Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Владимирский государственный университет
имени Александра Григорьевича и Николая Григорьевича Столетовых»
(ВлГУ)

На правах рукописи

Метлинов Александр Дмитриевич

МОДЕЛИ И АЛГОРИТМЫ ПОВЫШЕНИЯ КРИПТОСТОЙКОСТИ И ПРОИЗВОДИТЕЛЬНОСТИ ЗАЩИЩЕННОГО КАНАЛА СВЯЗИ В ТЕЛЕКОММУНИКАЦИОННЫХ СЕТЯХ *TCP/IP*

Специальность: 05.12.13 – Системы, сети и устройства телекоммуникаций

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель:
Монахов Михаил Юрьевич
д.т.н., профессор

Владимир 2018

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность и степень разработанности темы. Содержание проблемы информационной безопасности (ИБ) и защиты информации (ЗИ) в системах и сетях телекоммуникаций интерпретируются следующим образом. По мере развития и усложнения моделей, алгоритмов и средств обработки и передачи информации в защищенных каналах связи (КС) телекоммуникационных сетей *TCP/IP* повышается уязвимость существующих протоколов безопасности КС, напрямую влияющая на возможность несанкционированного копирования, уничтожения, блокирования или искажения информации.

Основные угрозы ИБ направлены на перехват и имперсонацию сообщений (нарушение конфиденциальности и целостности передаваемых данных), нередки атаки на доступность узлов канала и их подмену. КС сетей *TCP/IP* не имеют встроенных средств защиты, существующие механизмы обеспечения ИБ реализованы на сеансовом уровне *OSI* и имеют множество уязвимостей.

Проблема ИБ и ЗИ в системах и сетях телекоммуникаций исследовалась в трудах ведущих российских ученых Герасименко В.А., Домарева В.В., Иванова М.А., Завгороднего В.И., Лукацкого А.В., Медведковского И.Д., Мурина Д.М., Никитина О.Р., Панасенко С., Соколова А.В., Черемушкина А.В., Шаньгина В.Ф., Хорева А.А. Значительный вклад в решение выделенной проблемы внесли зарубежные исследователи Адлеман Л., Блэкберн Р., Брикелл Э., Грэм Р., Гудман Р., Вастон А.С., Калиф М., Касахар М., Кимур М., Кобаяс К., Маколи Э., Минхуа Ц., Мерфи Ш.П., Нидеррайтер Х., Нгуен Ф., Нием В., Патерсон С., Ривест Р.Л., Хусейн А., Шамир А., Шеннон К., Шнайер Б., Шор Б., Штерн Ж.Г и другие.

Анализируя результаты исследований, можно сделать вывод, что наиболее перспективным подходом в обеспечении безопасности передаваемых сообщений является использование криптостойкого шифрования *TCP/IP*-потока. Это не позволяет злоумышленнику анализировать служебную информацию и уменьшает вероятность доступа к незашифрованным данным в узлах сети. Стандартные протоколы *SSL* и *TLS* обеспечивают криптозащиту КС, но имеют недостаточную

производительность и относительно низкую криптостойкость. Для повышения производительности и криптостойкости защищенного КС перспективным является использование средств симметричной рюкзачной криптографической системы.

Таким образом, исследования, направленные на разработку новых моделей и алгоритмов повышения криптостойкости и производительности защищенного КС на базе симметричной рюкзачной криптографической системы в сетях *TCP/IP*, актуальны и имеют практическое значение в решении проблемы обеспечения ИБ сетей телекоммуникаций предприятий.

Объект исследования – защищенный КС в телекоммуникационных сетях *TCP/IP*.

Предмет исследования – методы и средства криптографической защиты информации в КС сетей *TCP/IP*.

Цели и задачи работы. Целью работы является повышение криптостойкости и производительности защищенного КС в телекоммуникационных сетях *TCP/IP*.

В соответствии с целью были поставлены и решены следующие научные задачи:

- анализ методов и средств обеспечения ИБ и ЗИ в КС телекоммуникационных сетей *TCP/IP*, выявление особенностей организации защищенного КС при помощи криптографических протоколов *SSL* и *TLS*;
- разработка семейства моделей и алгоритмов повышения производительности и криптостойкости симметричных рюкзачных криптосистем в защищенных КС сетей *TCP/IP*;
- модификация моделей симметричных рюкзачных криптосистем в защищенных КС сетей *TCP/IP* семейством *CBC*-блочных алгоритмов шифрования и дешифрования информации;
- экспериментальное исследование предложенных средств и внедрение результатов работы.

Научная новизна. Получены следующие научные результаты:

- разработаны математические модели рюкзачной системы защиты КС, отличающаяся наличием общей памяти (ОП) между узлом-отправителем и узлом-получателем, высоким уровнем плотности укладки рюкзака, использованием линейно-рекуррентных последовательностей, позволяющие повысить криптостойкость и производительность КС (п.5 и п.10 паспорта 05.12.13);

- модифицирован протокол *TLS* путем введения в структуру данных полей для хранения ОП, а также добавления модулей шифрования и дешифрования, реализующих рюкзачную систему защиты, что позволяет повысить его (протокола) функциональные свойства (п.5 и п.10 паспорта 05.12.13);

- разработаны алгоритмы передачи и приема сообщений в защищенном КС в телекоммуникационных сетях *TCP/IP*, построенном на базе рюкзачной системы защиты с ОП (п.5 и п.10 паспорта 05.12.13).

Практическая значимость работы. Разработано информационное и программное обеспечение комплекса алгоритмов симметричной рюкзачной криптосистемы с ОП, включающее: программный комплекс *CBC*-блочной симметричной рюкзачной криптологической системы для вариации с плотностью укладки больше единицы при величине рюкзачного базиса 128 бит (свидетельство о гос. регистрации программы для ЭВМ №2014614981); программный тестовый комплекс для симметричной рюкзачной криптосистемы (свидетельство №2014614937); программный модуль генератора общей памяти для симметричной рюкзачной криптосистемы (свидетельство №2015616165). Внедрение разработанной криптосистемы в фазы работы стандартного протокола *TLS* (аутентификации клиента и сервера, создания кода аутентификации сообщений и работы симметричных блочных алгоритмов шифрования и дешифрования сообщений) позволяет повысить эффективность защищенного КС сетей *TCP/IP*: канал работает в среднем на 7-9% быстрее, чем использующий *DH_AES* в стандартном *TLS* и на 25-28% быстрее, чем рюкзачная криптосистема, в основе которой лежит супервозрастающий базис независимо от типа и объема исходных данных. Разработанная криптосистема с общей памятью при минимальном размере

рюкзачного базиса в 64 бита имеет плотность укладки в пределах $0.9907 \leq \rho \leq 0.9989$, что удовлетворяет современным требованиям криптостойкости.

Результаты исследований внедрены в ООО «Русский мастер», Владимирская область, поселок Льнозавод; в ООО «ДИВАНиЯ», Владимирская область, поселок Льнозавод, а также в ИП Щерба А.Ю., город Владимир.

Методология и методы исследования. В данном исследовании были использованы методы криптографии, линейной и нелинейной алгебры, теории сложности алгоритмов, математического анализа, математической статистики и теории вероятности.

Положения, выносимые на защиту:

- математические модели рюкзачной системы защиты КС;

- модификация протокола *TLS*;

- алгоритмы передачи и приема сообщений в защищенном КС в телекоммуникационных сетях *TCP/IP*;

- результаты экспериментального исследования и внедрения предложенных моделей и алгоритмов.

Степень достоверности результатов исследований. Основные результаты, полученные в работе, являются обоснованными либо на доказательном, либо на экспериментальном уровне. Достоверность практических результатов достигается за счет большого количества экспериментов при решении задач и использования собственного и стандартного программного обеспечения.

Апробация работы. Материалы диссертационной работы докладывались и обсуждались на:

- IX Международной научной конференции «Перспективные технологии в средствах передачи информации» (Владимир 2013);

- XI Международной научной конференции «Перспективные технологии в средствах передачи информации» (Владимир 2015);

- I Международной научно-практической конференции «Информационная безопасность в свете Стратегии Казахстан - 2050» (Астана 2013);

- II Международной научно-практической конференции «Информационная безопасность в свете Стратегии Казахстан - 2050» (Астана 2014);

- III Международной научно-практической конференции «Информационная безопасность в свете Стратегии Казахстан - 2050» (Астана 2015);

- XXXIII Всероссийской НТК «Проблемы эффективности и безопасности функционирования сложных технических и информационных систем» (Серпухов 2014);

- XXXVI Всероссийской НТК «Проблемы эффективности и безопасности функционирования сложных технических и информационных систем» (Серпухов 2017);

- V Всероссийской научно-технической конференции ИКВО НИУ ИТМО «Проблема комплексного обеспечения информационной безопасности и совершенствование образовательных технологий подготовки специалистов силовых структур» (Санкт-Петербург 2014);

- V международной научной конференции «Современные методы и проблемы теории операторов и гармонического анализа и их приложения V» (Ростов-на-Дону 2015) и другие.

Публикации: опубликовано 14 работ, 4 в изданиях из перечня ВАК. Получено 3 свидетельства о государственной регистрации программ для ЭВМ.

Личный вклад. Все результаты, изложенные в диссертации, получены автором лично или при его непосредственном участии. Постановка цели и задач, обсуждение планов исследований и результатов выполнены совместно с научным руководителем.

1 ЗАЩИЩЕННЫЙ КАНАЛ СВЯЗИ ТЕЛЕКОММУНИКАЦИОННЫХ СЕТЕЙ *TCP/IP*. АНАЛИЗ ОБЪЕКТА ИССЛЕДОВАНИЯ

В данной главе диссертационного исследования описываются объект и предмет исследования. Анализируются методы обеспечения информационной безопасности в каналах связи телекоммуникационных сетей *TCP/IP*, рассмотрены их основные элементы и структура, возможности, недостатки и уязвимости.

1.1 Объект и предмет исследования

Объектом диссертационного исследования является защищенный канал связи (КС) в телекоммуникационных сетях *TCP/IP* [59]. Канал связи (англ. *channel, data line*) — система технических средств и среда распространения сигналов для передачи данных (информации) от отправителя (источника) к получателю (приёмнику) [23, 25].

В [23] предложена модель КС, как модель линейной передачи сообщений (рисунок 1.1).

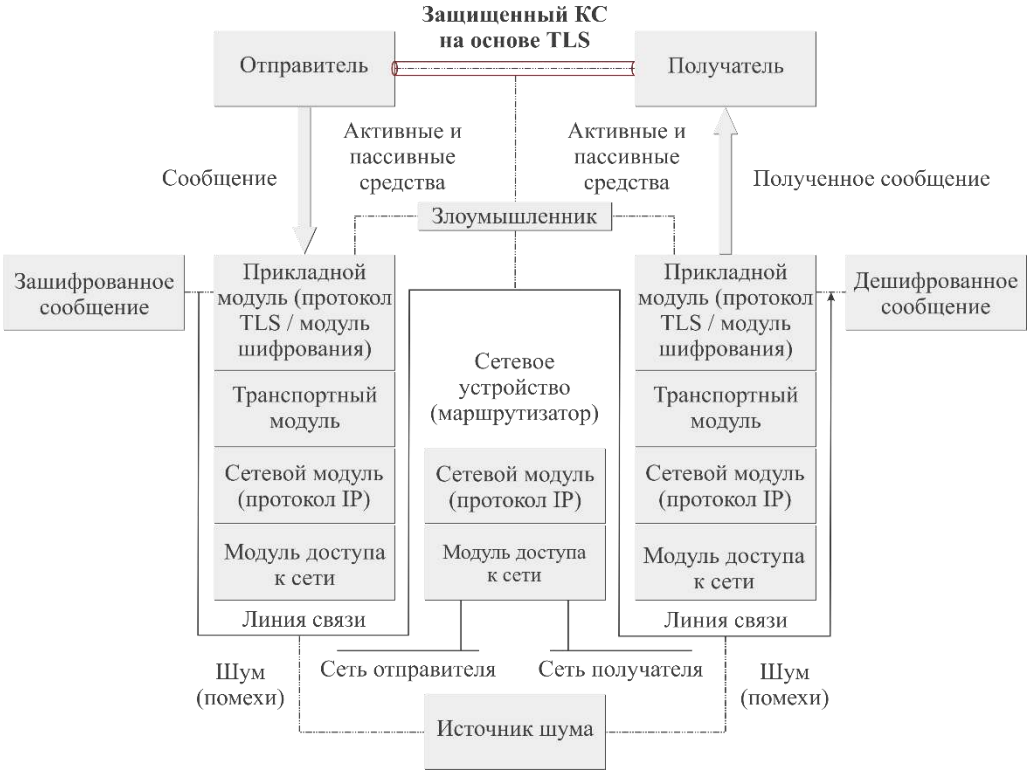

Рисунок 1.1 – Структурная схема канала связи сети *TCP/IP*

Выделим особенности КС [20, 21, 23, 52] в телекоммуникационных сетях *TCP/IP*, принципиальные для настоящего диссертационного исследования [7, 8, 25, 68]:

1. Основа телекоммуникационной сети – стек протоколов *TCP/IP*, поэтому часто такая сеть носит название «сеть *TCP/IP*».
2. С точки зрения обмена данными сеть *TCP/IP* представляет собой множество КС между отправителями и получателями.
3. КС не имеет встроенных средств защиты передаваемых сообщений.
4. Основные механизмы обеспечения ИБ КС реализованы на сеансовом уровне сетевой модели *OSI* и имеют множество уязвимостей [44, 58].
5. Основные угрозы и атаки в КС направлены на перехват передаваемых сообщений – угрозы конфиденциальности и целостности передаваемых данных [34, 57]. Незначительное количество атак относится к атакам на доступность узлов канала и их подмену.
6. Приоритетным направлением обеспечения ИБ КС в сетях *TCP/IP* является использование криптографических механизмов защиты [68, 69, 72, 74].
7. При обеспечении ИБ КС в сетях *TCP/IP* предполагается, что злоумышленник во внешней среде присутствует всегда и обладает неограниченными вычислительными ресурсами [39].

Предметом исследования диссертации являются методы и средства криптографической защиты информации в КС сетей *TCP/IP*.

1.2 Стек протоколов *TCP/IP*

Стек протоколов *TCP/IP* [11, 12] - набор сетевых протоколов передачи данных, используемых в сетях, включая сеть интернет.

Уровни протоколов *TCP/IP* расположены по принципу стека (рисунок 1.2).

