При потребности сменить ключ, его можно отправить по квантовому каналу, а дополнительную информацию необходимую для однозначного определения ключа с квантового канала, послать по защищенному, ранее, открытому каналу. Таким образом, даже если ключ будет скомпрометирован после передачи нового ключа, злоумышленник не сможет восстановить новый ключ, а только восстановит дополнительную информацию, которая не несет в себе содержимое ключа. Этот способ тоже достаточно сложен для использования в многопользовательской системе.

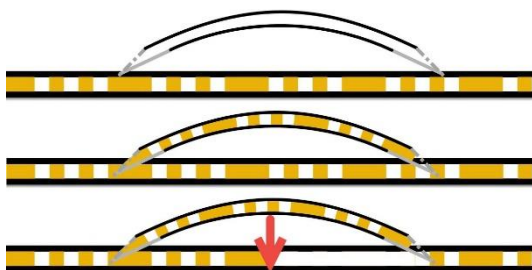
Третий способ заключается в использовании оптоволокна для открытого канала связи и обеспечения возможности контроля разрыва данного вида кабеля. Предполагается, что по кабелю постоянно осуществляется передача данных, либо сообщений, либо сигнала особой последовательности во время простоя. Злоумышленник может прослушивать оптический канал перенаправляя небольшое количество фотонов

в свой приемник, но это никак не скажется на надежности обмена ключевой информацией.



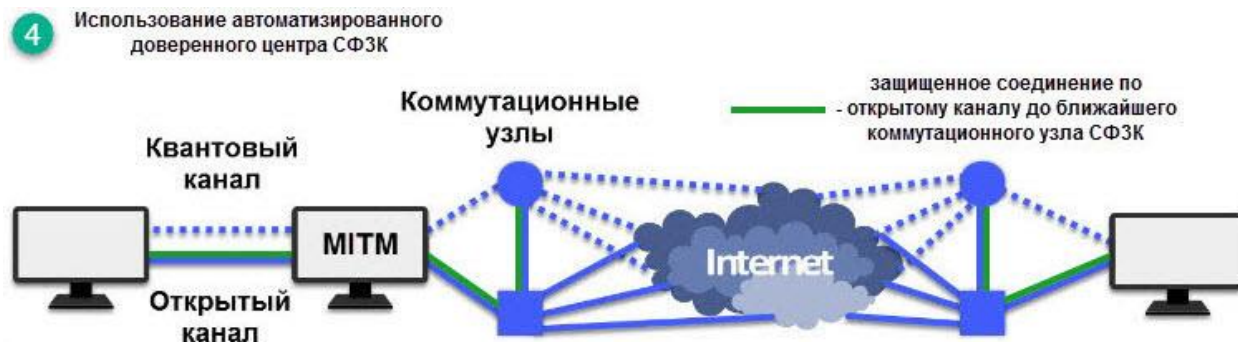
Для осуществления атаки злоумышленнику недостаточно прослушивать канал или исказить его, ему необходимо осуществить врезку по середине, чтобы иметь возможность подменять информационный сигнал. Непосредственно в тот момент, когда он попытается разорвать оптический канал и закрепить свое оборудование, произойдет нарушения корректности передаваемой информации, по которому можно будет определить, что случился разрыв кабеля и заблокировать любой обмен по нему, возможно, даже определить расстояние до участка разрыва (врезки).

Данный метод имеет недостатки в виде необходимости постоянного поддержания подачи светового сигнала по оптоволокну и обеспечения контроля целостности кабеля на конечных узлах в многопользовательской системе. Также, теоретически, остается возможность осуществления более технологичной врезки. В итоге, можно сделать вывод, что способ недостаточно практичен в использовании.



Четвертым способом может являться создания защищенного соединения, по открытому каналу, оконечной станции с автоматизированной системой физической коммутации квантовых каналов (СФЗК). При регистрации пользователя в системе, на

ближайшем коммутационном узле СФЗК, ему разово выдается набор симметричных ключей, которые он использует вручную или через специальное устройство для связи со СФЗК. Последним ключом из этого списка передается новый список ключей в СФЗК.



Через данный защищенный канал передается дополнительная информация о начале и завершении передачи данных по квантовому каналу, что позволяет добиться невозможности для злоумышленника осуществить получение ключа, а затем этот же ключ передать получателю. Также, по окончании передачи всего ключа, компьютер получателя шифрует им хэш текста и возвращает вместе с данным текстом отправителю через защищенной соединением СФЗК для проверки идентичности ключей на обоих компьютерах. Таким образом СФЗК выступает в роли автоматизированного доверенного центра. Между каждым коммутационным узлом СФЗК, также, проложено защищенного соединения.

Передаваемый ключ ни каким образом не попадает в СФЗК, так как передается по квантовому каналу, который коммутируется физическим (механическим) способом. Дополнительная связь с СФЗК обеспечивает лишь защиту от MITM. Осуществить MITM можно только получив удаленный доступ к автоматизированной системе СФЗК при наличии закладок в программном обеспечении и одновременной врезке в квантовый канал только в момент непосредственной передачи ключа. Для защиты от закладок необходимо разрабатывать системы экспертного и народного контроля. Например, перед опломбированием устройства производить публичную проверку топологии микросхем и компиляцию открытого кода с проверкой его хэш-суммы.

В итоге, можно сказать, что реализация системы безопасного обмена данными при помощи квантового канала не так уж проста и имеет множество уязвимостей. Но их можно практически полностью исключить, даже, в многопользовательской системе. Также, немаловажной является и надежность алгоритма шифрования, которая влияет на безопасность системы, в целом.