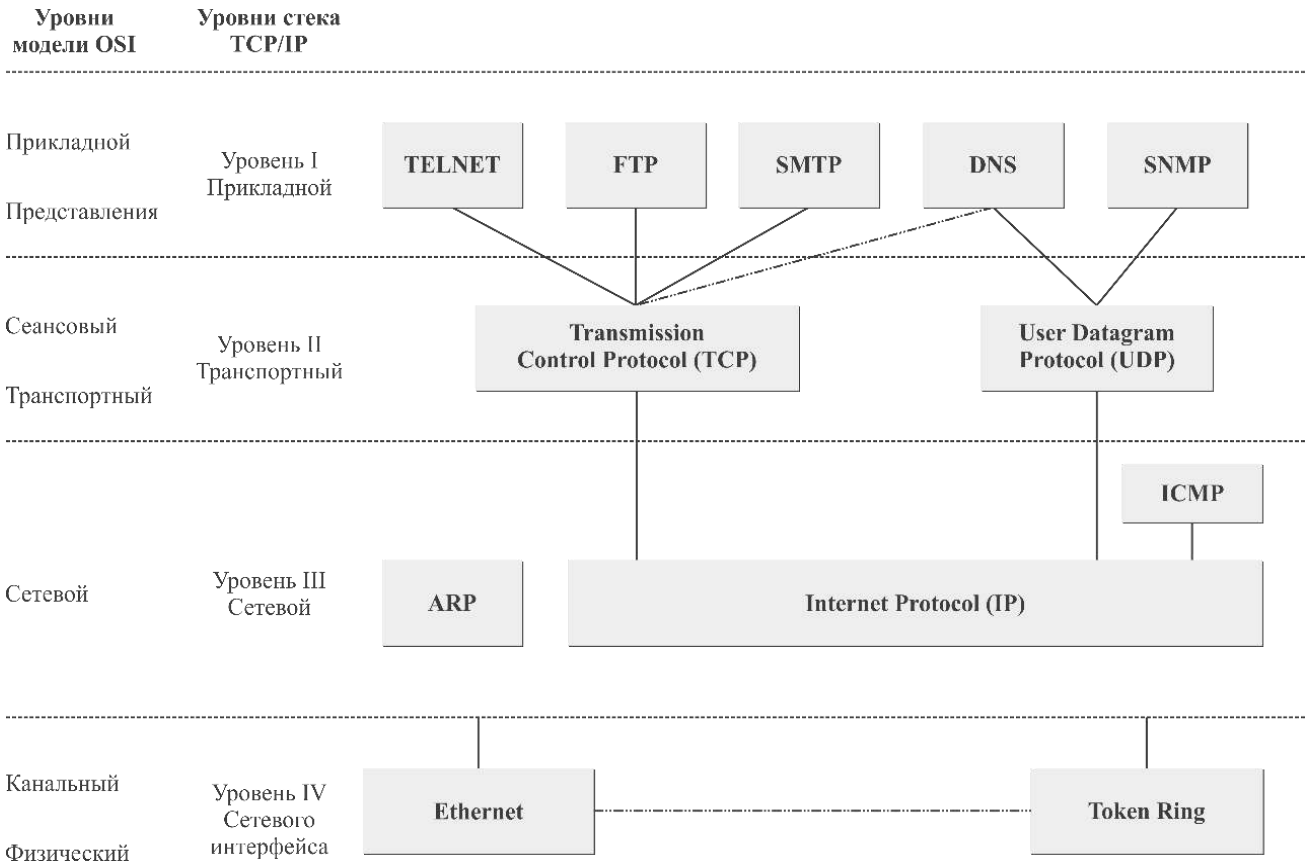

Рисунок 1.2 – Структура стека протоколов *TCP/IP*

Стек протоколов *TCP/IP* включает в себя четыре уровня:

1. Прикладной уровень (*application layer*) - *HTTP*, *RTSP*, *FTP*, *DNS*.

2. Транспортный уровень (*transport layer*) - *TCP*, *UDP*, *SCTP*, *DCCP*.

3. Сетевой уровень (*internet layer*) – *IP*.

4. Канальный уровень (*link layer*) - *Ethernet*, *IEEE 802.11 Wireless Ethernet*, *SLIP*, *Token Ring*, *ATM* и *MPLS*, физическая среда и принципы кодирования информации *T1*, *E1*.

На базе *TCP/IP* построено взаимодействие пользователей в *IP*-сетях. Стек является независимым от физической среды передачи данных, благодаря чему, в частности, обеспечивается полностью прозрачное взаимодействие между проводными и беспроводными сетями [15].

На прикладном уровне [13] работает большинство сетевых приложений. Протоколы транспортного уровня могут решать проблему негарантированной доставки сообщений, а также гарантировать правильную последовательность

прихода данных. В стеке *TCP/IP* транспортные протоколы определяют для какого именно приложения предназначены эти данные. Сетевой уровень разработан для передачи данных из одной сети в другую. Канальный уровень описывает, каким образом передаются пакеты данных через физический уровень, включая кодирование.

Единицей данных протокола *TCP* является сегмент. Информация, поступающая к протоколу в рамках логического соединения от протоколов более высокого уровня, рассматривается *TCP* как неструктурированный поток байтов. Поступающие данные буферизуются. Для передачи на сетевой уровень из буфера «вырезается» сегмент, состоящий из заголовка и блока данных [14, 20].

Для организации надежной передачи данных предусматривается установление логического соединения между двумя прикладными процессами. В рамках соединения осуществляется обязательное подтверждение правильности приема для всех переданных сообщений, и при необходимости выполняется повторная передача. Соединение в протоколе идентифицируется парой полных адресов обоих взаимодействующих процессов (оконечных точек). Адрес каждой из оконечных точек включает *IP*-адрес (номер сети и номер компьютера) и номер порта. Одна оконечная точка может участвовать в нескольких соединениях.

1.3 Методы и средства обеспечения информационной безопасности в каналах связи сетей *TCP/IP*

При эксплуатации технических средств КС возможны следующие дестабилизирующие факторы, определяющие угрозы нарушения конфиденциальности и целостности данных в КС [33, 40, 71]:

- Побочные электромагнитные излучения информативного сигнала от технических средств КС.
- Наводки информативного сигнала, обрабатываемого техническими средствами, на линии связи канала.

- Электрические сигналы или радиоизлучения, обусловленные воздействием на средства передачи информации высокочастотных сигналов, создаваемых с помощью разведывательной аппаратуры.
- Радиоизлучения или электрические сигналы от внедренных специальных электронных устройств перехвата информации (закладок) в канале, модулированные информативным сигналом.
- Радиоизлучения или электрические сигналы от электронных устройств перехвата информации, подключенных к каналам связи или техническим средствам обработки информации.
- Воздействие на технические или программные средства передачи информации и сам канал связи в целях уничтожения, искажения данных, работоспособности технических средств, средств защиты информации, адресности и своевременности информационного обмена.

Перехват информации в КС или воздействие на нее с использованием технических средств может вестись из зданий, расположенных в непосредственной близости, мест временного пребывания, а также с помощью скрытно устанавливаемой автономной аппаратуры.

Для открытого КС возможно несанкционированное подключение к линии. Возможно перехватить пароль, управляющую информацию, данные, передаваемые по КС.

Для закрытого КС возможно несанкционированное подключение к линии. Возможен перехват паролей, управляющей информации, момента установления защищенного соединения (ключей).

Действия злоумышленника, направленные против какого-либо узла (или, возможно, целой сети):

1. Перехват (и, возможно, модификация) данных, передаваемых через сеть от одного узла другому.
2. Имперсонация (узел злоумышленника выдает себя за другой узел, чтобы воспользоваться какими-либо привилегиями имитируемого узла).
3. Принуждение узла к передаче данных на завышенной скорости.

4. Приведение узла в состояние, когда он не может нормально функционировать, передавать и принимать данные (так называемая атака *DoS - denial of service*, отказ в обслуживании [68]).

Для достижения своих целей злоумышленник использует прослушивание (*sniffing*), сканирование сети и генерацию пакетов. Под генерацией пакетов понимается создание и отправка специально сконструированных дейтаграмм или кадров, позволяющих злоумышленнику выполнить ту или иную атаку. Особо выделим здесь фальсификацию пакетов - создание *IP*-дейтаграмм или кадров уровня доступа к сети, направленных якобы от другого узла (*spoofing*).

Методы обеспечения ИБ КС можно разделить на две группы [23, 24]:

- Основанные на ограничении физического доступа к КС (включая аппаратуру связи).
- Основанные на преобразовании сигналов в линии к форме, исключающей (затрудняющей) для злоумышленника восприятие или искажение содержания передачи.

Методы первой группы в основном находят применение в системах правительственной связи, где осуществляется контроль доступа к среде передачи данных [23, 24].

Методы второй группы направлены на обратимое изменение формы представления передаваемой информации. Преобразование должно придавать информации вид, исключающий ее восприятие при использовании аппаратуры, стандартной для данного КС.

Для открытого КС обеспечение ИБ основано на ограничении доступа к линии связи [70].

Для закрытого КС безопасность информации основана как на ограничении доступа к линии связи, так и на криптографической защите. Данные «закрываются» с использованием различных алгоритмов шифрования.

Большинство угроз перехвата и имперсонации можно предотвратить, используя фильтрацию на маршрутизаторе (или соответственно настроенный прокси-сервер), превентивное сканирование сети и анализ сетевого трафика.

Фильтры на маршрутизаторе, соединяющем сеть предприятия с интернетом, применяются для запрета пропуска дейтаграмм, которые могут быть использованы для атак как на сеть организации из интернета, так и на внешние сети злоумышленником, находящимся внутри организации. Фильтрация *TCP*-сегментов выполняется в соответствии с политикой безопасности. Если, например, во внутренней сети нет хостов, к которым предполагается доступ из интернета, но разрешен доступ внутренних хостов в интернет, то следует запретить пропуск *TCP SYN*-сегментов, не имеющих флага *ACK*, из интернета во внутреннюю сеть.

Атаки на *TCP/IP* можно разделить на два вида [59]: пассивные и активные. Атака типа подслушивание заключаются в перехвате сетевого потока и его анализе. Для осуществления подслушивания злоумышленнику необходимо иметь доступ к ПК, расположенному на пути сетевого потока, который необходимо анализировать; например, к маршрутизатору или *PPP*-серверу на базе *UNIX*. Если злоумышленнику удастся получить достаточные права на этом ПК, то с помощью специального ПО сможет просматривать весь трафик, проходящий через заданные интерфейс.

Второй вариант – злоумышленник получает доступ к ПК, который расположен в одном сегменте сети с системой, которой имеет доступ к сетевому потоку. Например, в сети «тонкий *ethernet*» сетевая карта может быть переведена в режим, в котором она будет получать все пакеты, циркулирующие по сети, а не только адресованной ей конкретно. В данном случае злоумышленнику не требуется доступ к *UNIX* - достаточно иметь ПК с *DOS* или *Windows* (частая ситуация в университетских сетях).

Поскольку *TCP/IP*-трафик, как правило, не шифруется, злоумышленник, используя соответствующий инструментарий, может перехватывать *TCP/IP*-пакеты, например, *telnet*-сессий [60-62] и извлекать из них имена пользователей и их пароли.

Следует заметить, что данный тип атаки невозможно отследить, не обладая доступом к системе злоумышленника, поскольку сетевой поток не изменяется. Единственная надежная защита от подслушивания - шифрование *TCP/IP*-потока

(например, *secure shell*) или использование одноразовых паролей (например, *S/KEY*) [35, 52].

Естественно, подслушивание может быть и полезно. Так, данный метод используется большим количеством программ, помогающих администраторам в анализе работы сети (ее загруженности, работоспособности и т.д.). Один из ярких примеров – *tcpdump*.

Активные атаки. Злоумышленник взаимодействует с получателем информации, отправителем и/или промежуточными системами, возможно, модифицируя и/или фильтруя содержимое *TCP/IP*-пакетов.

Активные атаки можно разделить на две части:

- Злоумышленник предпринимает шаги для перехвата и модификации сетевого потока или попыток «притвориться» другой системой.
- *TCP/IP* используется для того, чтобы привести жертву в нерабочее состояние.

Обладая достаточными привилегиями в ОС, злоумышленник может вручную формировать *IP*-пакеты и передавать их по сети. Естественно, поля заголовка пакета могут быть сформированы произвольным образом. Получив такой пакет, невозможно выяснить, откуда реально он был получен, поскольку пакеты не содержат пути их прохождения. Конечно, при установке обратного адреса, не совпадающего с текущим *IP*-адресом, злоумышленник никогда не получит ответ на отосланный пакет.

Пассивное сканирование. При его использовании злоумышленник посылает *TCP/IP SYN*-пакет на все порты. Для *TCP*-портов, принимающих соединения извне, будет возвращен *SYN/ACK*-пакет. Остальные вернут *RST*-пакеты. Проанализировав данный ответ, злоумышленник может быстро понять, на каких портах работают программы. Метод не обнаруживается, поскольку реальное *TCP/IP*-соединение не устанавливается.

1.4 Организация защищенного канала связи на основе криптографических протоколов *SSL* и *TLS*

Протокол *SSL* (*secure socket layer*) [42-43] обеспечивает защиту данных между сервисными протоколами (*HTTP*, *NNTP*, *FTP*) [29] и *TCP/IP*. Использует как асимметричную, так и симметричную криптографию [8, 34, 36-38].

Протокол предоставляет безопасный канал, который имеет три свойства:

- Является частным - шифрование используется для всех сообщений после диалога, который служит для определения секретного ключа.
- Является аутентифицированным - серверная сторона диалога всегда аутентифицируется, в то время как клиентская - аутентифицируется опционно.
- Является надежным - транспортировка сообщений включает в себя проверку целостности.

Следует отметить, что *SSL* не только обеспечивает защиту данных в интернете, но также производит опознание сервера и клиента (*server/client authentication*).

Одним из критериев уровня защиты, является размер используемых ключей. Чем больше этот размер, тем соответственно надежнее защита. Браузеры в основном используют три размера: 40, 56 и 128 бит, соответственно [30, 31].

Работу протокола можно разделить на два уровня (слоя) [42], как показано на рисунке 1.3:

1. Подтверждения подключения (*Handshake Protocol Layer*).
2. Записи.

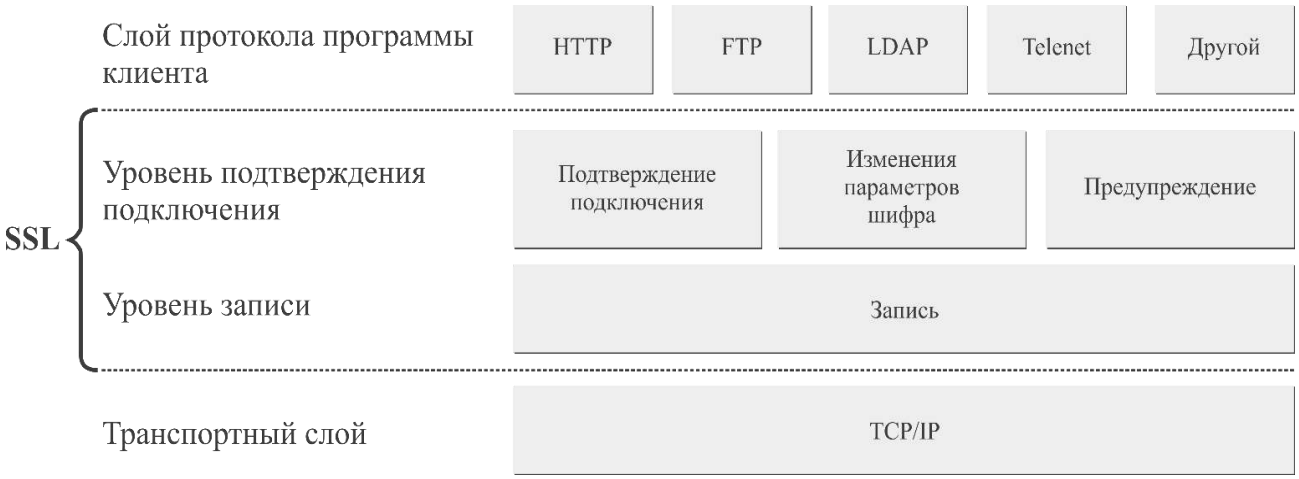

Рисунок 1.3 – Схема работы протокола *SSL*

Первый уровень, в свою очередь, состоит из трех подпротоколов: подтверждения подключения (*Handshake Protocol - HP*), изменения параметров шифра (*Cipher Spec Protocol - CSP*), предупредительного (*Alert Protocol - AP*).

HP используется для синхронизации (согласования) данных сессии между клиентом и сервером. К данным сессии относятся:

- Идентификационный номер сессии.
- Сертификаты обеих сторон.
- Параметры алгоритма шифрования.
- Алгоритм сжатия информации.
- «Общий секрет» применен для создания ключей, открытый ключ.

CSP используется для изменения данных ключа (*keyingmaterial*), состоит из одного сообщения, в котором сервер говорит, что отправитель хочет изменить набор ключей.

AP содержит сообщение, которое показывает сторонам изменение статуса или сообщает о возможной ошибке.

Уязвимости *SSL* [27, 28]:

- Идентичные криптографические ключи используются для аутентификации и шифрования сообщений.
- *SSL* 2.0 (развитие *SSL*) имеет слабую *MAC*-конструкцию (*Message Authentication Code*), которая использует *MD5* хэш-функцию с секретом

префикса.

- *SSL* 2.0 не имеет никакой защиты для *Handshake Protocol* (атаки типа *Man-in-the-Middle* могут остаться незамеченными).
- *SSL* 2.0 использует *TCP* закрытое соединение, чтобы указать конец данных (злоумышленник может подделывать *TCP FIN*, оставив получателя без сообщения о конце передачи данных).

SSL использует потоковое шифрование *RC4* [63] или блочное шифрование в режиме *CBC* [51]. Проблема *CBC*-шифрования в *SSL 3.0* [41] заключается в том, что дополнение блоков (*padding*) может быть произвольным (за исключением последнего байта), на него не распространяется *MAC*. Целостность дополнения не может быть полностью подтверждена в ходе дешифрования, поскольку в *SSL 3.0* сообщение сначала подписывается с помощью *MAC*, затем дополняется паддингом, и уже после шифруется блочным шифром.

Протокол *TLS* (*Transport Layer Security*) [22] используется для обеспечения конфиденциальности и целостности данных при коммуникации двух приложений. В результате:

- Соединение является конфиденциальным - для шифрования данных используется симметричная криптография, ключи генерируются независимо для каждого соединения и базируются на секретном коде, получаемом с помощью другого протокола.
- Соединение является надежным - процедура передачи сообщения включает в себя проверку целостности с помощью вычисления *MAC*, для расчета *MAC* используются хэш-функции.

Перед тем, как начать обмен данными через *TLS*, клиент и сервер должны согласовать параметры соединения: версию протокола, способ шифрования, проверить сертификаты (если необходимо). Схема процедуры соединения (*TLS Handshake*) показана на рисунке 1.4.

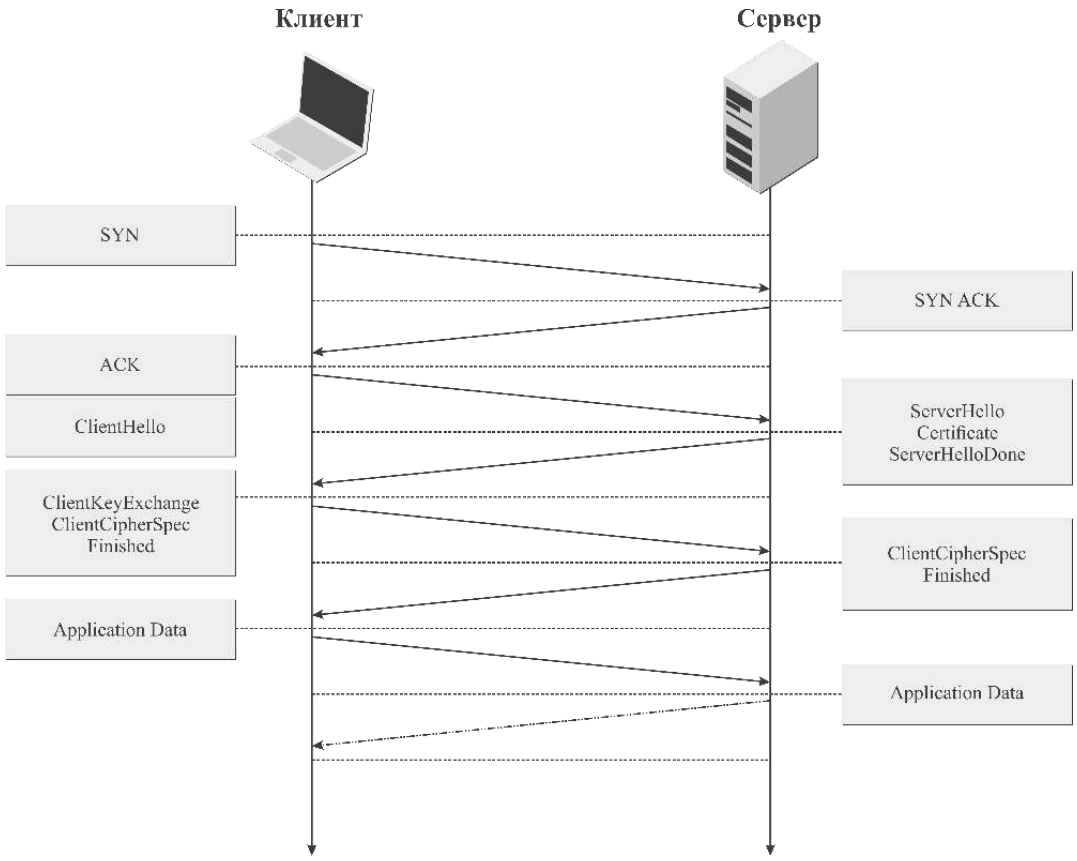

Рисунок 1.4 – Схема процедуры соединения *TLS*

Процедура соединения:

- Между клиентом и сервером устанавливается *TCP*-соединение.
- Клиент посылает на сервер спецификацию (версию протокола, которую хочет использовать, поддерживаемые методы шифрования и т.д.).
- Сервер утверждает версию используемого протокола, выбирает способ шифрования из предоставленного списка, прикрепляет свой сертификат и отправляет ответ клиенту.
- Клиент проверяет присланный сертификат и инициирует либо *RSA* [65-67], либо обмен ключами по Диффи-Хеллману [35].
- Сервер обрабатывает присланное клиентом сообщение, сверяет *MAC* и отправляет клиенту заключительное сообщение (*Finished*) в зашифрованном виде.
- Клиент расшифровывает полученное сообщение, сверяет *MAC* и если нет ошибок, то соединение считается установленным (начинается обмен

данными приложений).

Конец процедуры.

В *TLS* используется обмен ключами по алгоритму *RSA* [65-67]: клиент генерирует симметричный ключ, подписывает его с помощью открытого ключа сервера и отправляет его на сервер. В свою очередь, на сервере ключ клиента расшифровывается с помощью закрытого ключа. После этого обмен ключами объявляется завершённым.

Данный алгоритм имеет недостаток: эта же пара отрытого и закрытого ключей используется и для аутентификации сервера. Соответственно, если злоумышленник получает доступ к закрытому ключу сервера, он может расшифровать весь сеанс связи.

Обмен ключами Диффи-Хеллмана представляется более защищённым, так как установленный симметричный ключ никогда не покидает клиента или сервера и, соответственно, не может быть перехвачен злоумышленником, даже если тот знает закрытый ключ сервера.

В протоколе определены следующие криптографические операции: цифровая подпись, блочное шифрование и шифрование с открытым ключом.

Процедура записи фрагментирует сообщение на блоки нужной длины, осуществляет сжатие данных, вычисляет *MAC* и зашифровывает их. На другом конце соединения, полученные данные расшифровываются, проверяется их целостность, далее они декомпрессируются, дефрагментируются и передаются протоколам более высокого уровня. Выше протокола записи могут располагаться следующие протоколы: протокол рукопожатия, *alert*-протокол, протокол изменения шифрования и прикладной протокол, безопасность которого обеспечивается.

В *TLS* вводится понятие состояния соединения, которое определяет параметры выполнения протокола записи. Такими параметрами являются алгоритм сжатия, алгоритм шифрования и *MAC*-алгоритм [47], а также параметры этих алгоритмов, т.е. секреты *MAC*, ключи алгоритма шифрования и инициализационные вектора. Для каждого направления (соответственно чтение

или запись) параметры соединения могут различаться.

Существует четыре состояния соединения: текущие состояния чтения и записи и ожидаемые состояния чтения и записи. Параметры безопасности для ожидаемых состояний устанавливаются протоколом рукопожатия, а протокол изменения шифрования делает ожидаемое состояние текущим, в результате чего соответствующие параметры текущего состояния сбрасываются и заменяются параметрами ожидаемого состояния. Параметры ожидаемого состояния инициализируется пустыми значениями. Вначале текущее состояние всегда определяется без использования шифрования, сжатия и *MAC*.

1.5. Повышение эффективности криптографических протоколов

Основными направлениями повышения эффективности использования рассмотренных в предыдущем разделе протоколов защиты информации в КС сетей *TCP/IP* является повышение скорости работы (производительности алгоритмов, лежащих в их основе) и криптостойкости. Предлагаемые подходы рассмотрим на примере протокола *TLS*.

Общий граф работы *TLS* с уязвимыми местами представлен на рисунке 1.5.

Уязвимость, обозначенная цифрой 1. Из-за большой вариации используемых алгоритмов ассиметричной криптографии с разными длинами ключей возможно осуществление *MITM*-атаки (рисунок 1.6), отсюда, все дальнейшие передаваемые данные будут скомпрометированы и известны злоумышленнику [63, 64]. Для генерации сеансовых ключей (при аутентификации сервера и клиента) на этапе создания *VPN*-туннеля [54, 55], вместо медленного *RSA* (других ассиметричных алгоритмов) представляется возможным:

- Использовать симметричный вариант рюкзачной криптосистемы с *pre-shared key* [8] в виде общей памяти.
- Для генерации *Master Secret* использовать уникальную для пары (клиент, сервер) хеш-функцию на основе данной симметричной рюкзачной криптосистемы.

Выдвинем гипотезу, что данное решение поможет избежать *MITM*-атак и атак свойственным ассиметричным алгоритмам, использующихся в стандартных криптонаборах *TLS*, а также повысить скорость работы процесса аутентификации клиента и сервера.

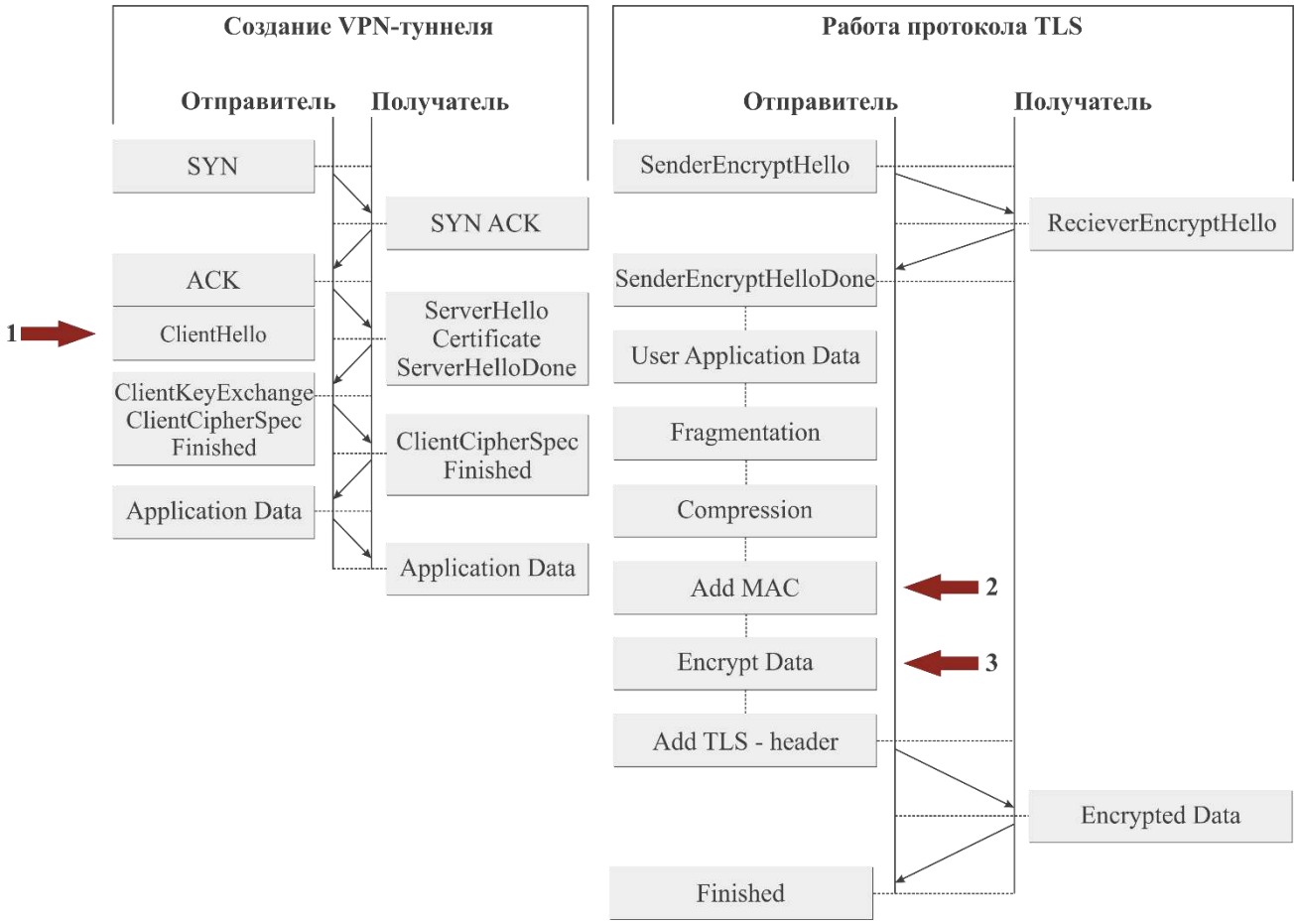

Рисунок 1.5 – Общий граф работы *TLS* с потенциально уязвимыми местами

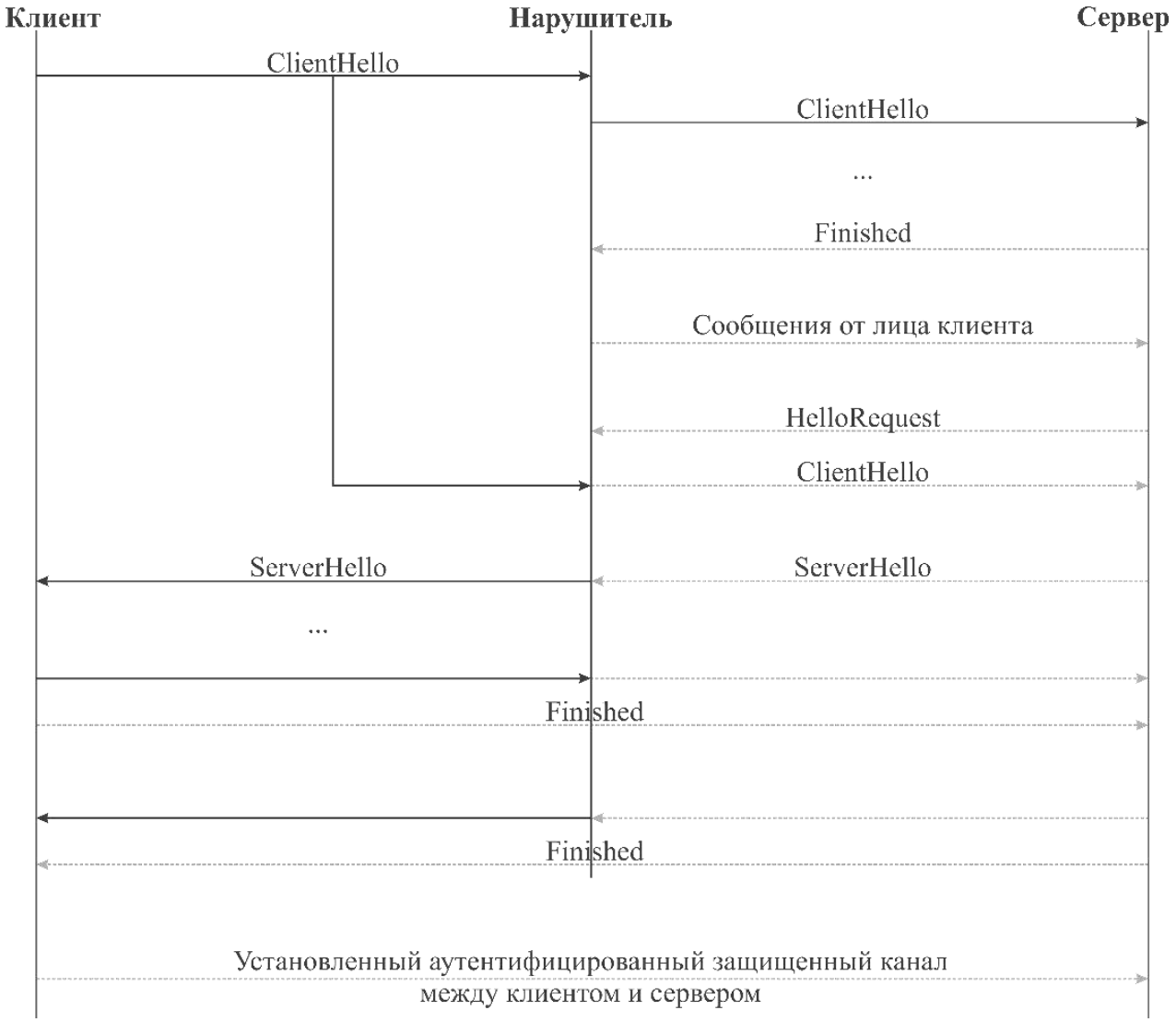

Рисунок 1.6 – Пример *MITM*-атаки на этапе обмена сообщениями *hello*

Уязвимость, обозначенная цифрой 2. В *TLS* - *MAC*, в той или иной форме содержится во всех зашифрованных записях. Способ вычисления кода аутентификации зависит от используемого шифронабора и режима шифрования. *HMAC* [47] - *MAC*, основанный на криптографической хеш-функции используется, например, с шифрами *TLS* в режиме *CBC*. *HMAC* вычисляется независимо от функции шифрования. Более того, *MAC* изначально используется неверно: сперва для открытого текста вычисляется *HMAC*, а потом сообщение, с присоединённым кодом аутентификации, шифруется. Именно так работает режим *CBC* в *TLS*. Внутри записи защищённые данные размещаются вместе с кодом аутентификации, и то, и другое - зашифровано. Архитектурная проблема состоит в том, что *MAC*

можно проверить только после расшифрования сообщения и, если значение оказалось некорректным, реализация *TLS* должна сообщить об ошибке, из-за этого активный атакующий, манипулируя шифрованным текстом, может использовать расшифровывающий узел в качестве криптографического оракула, который постепенно раскроет весь секретный текст. На этом основано несколько практических атак на *TLS*. Для обхода этой проблемы на основе реализации рюкзачного симметричного блочного шифра с помощью стандартной схемы свертки блоков за счет общей памяти можно предложить использовать построения параметризованного по наличию общей памяти стандартного алгоритма хеширования, применяемого для контроля достоверности передаваемой информации от отправителя к получателю. Свертка блочного шифра порождает такую хэш-функцию для пары (*Sender*, *Receiver*), которая является строго индивидуальной и будет лишена вышеописанных недостатков.

Уязвимость, обозначенная цифрой 3. В дополнение к применению в основе шифронабора *TLS* стандартных алгоритмов (*RC2*, *RC4*, *AES*, *DES*, *3DES* и т.п.) [73] предлагается использовать алгоритмы преобразования информации с помощью *CBC*-блочного варианта симметричной рюкзачной криптосистемы с общей памятью, которая сама по сути является аналогом *pre-shared key*. Пусть под скоростью работы протокола *TLS* понимается величина (1.1).

$$IPS(TLS) = \frac{1}{T_{auth} + T_{frag} + T_{cmps} + T_{MAC} + T_{enc}}, \quad (1.1)$$

где T_{auth} - время работы алгоритма аутентификации клиента и сервера; T_{frag} – время работы алгоритма фрагментации сообщений; T_{cmps} - время работы алгоритма компрессии сообщений; T_{MAC} - время работы алгоритма создания кода аутентификации сообщений и T_{enc} – время работы симметричных блочных алгоритмов шифрования и дешифрования сообщений.

Предполагается, что разрабатываемые алгоритмы выигрывают по скорости работы (симметричные рюкзачные шифры одни из самых быстрых на данный

момент), также по криптостойкости, так как обладают криптостойкостью к атакам, которым подвержены шифры из стандартного шифронабора, а также умеют работать в различных режимах.

Внедряя предложенную криптосистему в такие фазы работы протокола *TLS*, как фаза аутентификации клиента и сервера, фаза создания кода аутентификации сообщений и фаза работы симметричных блочных алгоритмов шифрования и дешифрования сообщений, мы тем самым существенно уменьшаем величины T_{auth}, T_{MAC} и T_{enc} из-за чего в соответствии с (1.1) увеличивается значение $IPS(TLS)$, то есть повышается производительность протокола *TLS* и его эффективность.

Сравнение систем организации защищенного канала связи в сетях *TCP/IP* представлено в таблице 1.1.

Рассмотренные протоколы безопасности (*SSL, TLS*) обеспечивают существенную криптозащиту в КС, тем не менее, имеют множество критичных недостатков, которые свойственны криптоалгоритмам, лежащим в их основе. Наиболее перспективными на данный момент средствами, которые включают в себя все преимущества вышеописанных протоколов, но в тоже время лишены всех их недостатков являются механизмы симметричной криптографии, конкретно, алгоритмы и протоколы симметричных рюкзачных криптосистем [1, 8, 10], которые обладают массой преимуществ по сравнению с другими алгоритмами: высокая скорость работы, простота реализации, минимальное потребление ресурсов пользователя, отличная масштабируемость и т.п.

Таблица 1.1 – Сравнение систем организации защищенного КС

Характеристики	Стандартный *SSL*	Стандартный *TLS*	*Cisco Easy VPN*	*TLS* на основе «рюкзака с общей памятью»
Скорость работы алгоритма	средняя	средняя	средняя	высокая
Масштабируемость	высокая	высокая	низкая	высокая
Сложность реализации	средняя	средняя	высокая	средняя
L^3-атака	+	+	+	-
MITM-атака	+	+	+	-
Зависимость стойкости алгоритма от длины ключа	высокая	высокая	высокая	средняя
Потребление ресурсов канала и пользователя	низкое	низкое	среднее	низкое
Криптостойкость по отношению к типовым атакам (перебор, подслушивание и т.п.)	средняя	средняя	средняя	высокая
NIST-тестирование	+	+	+	+
Требования к лицензированию	-	-	+	-

В нашей модификации по сравнению с существующими версиями рюкзачных криптографических систем, новыми являются следующие две идеи:

- Предполагается наличие общей памяти для пары (*Sender*, *Receiver*), однако недоступной противнику в КС. Обозначим это числовое множество как $O = \{o_1, o_2, \ldots o_B\}$. На основе общей памяти с использованием двоичного вектора $E = (e_1 \ldots e_B)$ длины B определяем выбранный параметр Ψ_0 из всех возможных сумм $\{\Psi\}$ произведений неповторяющихся элементов общей памяти и элементов вектора E $(1 \ldots 2^B-1)$, который участвует в определении базиса задачи об укладке рюкзака [7].
- Отказ от супервозрастающих базисов в пользу линейно - рекуррентных последовательностей порядка m, где $m \geq 2$. Весьма частным случаем таких последовательностей для $m = 2$ является последовательность Фибоначчи [1].

Исходя из вышесказанного, задача диссертационного исследования заключается в повышении эффективности (производительности и криптостойкости) защищенного КС сетей *TCP/IP* на базе протокола *TLS* с помощью встраивания в основные процедуры протокола алгоритмов симметричных рюкзачных криптосистем.

Для ее достижения необходимо решить следующие частные задачи:

- Разработать подход к организации симметричных рюкзачных криптосистем в защищенных КС сетей *TCP/IP*.
- Разработать семейство моделей и алгоритмов повышения производительности и криптостойкости симметричных рюкзачных криптосистем в защищенных КС сетей *TCP/IP*.
- Разработать ПО для экспериментального исследования предложенных моделей и алгоритмов в практических задачах.
- Предложить структурную схему защищенного КС на базе симметричной рюкзачной криптосистемы.

Выводы к главе 1

1. Каналы связи сетей *TCP/IP* не имеют встроенных средств защиты передаваемых сообщений. Механизмы обеспечения информационной безопасности реализованы на сеансовом уровне сетевой модели *OSI* и имеют множество уязвимостей. Основные угрозы и атаки в каналах связи направлены на перехват и имперсонацию сообщений – нарушение конфиденциальности и целостности передаваемых данных. Незначительное количество атак относится к атакам на доступность узлов канала и их подмену.

2. Наиболее перспективным подходом в обеспечении безопасности передаваемых сообщений является использование криптостойкого шифрования *TCP/IP*-потока. Шифрование в каждом канале связи не позволяет злоумышленнику анализировать служебную информацию и уменьшает вероятность доступа к незашифрованным данным в узлах сети. При этом злоумышленник может

проводить анализ только открыто передаваемых данных, но не может нелегально использовать канал связи.

3. Протоколы *SSL* и *TLS* обеспечивают существенную криптозащиту канала связи, тем не менее, имеют недостаточную производительность и относительно низкую криптостойкость. Все версии *SSL* имеют значительное количество уязвимостей, что обуславливает необходимость постоянной установки «заплаток».

4. Для повышения производительности и криптостойкости защищенного канала связи перспективным является использование средств симметричной рюкзачной криптографии.

2 РАЗРАБОТКА МОДЕЛЕЙ И АЛГОРИТМОВ ОРГАНИЗАЦИИ ЗАЩИЩЕННОГО КАНАЛА СВЯЗИ В СЕТЯХ *TCP/IP* НА БАЗЕ СИММЕТРИЧНОЙ РЮКЗАЧНОЙ КРИПТОСИСТЕМЫ

В данной главе приводится описание алгоритмов шифрования, дешифрования, хеширования и передачи сообщений от отправителя (*sender*) к получателю (*receiver*) по каналу связи, даны блок-схемы их работы. Описывается вариация актуальной на сегодняшний день системы организации защищенного канала связи в сетях *TCP/IP* с помощью *CBC*-блочной симметричной рюкзачной криптосистемы с общей памятью, которая обладает высокой скоростью работы и заданным уровнем криптостойкости.

2.1 Симметричная рюкзачная криптографическая система

Задача о рюкзаке в криптографии [2] — это задача, на основе которой Ральф Меркл и Мартин Хеллман разработали первый алгоритм шифрования с открытым ключом (криптосистема Меркла-Хеллмана - КМХ [34, 53]). Для шифрования сообщений использовалось решение задачи о рюкзаке. Считалось, что КМХ может обеспечивать необходимую криптостойкость.

Задача о рюкзаке относится к классу *NP*-полных [46], поэтому при ее решении нужно выбирать между точными алгоритмами, которые не применимы для «больших» рюкзаков и приближенными, которые работают быстро, но не обеспечивают оптимального решения.

Характеристикой криптостойкости является плотность укладки рюкзака. Большинство неудач известных алгоритмов вызвано низкими значениями данного коэффициента, в результате чего, в соответствии с теорией Костера и Одлыжко [32, 50, 57] появляется возможность проведения L^3-атаки. При успешном осуществлении данной атаки злоумышленник имеет возможность скомпрометировать защищаемую информацию, нарушив ее конфиденциальность, целостность или доступность.

В 1980 году был предложен способ повысить криптостойкость КМХ [3, 8], тем не менее, уже в 1983 году схема была взломана [69]. Однако, поиски модификаций КМХ, лишённых выявленных недостатков, продолжились. В 1985 году была создана схема, основанная на модульных рюкзаках с секретной лазейкой с использованием китайской теоремы об остатках, в 1986 году - на основе алгебраической теории кодирования, в 1988 году - с использованием мультипликативного рюкзака, в 2001 году - на основе мультипликативных рюкзаков с использованием алгоритма *Schalkwijk* [34, 53].

Продолжались и улучшения классического КМХ. Отметим подход на основе диофантовых уравнений и концепцию, где отправитель может выбрать ключ шифрования из набора ключей, алгоритм *Chor-Rivest*, который на сегодня считается безопасным [14, 23, 24, 53].

Наиболее известные из «нерюкзачных» алгоритмов: *RSA*, *EIGamal* и *Rabin* [25]. Однако они медленны и не подходят для быстрой шифровки и дешифровки больших объёмов сообщений. Решением проблемы являются гибридные криптосистемы, где шифрование сообщений осуществляется быстрым симметричным алгоритмом со случайным ключом, а алгоритм на открытых ключах применяется для шифрования самого случайного (сеансового) ключа.

Симметричная криптографическая система:

Симметричная криптосистема - способ шифрования, в котором для шифрования и дешифрования применяется один и тот же криптографический ключ. Алгоритм шифрования выбирается сторонами до начала обмена сообщениями.

Данные криптосистемы характеризуются наиболее высокой скоростью шифрования и с их помощью обеспечиваются как конфиденциальность и подлинность, так и целостность передаваемой информации. Конфиденциальность передачи информации с помощью симметричной криптосистемы зависит от надежности (криптостойкости) шифра и обеспечения конфиденциальности ключа шифрования. Структура типовой симметричной криптосистемы представлена на рисунке 2.1.

Здесь M – исходное сообщение (отрытый текст); k – криптографический (закрытый) ключ – секретная информация, используемая криптографическим алгоритмом при шифровании / дешифровании сообщений; S – передаваемое (зашифрованное) сообщение; F и F^{-1} – алгоритмы шифрования и дешифрования сообщений соответственно.

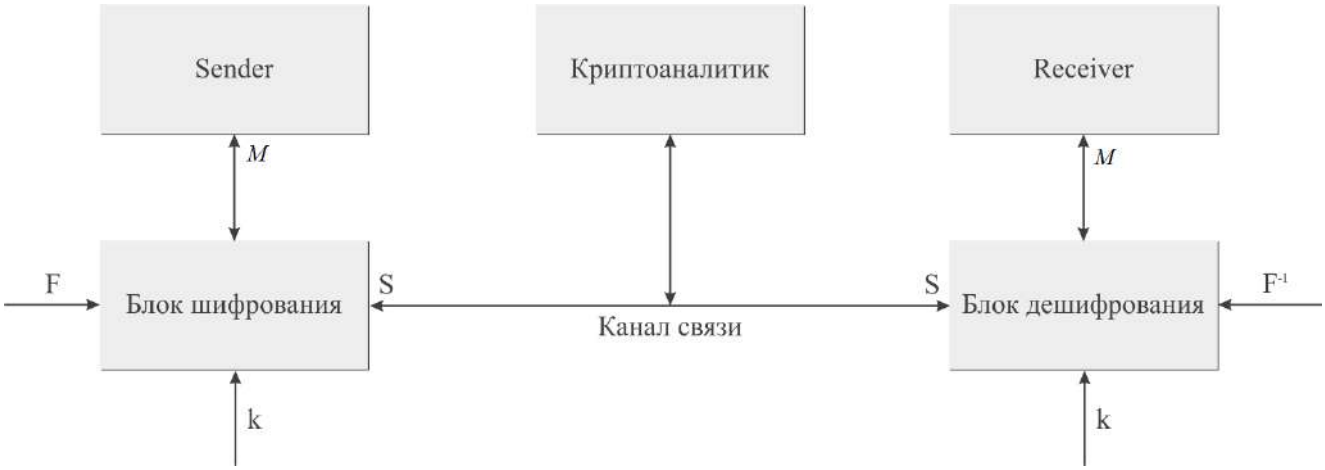

Рисунок 2.1 – Типовая схема симметричной криптосистемы

Само сообщение, которое отправитель (*Sender*) хочет послать получателю (*Receiver*) через небезопасный канал связи (КС) называется открытым текстом. Шифрованное сообщение называется закрытым текстом. Процесс преобразования закрытого текста в открытый называется дешифрованием. Криптоаналитиками называются те, кто используют криптоанализ – методы «взлома» закрытого текста.

Считается [39], что криптоаналитик обладает следующими возможностями:

- может получить любое сообщение, передаваемое по КС;
- может стать стороной, принимающей сообщения от любой передающей стороны;
- может посылать любому пользователю сообщения от имени любого другого пользователя.

Ограничения криптоаналитика:

- не может угадывать случайные числа, выбранные из достаточно большого множества;
- не может расшифровать, не имея ключа, либо корректно зашифровать сообщение при условии использования некоторого алгоритма шифрования;
- не может найти закрытый ключ по открытому ключу;
- контролируя средства связи, не может получить доступ к закрытым, внутренним ресурсам, например, к памяти или жёсткому диску пользователя, а также к закрытому КС.

Существует два типа симметричных алгоритмов, основанных на ключах: с закрытым и открытым ключом. Алгоритмы с открытым ключом разработаны таким образом, что ключ используемый для шифрования отличается от ключа, используемого для дешифрования. Ключ дешифрования не может быть рассчитан по ключу шифрования. В случае с закрытым ключом – для шифрования и дешифрования используется один и тот же ключ. В нашем случае мы рассматриваем только симметричные криптосистемы с закрытым ключом, так как при введении и использовании открытого ключа значительно снижается скорость работы алгоритмов шифрования и дешифрования и существует атака Шамира [14] на пару связанных между собой ключей – открытого и закрытого, которая компрометирует такую симметричную криптосистему.

Для повышения криптостойки симметричных криптосистем применяется режим сцепления блоков шифротекста (*Cipher Block Chaining, CBC*) [23, 24].

Каждый блок открытого текста (кроме первого) побитово складывается по модулю 2 (операция *XOR*) с предыдущим результатом шифрования (рисунок 2.2 и рисунок 2.3).

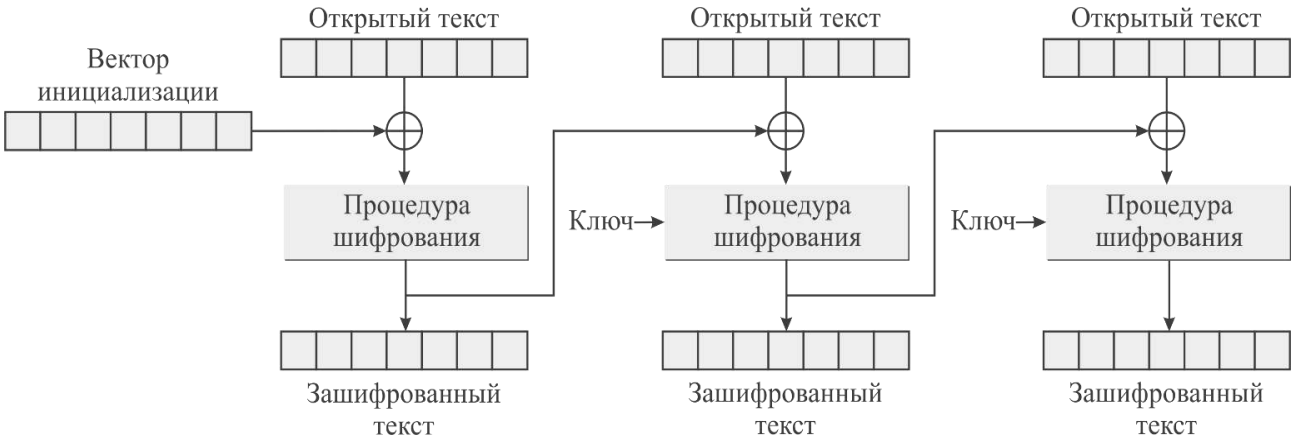

Рисунок 2.2 – Схема работы алгоритма шифрования в режиме *CBC*

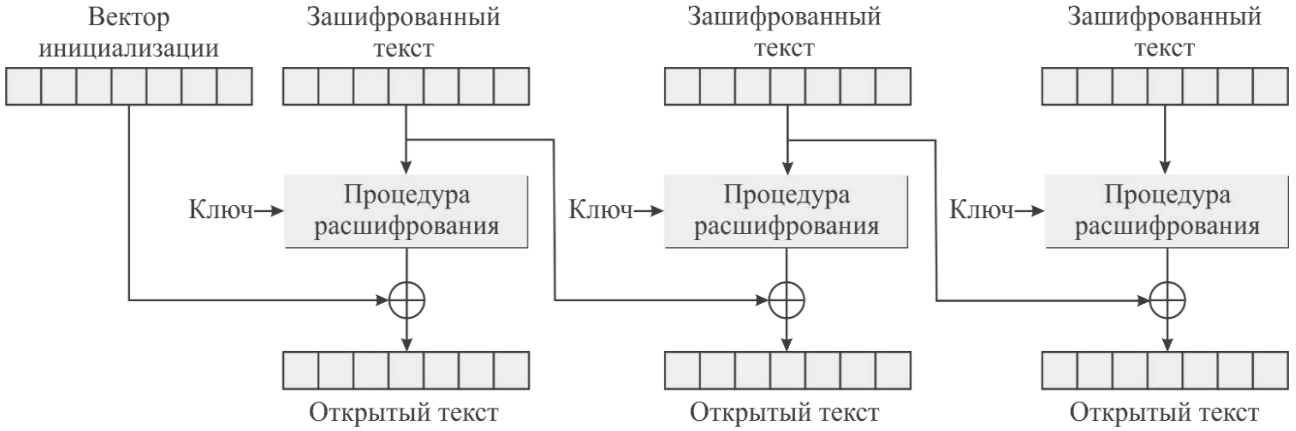

Рисунок 2.3 – Схема работы алгоритма дешифрования в режиме *CBC*

Симметричная рюкзачная криптографическая система:

Задача о рюкзаке в криптографии - это задача, на основе которой Меркл и Хеллман [2] разработали первый алгоритм шифрования, где для шифрования сообщений использовалось решение задачи о рюкзаке, как известно являющейся *NP*-сложной [46]. Считалось, что она (криптосистема) могла обеспечивать достаточную криптостойкость. На данный момент создано множество рюкзачных криптосистем, например [34-36]. Однако практически все они на сегодняшний день взломаны или признаны потенциально не безопасными.

Рассмотрим классическую постановку задачи. Задача о рюкзаке звучит так: задан набор $A = (a_1, ..., a_y)$ из y различных положительных целых чисел. Пусть есть число z целое и положительное. Задачей является нахождение такого набора a_i,

чтобы в сумме они давали ровно *z*. Ясно, что решение этой задачи существует не всегда.

Рюкзачный вектор $A = (a_1, ..., a_y)$ - упорядоченный набор из *y* предметов. Для шифрования исходное сообщение (открытый текст) *M* разбивают на блоки.

Идея стандартного алгоритма рюкзачной криптосистемы заключается в том, что сообщение шифруется как решения набора задачи о ранце. Предметы выбираются с помощью блока открытого текста, длина блока равна количеству предметов. Биты открытого текста соответствую ценности предметов.

Шифровать можно двумя способами:

1. *S* получается возведением элементов *A* в степень соответствующих им битов *M* и дальнейшим перемножением результатов. Такой способ называют мультипликативным.

2. Блоки *S* получаются путём умножения элементов *A* на соответствующие им значения разрядов блоков *M* и дальнейшим сложением результатов. Такой способ называют аддитивным.

Алгоритм получения стандартного шифра для рюкзачного вектора *A* = (3 4 6 7 10 11) с длинной *y* = 6 (аддитивный способ) приведен в таблице 2.1. Открытый текст «111110001100000000000001», а шифротекст «3013011».

Таблица 2.1 - Пример получения шифротекста

Открытый текст	Вещи в рюкзаке	Шифротекст
1 1 1 1 1 0	3 4 6 7 10 11	3 + 4 + 6 + 7 + 10 = 30
0 0 1 1 0 0	3 4 6 7 10 11	6 + 7 = 13
0 0 0 0 0 0	3 4 6 7 10 11	0
0 0 0 0 0 1	3 4 6 7 10 11	11

Для заданного *A* все шифротексты есть числа, не превышающие 41 - суммарный вес всех предметов в рюкзачном векторе. Для каждого исходного текста существует единственный шифротекст.

Применимость рюкзачных криптосистем основывается на том, что существуют две проблемы рюкзака: «лёгкая» и «трудная».

«Лёгкая» проблема заключается в том, что для некоторых рюкзачных векторов задача о рюкзаке легко решаема, но использование рюкзачных векторов (базисов) малой размерности и супервозрастающих базисов позволяет легко взламывать закрытое сообщение за линейное время [32, 50, 57].

«Трудная» проблема - открытый ключ, так как его легко применять для шифрования и невозможно для дешифровки сообщения. Меркл и Хеллман [53], используя модульную арифметику, разработали способ трансформации «лёгкого» рюкзака в «трудный».

Супервозрастающая последовательность [48, 49] – это последовательность, для которой всегда выполняется условие: для любого i:

$$f_{i+1} \geq \sum_{i=1}^{y} f_i, \tag{2.1}$$

где y – количество элементов рюкзачного вектора.

Минимальная супервозрастающая последовательность – это такая супервозрастающая последовательность, для которой выполняется условие: для любого i:

$$f_{i+1} = \left(\sum_{i=1}^{y} f_i\right) + 1. \tag{2.2}$$

Примером такой последовательности является последовательность степеней двойки $\{1, 2^1, 2^2, \ldots, 2^y\}$.

Заметим, что оценка скорости шифрования S осуществляется через логарифмическую функцию. Для последовательности степеней двойки она, например, будет равна $\sim log\ (2^y)$.

Еще одна «трудная проблема» - рюкзак не со сверхвозрастающим рюкзачным вектором. Единственный способ решения такой задачи - это проверка всех

возможных решений, пока не найдётся правильное. Самый быстрый алгоритм имеет экспоненциальную зависимость от размера рюкзачного вектора.

Безопасность рюкзачных криптосистем:

Реальная рюкзачная криптосистема должна содержать большое число элементов. Вскрытие такой криптосистемы при помощи грубой силы, т.е. перебором, будет трудной (невозможной) задачей. Рюкзачные криптосистемы не являются безопасными.

Важной характеристикой любой рюкзачной криптосистемы является плотность укладки (рюкзака), которая служит мерой криптостойкости:

$$\rho(a) = \frac{y}{max_{1 \le i \le y} log_2 a_y}, \quad (2.3)$$

где y – количество элементов возрастающего базиса (возрастающей последовательности); a_y – максимальный из всех элементов возрастающего базиса (возрастающей последовательности).

Большинство неудач предыдущих алгоритмов (причина отсутствия доверия к данному типу криптосистем) вызвано низкими значениями $\rho(a)$, в результате чего в соответствии с теорией Костера и Одлыжко [32, 50, 56, 57] появляется возможность проведения L^3-атаки. При успешном осуществлении данной атаки злоумышленник имеет возможность скомпрометировать защищаемую информацию, нарушив ее конфиденциальность, целостность или доступность.

Идея L^3-атаки состоит в том, чтобы преобразовать параметры задачи о рюкзаке в базис для некоторой целочисленной решетки Λ. Затем найти в этом базисе короткий рюкзачный вектор с помощью L^3-алгоритма редукции базиса решетки. Существует вероятность того, что с помощью найденного короткого рюкзачного вектора можно обратно преобразовать решение задачи рюкзачной криптосистемы.

В работах [32, 50, 57] было показано, что проведение L^3-атаки возможно на криптографические рюкзачные криптосистемы с $\rho(a) < 0.9408$, и скорее всего, возможно для всех криптографических рюкзачных криптосистем с $\rho(a) < 1$.

Множество методов и способов увеличения криптостойкости систем заключаются в увеличении вычислительной сложности задач, лежащих в основе алгоритмов шифрования и дешифрования. В случае с рюкзачными криптосистемами простое увеличение вычислительной сложности задач, лежащих в основе алгоритмов шифрования и дешифрования, будет недостаточным.

L^3-атака первоначально нацелена на последовательность (рюкзачный вектор), которая является супервозрастающей и процесс ее формирования. По этой причине особую важность в разработке надежного способа организации защищенного канала связи в сетях *TCP/IP* играет принципиально новая вариация рюкзачной криптосистемы, в которой особую роль занимают плотность укладки и общая память.

Таким образом, можно выделить основные проблемы рюкзачных криптосистем, которые существенным образом влияют на их криптостойкость и скорость работы:

- использование рюкзачных векторов (базисов) малой размерности, для которых задача о рюкзаке легко решаема даже вручную из-за очень низкого показателя плотности укладки;
- использование в качестве рюкзачных векторов (базисов) сверхрастущих векторов (супервозрастающих последовательностей / базисов), для которых возможно проведение L^3-атаки, следствием которой является возможность компрометации защищаемой информации, нарушения ее конфиденциальности и целостности;
- низкая скорость работы из-за применения сверхрастущих рюкзачных векторов (супервозрастающих последовательностей / базисов) в попытке повышения криптостойкости рюкзачных криптосистем, чтобы единственным способом решения такой задачи стал проверка всех возможных вариантов укладки, пока не найдётся правильное решение -

самый быстрый такой алгоритм имеет экспоненциальную зависимость от размера рюкзачного вектора, что очень медленно;

- атака Шамира на пару связанных между собой ключей – открытого и закрытого.

Для решения этих проблем предлагается использовать следующие нововведения в рюкзачных криптосистемах:

- введение и использование общей памяти между отправителем и получателем сообщений в КС сетей *TCP/IP*;
- отказ от базисов малой размерности и супервозрастающих базисов в пользу линейно-рекуррентных последовательностей порядка m, где $m \geq 2$;
- повышение показателя плотности укладки до $\rho(a) \sim 1$, при котором затруднено проведение L^3-атаки.

До сих пор [26], основное внимание криптографов сфокусировано на криптосистемах, базирующихся на целочисленной факторизации, дискретном логарифме и эллиптических кривых. Однако исследования рюкзачных криптосистем продолжается, но такие криптосистемы не популярны и не вызывают доверия, учитывая неудачи предыдущих алгоритмов. Если будет создан алгоритм, полностью использующий всю трудность задачи о ранце (NP-полноту), с высокой плотностью и с трудными для открытия секретными лазейками, то это будет система лучше, чем на основе целочисленной факторизации и дискретного логарифмирования (их NP полнота не доказана). Кроме того, данная система будет очень быстрой.

Для рюкзачной криптосистемы с размером рюкзачного вектора $y = 100$ одна итерация алгоритма шифрования или дешифрования может быть более чем в 100 раз быстрее *RSA* (с модулем в 500 бит), что говорит о безупречной скорости работы данного типа криптосистем.

2.2 Математические модели рюкзачной системы защиты КС. Модифицированная структура данных *TLS*

Параметры моделей: M – исходное сообщение (отрытый текст); $M = M_1 \,||\, \dots \,||\, M_n \,||\, M_N$, где N – количество блоков в открытом сообщении; $M_n = m_{n1} \,||\, \dots \,||\, m_{ng} || \, m_{nG}$ - побитовое представление каждого блока открытого сообщения, G – количество бит в одном блоке и $m_{ng} \in \{0,1\}$; k – криптографический ключ; S – передаваемое (зашифрованное) сообщение; $S = S_1 \,||\, \dots \,||\, S_n \,||\, S_N$, где N – количество блоков в закрытом сообщении; $S_n = s_{n1} \,||\, \dots \,||\, s_{ng} || s_{nG}$ - побитовое представление каждого блока закрытого сообщения, $s_{ng} \in \{0,1\}$; $F_k(M)$ и $F_k^{-1}(S)$ – алгоритмы прямого и обратного преобразования информации на ключе k; O – о б щ а я п а м я т ь между отправителем и получателем; $O = \{o_1, \dots, o_b, o_B\}$, где B – количество документов общей памяти, || - операция конкатенации.

Модель 1. Прямое преобразование $M \xrightarrow{F_k(M)} S$: $S_n = \left(\sum_{n=1}^{G} m_g * k_g(\Psi_0)\right)$; где $k_1(\Psi_0) = 1$, $k_2(\Psi_0) = 1 \oplus \Psi_0$, …, $k_g(\Psi_0) = k_{g-1}(\Psi_0) \oplus k_{g-2}(\Psi_0)$, для $g>2$, до тех пор, пока таких элементов $k_g(\Psi_0)$ не будет ровно G, $\oplus$ - операция сложения в выбранном поле Галуа GF_p; Ψ_0 - серединный элемент отсортированной последовательности $\{\Psi\}$ не повторяющихся произведений элементов вектора O на элементы вектора $E = \{e_1, \dots, e_b, e_B\}$, $e_b \in \{0,1\}$, генерируемых случайным образом, вычисляемый в соответствии с алгоритмом, записанным псевдокодом по рекомендации *IETF*:

```
init vector_O [1…B] // вектор O
init vector_E [1…B] // вектор E
init vector_K [1…G] // вектор K - криптографический ключ
init temp_vector_ Ψ [1…B*B] // временный вектор произведений Ψ
init vector_ Ψ [1…B*B] // вектор произведений Ψ

for (i=1; i <=B; i++) // вычисление всех возможных произведений O и E
        for (j=1; j <=B; j++)
                t_Ψ(j) = O(i) * E(j)

for (i=1; i <=B*B; i++) // удаление одинаковых элементов из вектора Ψ
        for (j=1; j <=B*B; j++)
                if (t_Ψ(i) !=  t_Ψ(j) )
                Ψ(j) = t_Ψ(j)

for (i=1; i <=B*B; i++) // сортировка вектора Ψ по возрастанию
        for (j= B*B; j > i; j--)
                if (Ψ(j − 1) >  Ψ(j))
                temp = Ψ(j − 1)
                Ψ(j − 1) = Ψ(j)
                Ψ(j) = temp

Ψ0= Ψ size(Ψ) / 2 // выбор срединного элемента
G = size(Ψ);
K(1) = 1;

for (Ψ −2; Ψ <–G; Ψ ++) // формирование вектора K
        k(Ψ) = k(Ψ − 1) + k(Ψ − 2)

for (i=1; i <=G; i++) // прямое преобразование в КС для одного блока текста
        s(i) = m(i) * k(i)
```

Модель 2. Обратное преобразование $S \xrightarrow{F^{-1}(S)} M$: $M_n = m_{n1} || \dots || m_{ng} || m_{nG}$ - побитовое представление каждого блока открытого сообщения; $m_{ng} = s_{ng}$ - $k_g(\Psi_0)$, $m_{n(g-1)} = s_{n(g-1)}$ - $k_{g-1}(\Psi_0)$, …, $m_{n1} = s_{n1}$ - $k_1(\Psi_0)$ – дешифрование с помощью «жадного алгоритма»; если $s_{ng} < k_g(\Psi_0)$ – то $k_g(\Psi_0) = k_{g-1}(\Psi_0)$. Ψ_0 определяется аналогично прямому преобразованию.

2.3 Алгоритмы передачи и приема сообщений в КС

Алгоритм передачи сообщений:

Шаг 1. Установить *TCP*-соединение по КС, задав *IP*-адрес и порт узла-получателя.

Шаг 2. Провести взаимную аутентификацию узла-отправителя и узла-получателя с помощью протокола Диффи – Хеллмана из стандартного шифронабора протокола безопасности *TLS*. При успешной аутентификации перейти к шагу 3, иначе конец алгоритма (закрыть соединение).

Шаг 3. Выполнить прямое преобразование сообщения $M \xrightarrow{F_k(M)} S$ в соответствии с математической моделью 1 рюкзачной системы защиты.

Шаг 4. Передать $E = \{e_1, \dots, e_b, e_B\}$ и $S = S_1 \,||\, \dots \,||\, S_n \,||\, S_N$ аутентифицированному узлу-получателю по установленному КС.

Шаг 5. Дождаться ответа от узла-получателя. Если ответа не поступило перейти к шагу 4. При повторном отсутствии ответа от узла-получателя закрыть установленное соединение. Конец алгоритма.

Алгоритм приема сообщений:

Шаг 1. Провести взаимную аутентификацию узла-отправителя и узла-получателя с помощью протокола Диффи – Хеллмана из стандартного шифронабора протокола безопасности *TLS*. При успешной аутентификации перейти к шагу 2, иначе конец алгоритма (закрыть соединение).

Шаг 2. Принять $E = \{e_1, \dots, e_b, e_B\}$ и $S = S_1 \,||\, \dots \,||\, S_n \,||\, S_N$ от узла-отправителя по установленному КС.

Шаг 3. Ответить узлу-отправителю об успешном принятии пересланных данных и перейти к шагу 4, иначе сообщить об ошибке и закрыть соединение.

Шаг 4. Выполнить обратное преобразование сообщения $M \xrightarrow{F_k(M)} S$ в соответствии с математической моделью 2 рюкзачной системы защиты КС. Конец алгоритма.

2.4 Алгоритмы шифрования и дешифрования сообщений в КС сетей *TCP/IP* при помощи симметричной рюкзачной криптосистемы с общей памятью

Обозначим через $O = \{o_1, \ldots, o_b, o_B\}$, где B – количество документов общей памяти, исходную согласованную совокупность документов, имеющихся и у отправителя, и у получателя. Назовем это множество общей памятью. Наличие общей памяти далее будет использовано для расширения ключевого пространства алгоритма шифрования и создания базисов (рюкзачных векторов) типа Фибоначчи.

На начальном этапе проектирования алгоритмов рюкзачной криптосистемы имеется открытое сообщение M, которое необходимо передать между отправителем и получателем в КС сетей *TCP/IP*; стандартный шифронабор, используемый в *TLS* (*RSA_AES, DH_AES, RSA_3DES, DH_3DES* и т.д.) и ключ шифрования k на этих шифронаборах. Требуется доработать этот шифронабор, внедрив в него собственные алгоритмы (шифрования, дешифрования и хеширования) с высокой скоростью работы и заданным уровнем криптостойкости.

Исходное открытое сообщение делится на блоки $M_1 \,||\, \ldots \,||\, M_n \,||\, M_N$, где N – количество блоков в открытом сообщении. В зависимости от аппаратной реализации технических средств, на которых будут выполняться алгоритмы шифрования и дешифрования, а также от особенностей их программного оснащения зафиксируем длину каждого блока равной G. В стандартном исполнении алгоритмов шифрования и дешифрования с помощью симметричной блочной рюкзачной системы с общей памятью величина G = 64 бита. При необходимости последний блок открытого текста дополняется до заданной величины незначащими символами, например «0».

Стандартный шифронабор в *TLS* определен как следующие функции:

```
enum { null, rc4, rc2, des, 3des, des40, aes } BulkCipherAlgorithm;
```

После модификации протокола *TLS* с помощью симметричной рюкзачной криптосистемы с общей памятью, в список шифронабора войдет новая функция *skc_sh – symmetric knapsack cryptosystem with the shared memory*.

```
enum { null, rc4, rc2, des, 3des, des40, aes, skc_sh } BulkCipherAlgorithm;
```

В качестве объекта для сравнения с разрабатываемыми алгоритмами на основе *CBC*-блочной рюкзачной криптосистемы с общей памятью возьмем функцию из шифронабора *TLS* на основе алгоритмов по Диффи-Хеллману с *AES* [35] (*DH_AES*), так как данный тип алгоритмов среди все остальных стандартных функций шифронабора *TLS* обладает самыми высокими показателями скорости работы и криптостойкости. Обмен ключами Диффи-Хеллмана представляется более защищённым, так как установленный симметричный ключ никогда не покидает клиента или сервера и, соответственно, не может быть перехвачен злоумышленником, даже если тот знает закрытый ключ сервера, шифрование при использовании *AES* наиболее криптостойко.

В настоящее время из-за возможности применения L^3-атаки большинство рюкзачных криптосистем в прикладной криптографии не используется, поэтому основной целью данной диссертационной работы является разработка, исследование и модификация протокола криптозащищенной передачи информации (*TLS*) в каналах связи телекоммуникационных сетей *TCP/IP* с помощью алгоритмов *CBC*-блочной симметричной рюкзачной криптосистемы с общей памятью и разреженной плотностью укладки криптографического рюкзака в рамках криптографических двухсторонних протоколов [7, 8], которые будут криптостойки к L^3-атаке, обладают высокой скоростью работы и повышенной криптостойкостью по сравнению с используемыми на данный момент стандартными алгоритмами (*RSA_AES, DH_AES, RSA_3DES, DH_3DES* и т.д.) в *TLS*.

Алгоритм шифрования сообщений:

Шаг 1. Проверить наличие общей памяти между отправителем (*sender*) и получателем (*receiver*) сообщения. Если общая память существует, то перейти к шагу 3.

Шаг 2. Создать общую память: *sender* и *receiver* формируют множество $O = \{o_1, \ldots, o_b, o_B\}$, с символьными или числовыми элементами o_b. Обмен элементами множества между отправителем и получателем может происходить по закрытому КС, либо с помощью других надежных мер (личная передача и т.п.). В рамках модификации протокола *TLS* обмен элементами множества общей памяти происходит с использованием уже включенного в стандартный шифронабор *TLS* алгоритма Диффи-Хеллмана (*DH*).

Шаг 3. Сгенерировать вектор $E = \{e_1, \ldots, e_b, e_B\}$, состоящий из элементов «0» и «1», генерируемых случайным образом.

Шаг 4. Выбрать поле Галуа GF_p. Задать максимально возможный элемент p для данного поля. Выбор данного элемента зависит от аппаратной реализации технических средств, на которых будут выполняться алгоритмы шифрования и дешифрования. Для аппаратной архитектуры *x64* (также *AMD64 / Intel64 / EM64T*) зафиксируем $p \sim 2^{64}$.

Шаг 5. Вычислить все возможные суммы $\{\Psi\}$ произведений неповторяющихся элементов общей памяти и элементов вектора E $(1 \ldots 2^B-1)$.

Шаг 6. Отсортировать последовательность всех возможных сумм $\{\Psi\}$ по возрастанию. Выбрать одну сумму Ψ_0 как серединный элемент отсортированной последовательности, его позиция равна $(2^B-1)/2$.

Шаг 7. Сгенерировать по выбранному Ψ_0, который много меньше максимально возможного элемента, из множества сумм $\{\Psi\}$ базис (рюкзачный вектор) типа Фибоначчи $\{k_g(\Psi_0)\}$, определяемый последовательностью:

$$k_1(\Psi_0) = 1,\ k_2(\Psi_0) = 1 \oplus \Psi_0,\ \ldots,\ k_g(\Psi_0) = k_{g-1}(\Psi_0) \oplus k_{g-2}(\Psi_0),$$

для $g>2$, где $\oplus$ - операция сложения в выбранном поле Галуа GF_p, до тех пор, пока таких элементов $k_g(\Psi_0)$ не будет ровно G.

Шаг 8. Исходный текст разбить на блоки $M = M_1 \,||\, \dots \,||\, M_n \,||\, M_N$, где N – количество блоков в открытом сообщении; $M_n = m_{n1} \,||\, \dots \,||\, m_{ng} || \, m_{nG}$ - побитовое представление каждого блока открытого сообщения, G – количество бит в одном блоке и $m_{ng} \in \{0,1\}$.

Шаг 9. Начиная с первого блока, пока не закончится открытый текст, выполнить: в базисе $\{k_g(\Psi_0)\}$ открытое сообщение шифруется в закрытое сообщение:

$$S_n = \left(\sum_{n=1}^{G} m_g * \; k_g(\Psi_0) \right).$$

Шаг 10. Полученную последовательность $(e_1, \dots, e_b, e_B)$ и зашифрованное сообщение S передать получателю. Структура пакета, отправляемого получателю, выглядит следующим образом: $\{(e_1, \dots, e_b, e_B), S\}$.

Конец алгоритма.

Блок-схема алгоритма шифрования представлена на рисунке 2.4.

Существует логарифмический по скорости алгоритм $O(\log_2 S)$ нахождения S в базисе $\{k_g(\Psi_0)\}$, дающий на выходе значения $k_1(\Psi_0)$, $k_2(\Psi_0)$, … , $k_g(\Psi_0)$, определяющие симметричный ключ для шифрования и дешифрования.

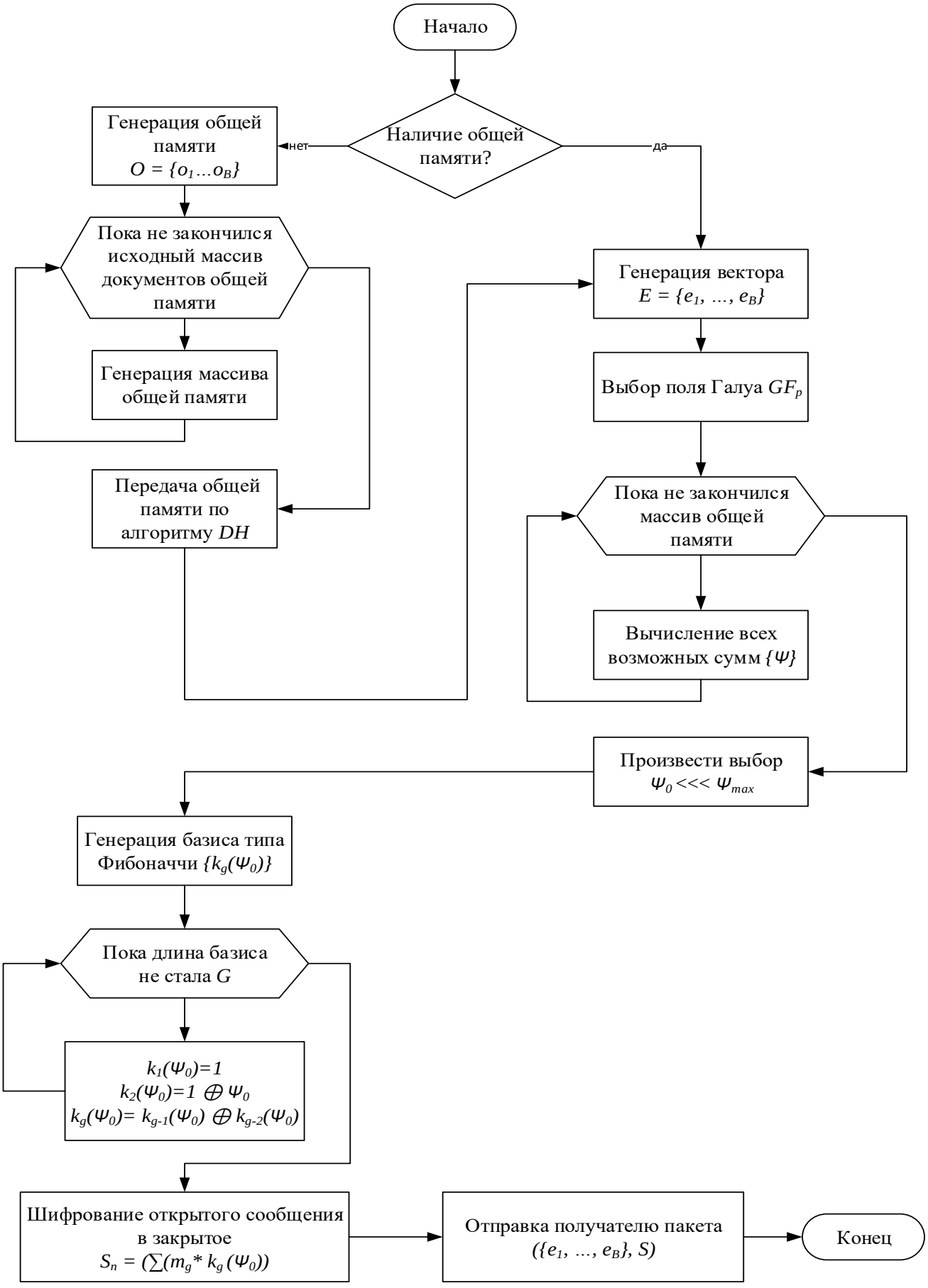

Рисунок 2.4 – Блок-схема алгоритма шифрования

Алгоритм дешифрования сообщений:

Шаг 1. Принять от *sender* последовательность $(e_1, \dots, e_B)$ и зашифрованное сообщение S.

Шаг 2. Вычислить все возможные суммы $\{\Psi\}$ произведений неповторяющихся элементов общей памяти и элементов вектора E $(1 \dots 2^B-1)$.

Шаг 3. Отсортировать последовательность всех возможных сумм $\{\Psi\}$ по возрастанию. Выбрать одну сумму Ψ_0 как серединный элемент отсортированной последовательности, его позиция равна $(2^B-1)/2$.

Шаг 4. Сгенерировать по выбранному Ψ_0, который много меньше максимально возможного элемента, из множества сумм $\{\Psi\}$ базис (рюкзачный вектор) типа Фибоначчи $\{k_g(\Psi_0)\}$, определяемый последовательностью:

$$k_1(\Psi_0) = 1,\ k_2(\Psi_0) = 1 \oplus \Psi_0,\ \dots,\ k_g(\Psi_0) = k_{g-1}(\Psi_0) \oplus k_{g-2}(\Psi_0),$$

для $g>2$, где $\oplus$ - операция сложения в выбранном поле Галуа GF_p, до тех пор, пока таких элементов $k_g(\Psi_0)$ не будет ровно G.

Шаг 5. Зашифрованный текст разбить на блоки $S = S_1 \,||\, \dots \,||\, S_n \,||\, S_N$, где N – количество блоков в закрытом сообщении; $S_n = s_{n1} \,||\, \dots \,||\, s_{ng} || s_{nG}$ - побитовое представление каждого блока закрытого сообщения, $s_{ng} \in \{0,1\}$.

Шаг 6. Исходное открытое сообщение $M = M_1 \,||\, \dots \,||\, M_n \,||\, M_N$ расшифровать с помощью «жадного алгоритма». Начиная с первого блока зашифрованного сообщения S, пока не закончатся зашифрованные блоки, выполнить: в базисе $\{k_g(\Psi_0)\}$ закрытое сообщение дешифруется в открытое сообщение. Пока $s_{ng} > k_g(\Psi_0)$ реализовать шаг 7, иначе выполнить условие в шаге 8 и вернуться к шагу 7.

Шаг 7. Задать битовое представление одного блока открытого сообщения $M_n = m_{n1} \,||\, \dots \,||\, m_{ng} || m_{nG}$. Отсюда побитово дешифровать сообщение как:

$$m_{ng} = s_{ng} - k_g(\Psi_0),\ m_{n(g-1)} = s_{n(g-1)} - k_{g-1}(\Psi_0),\ \dots,\ m_{n1} = s_{n1} - k_1(\Psi_0).$$

Шаг 8. Выполнить условие: когда $s_{ng} < k_g(\Psi_0)$ текущий элемент рюкзачного базиса пропустить и соответствующую разность осуществить по

предшествующему ему элементу рюкзачного вектора $k_{g-1}(\Psi_0)$. Пока не закончились символы (биты) зашифрованного блока выполнять шаги 7, 8.

Конец алгоритма.

Вышеописанным образом дешифровать все исходное сообщение и это можно сделать с логарифмической скоростью от значения S [3, 18, 22]. Блок-схема алгоритма дешифрования представлена на рисунке 2.5.

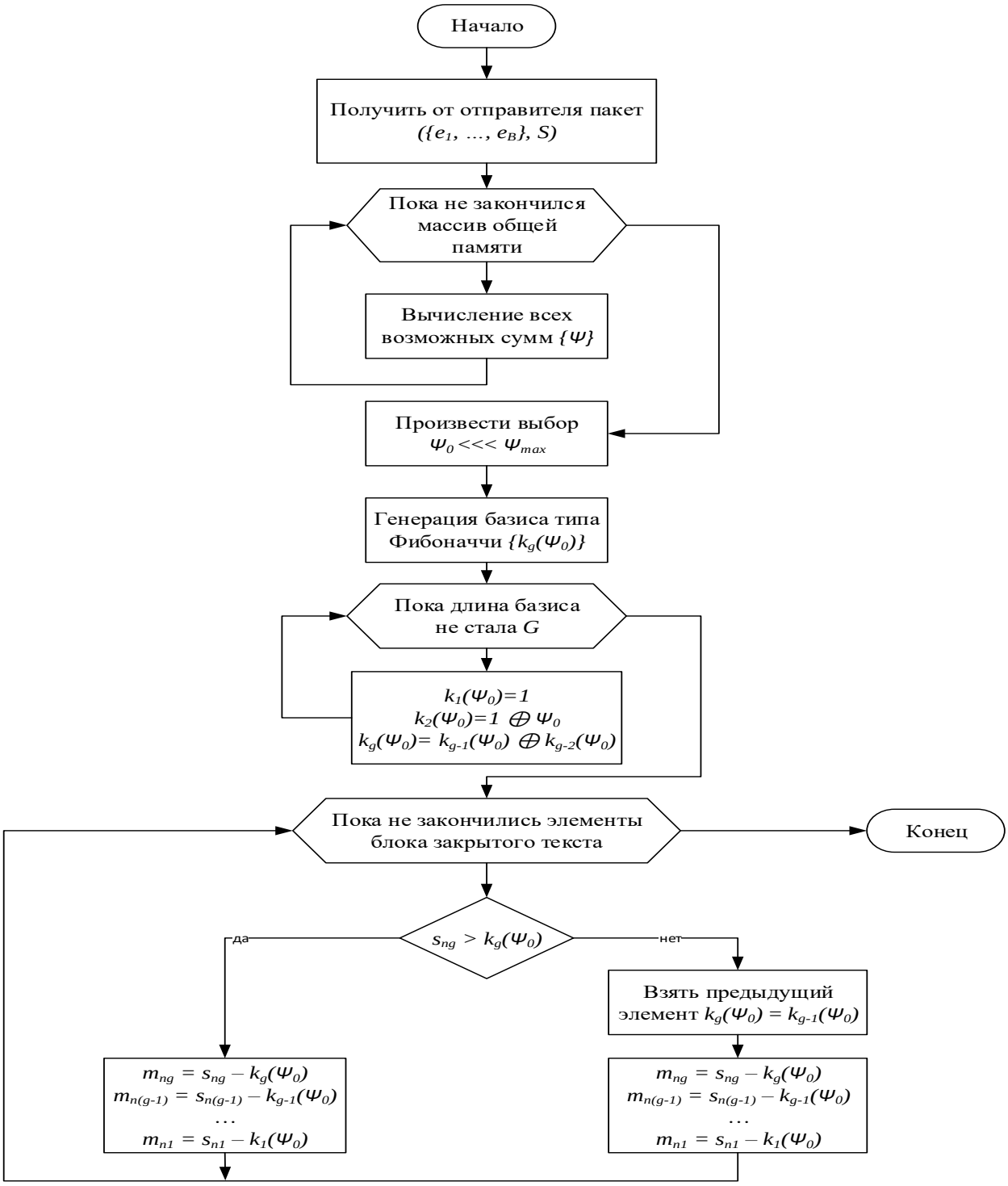

Рисунок 2.5 – Блок-схема алгоритма дешифрования

2.5 *CBC*-блочная модификация симметричной рюкзачной криптосистемы с общей памятью

На основе приведенных конструкций [4, 6, 16, 17] в реализации алгоритмов шифрования и дешифрования может быть построен масштабируемый блочный симметричный шифр, с несколькими режимами работы, в том числе и режимами кодовой книги (рисунок 2.6) и сцепления блоков (рисунок 2.7).

Пусть M – любой двоичный файл произвольной длины. Зададим конкатенацию блоков $M = M_1 \,||\, M_2 \,||\, \ldots \,||\, M_N$, где $M_n \in GF_2$, где длина каждого блока, за исключением последнего, фиксирована и равна G. При необходимости последний блок открытого текста дополняется до заданной длины незначащими символами, например «0». Алгоритм, который находит $M \leftrightarrow (e_1, \ldots, e_B,\ S)$ есть функция шифрования блока, которую обозначим $F_k(M)$. Обратный алгоритм – функция дешифрования $F_k^{-1}(S)$.

Рисунок 2.6 – Симметричный блочный шифр в режиме кодовой книги

В режиме зацепления блоков каждый блок открытого текста, кроме вектора инициализации, побитово складывается по модулю 2 с предыдущим результатом шифрования. Пусть S_0 – вектор инициализации (блок формируется с помощью фиксированного одностороннего алгоритма на основе общей памяти), тогда:

$$S_n = F_k\,(M_n \oplus S_{n-1}), \qquad (2.4)$$

где n – номер текущего блока; F_k – вышеописанный спроектированный алгоритм шифрования, используемый в симметричной рюкзачной криптосистеме с общей памятью; M_n – блок открытого текста.

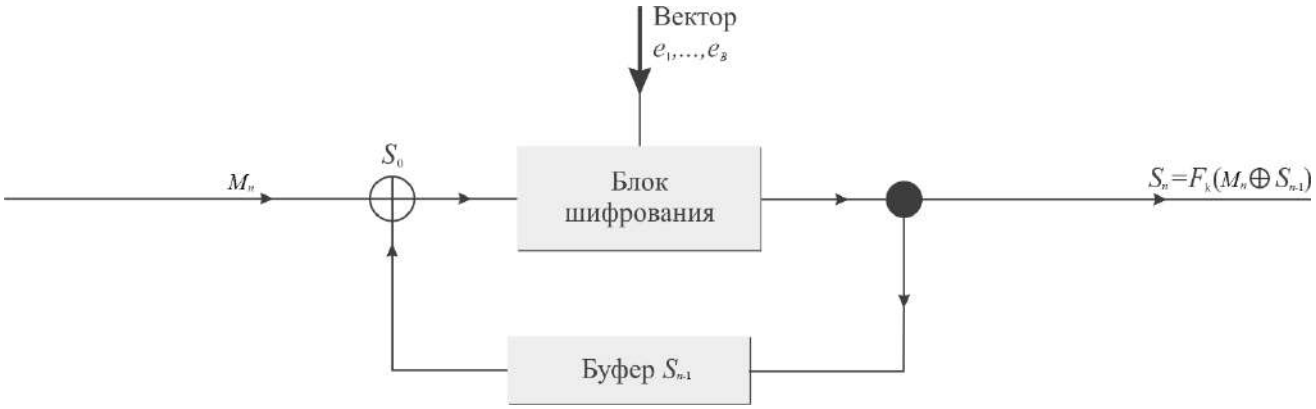

Рисунок 2.7 – Симметричный *CBC*-блочный рюкзачный шифр

CBC-блочный алгоритм шифрования сообщений:

Шаг 1. Сгенерировать вектор инициализации S_0. Блок формируется с помощью фиксированного одностороннего алгоритма на основе общей памяти $S_0 = f(O)$, где O – общая память между отправителем и получателем, f – фиксированная односторонняя функция.

Шаг 2. Исходный открытый текст разбить на блоки $M = M_1 \,||\, M_2 \,||\, \ldots \,||\, M_N$, длиной G-двоичных цифр каждый.

Шаг 3. Начиная с первого блока, пока не закончится открытый текст, выполнить: открытое сообщение шифруется в закрытое сообщение таким образом, что каждый блок открытого текста, кроме вектора инициализации, побитово складывается по модулю 2 с предыдущим результатом шифрования. Алгоритм шифрования каждого отдельного блока представлен в п.2.4.

Шаг 4. Полученную последовательность $(e_1, \ldots, e_B)$ и зашифрованное сообщение S передать получателю. Структура пакета, отправляемого получателю, выглядит аналогичным образом: $\{(e_1, \ldots, e_B)\ , S\}$.

Конец алгоритма.

Блок-схема *CBC*-блочного алгоритма шифрования представлена на рисунке 2.8.

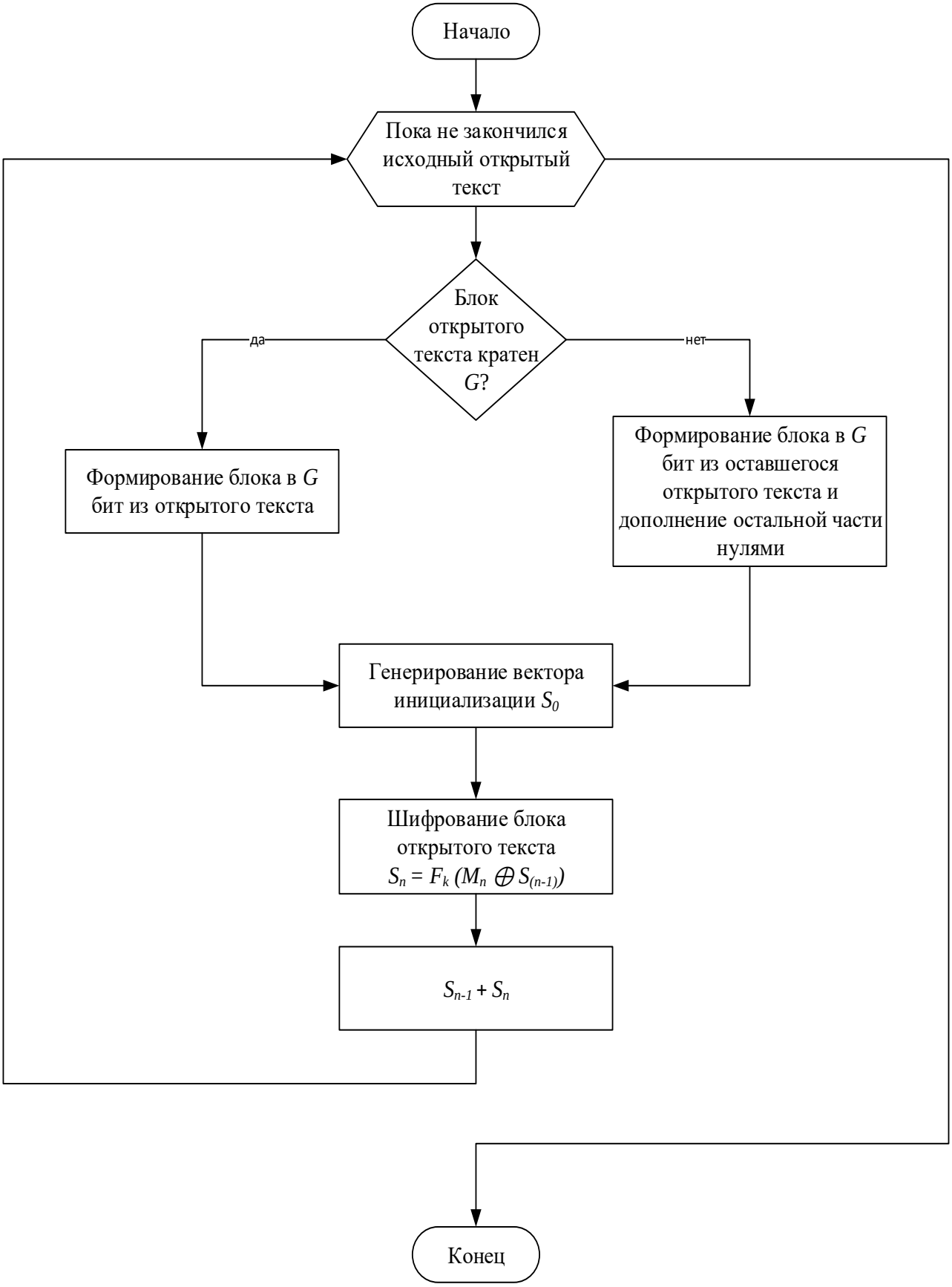

Рисунок 2.8 – Блок-схема *CBC*-блочного алгоритма шифрования

CBC-блочный алгоритм дешифрования сообщений:

Шаг 1. Сгенерировать вектор инициализации $M_0 = S_0$. Блок формируется с помощью фиксированного одностороннего алгоритма на основе общей памяти $M_0 = f(O)$, где O – общая память между отправителем и получателем, f – фиксированная односторонняя функция.

Шаг 2. Исходный закрытый текст разбить на блоки $S = S_1 \,||\, S_2 \,||\, \ldots \,||\, S_N$, длиной G-двоичных цифр каждый.

Шаг 3. Начиная с первого блока, пока не закончится закрытый текст, выполнить: закрытое сообщение дешифруется в открытое сообщение таким образом, что каждый блок закрытого текста, кроме вектора инициализации, побитово складывается по модулю 2 с предыдущим результатом дешифрования. Алгоритм дешифрования каждого отдельного блока также представлен в п.2.4.

Конец алгоритма.

Блок-схема *CBC*-блочного алгоритма дешифрования представлена на рисунке 2.9.

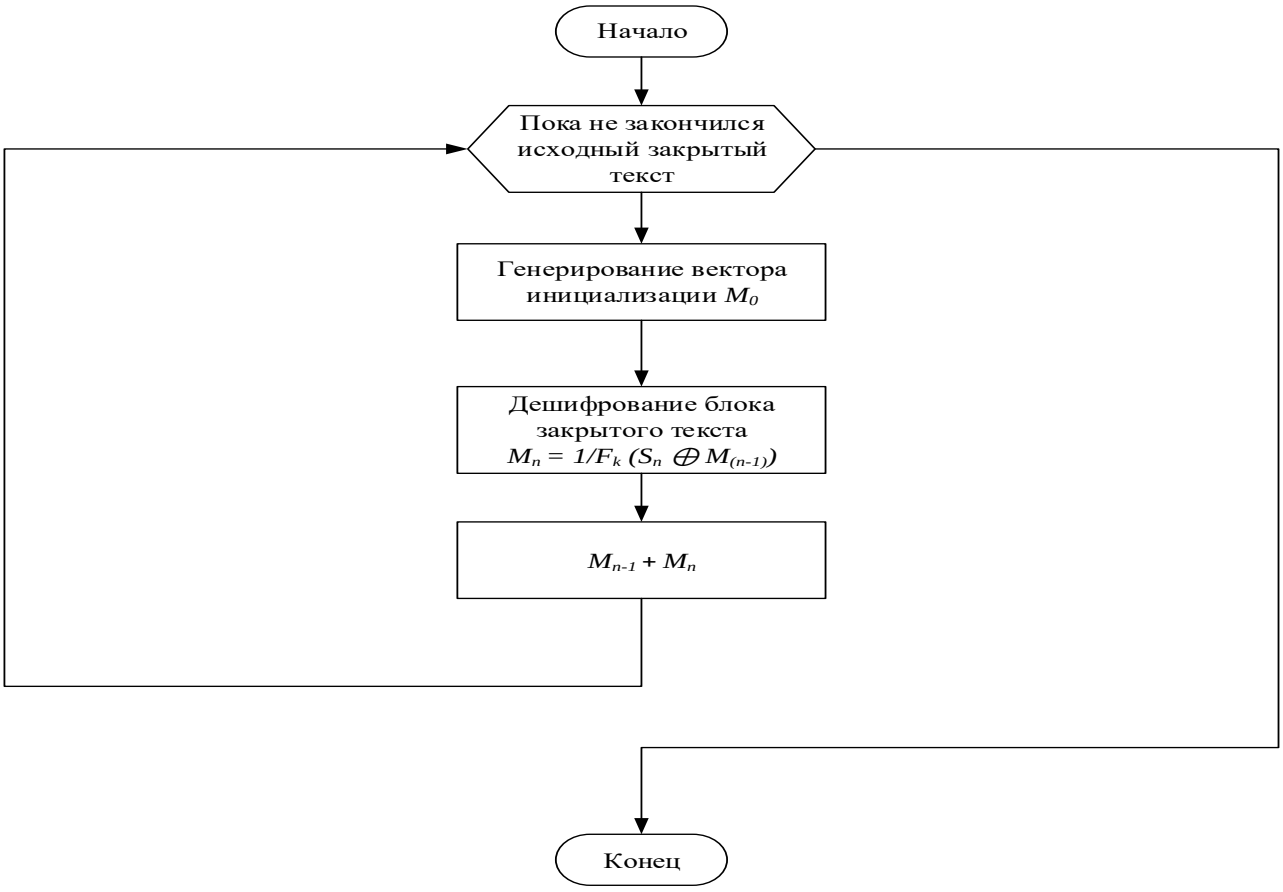

Рисунок 2.9 – Блок-схема *CBC*-блочного алгоритма дешифрования

В реализованной конструкции блочного шифра длина блока текста равна G бит, максимально возможный размер общей памяти оценивается в 2^G бит. Длина ключевой последовательности (рюкзачного базиса) $\{k_1(\Psi_0),\ k_2(\Psi_0)\ , \dots, k_g(\Psi_0)\}$ также равна G бит. Длина вектора $(e_1, \dots, e_B)$ равна количеству элементов в общей памяти B.

Преимущества предложенного подхода к организации дополненного шифронабора *TLS* на основе *CBC*-блочной симметричной рюкзачной криптосистемы:

- Уход от атаки [19] Лагариаса – Одлыжко (L^3-атака) за счет перевода плотности укладки рюкзачной криптосистемы за пределы интервала (0,1), где эта атака и применима на все предшествующие версии симметричных рюкзачных криптосистем, что до этого момента не позволяло использовать эти шифры в коммерческих приложениях.
- Невозможность взлома схемы шифрования методом *bruteforce* за счет секретности общей памяти, от которой параметрически зависят сами алгоритмы шифрования и дешифрования.
- Скоростные характеристики шифра [8] и его требуемый уровень криптостойкости позволяют использовать его в композиции со стандартами шифрования, используемых в сетевых протоколах передачи данных, а также в дополнения к стандартным алгоритмам шифронабора в *TLS*.
- Масштабирование длины ключа (за счет изменения объема общей памяти).
- Вариативность алгоритма шифрования в зависимости от целочисленных значений общей памяти.

На основе реализации блочного шифра с помощью стандартной схемы свертки блоков за счет общей памяти появляется возможность построения параметризованного по наличию общей памяти стандартного алгоритма хеширования, применяемого для контроля достоверности передаваемой информации от отправителя к получателю, вместо *HMAC*, который используется в

TLS. Свертка блочного шифра порождает такую хэш-функцию для пары (*Sender*, *Receiver*), которая является строго индивидуальной. В самом деле, все построения, в том числе и хеш-функция непрерывно зависят от согласованного параметра Ψ_0 общей памяти этой и только этой пары (*Sender*, *Receiver*).

В качестве вектора инициализации хеш-функции $H_0 = const$ будем использовать массив данных, полученный на основе общей памяти с помощью специального необратимого преобразования, в качестве которого выступает односторонняя функция.

В качестве алгоритма формирования выходного значения хеш-функции – итогового хеш-значения (хеш-суммы) будем использовать любой ключевой алгоритм. Ключом в данном алгоритме будет служить последовательность $\{k_1(\Psi_0), k_2(\Psi_0) \ , \dots , k_g(\Psi_0)\}$,полученная в ходе формирования закрытого сообщения S

$$H_n = (S \oplus H_{n-1})_k. \qquad (2.5)$$

Наличие общей памяти в приведенной криптосистеме [5, 8, 9], конечно, выходит за рамки шенноновской модели симметричного шифрования, однако, не противоречит криптографической модели безопасности Долев-Яо, представленной в [39]. Частный случай шенноновской модели шифрования получается в единственном случае, когда общая память отсутствует. В общем случае представленная криптосистема безопасна в модели Долев–Яо при условии невозможности компрометации общей памяти.

Несомненными достоинствами разработанного метода организации защищенного канала связи в сетях *TCP/IP*, являются: превосходная логарифмическая скорость алгоритмов шифрования и дешифрования; масштабируемость длины ключа, которая непрерывно зависит от размеров общей памяти и от выбора конечного поля.

Выводы к главе 2

1. Разработано семейство алгоритмов (шифрования, дешифрования и хеширования информации) для повышения производительности и криптостойкости симметричных рюкзачных криптосистем в защищенных КС сетей *TCP/IP*, которые отличаются от предыдущих версий рюкзачных криптосистем (симметричных систем в общем) наличием общей памяти, высоким уровнем плотности укладки рюкзака и отказом от супервозрастающих базисов в пользу линейно - рекуррентных последовательностей.

2. Предложена модификация разработанной симметричной рюкзачной криптосистемы с общей памятью семейством *CBC*-блочных алгоритмов шифрования и дешифрования информации, что еще в большей мере повысило криптостойкость разработанной системы.

3. Внедряя разработанную симметричную рюкзачную криптосистему с общей памятью в такие фазы работы протокола *TLS*, как фаза аутентификации клиента и сервера, фаза создания кода аутентификации сообщений и фаза работы симметричных блочных алгоритмов шифрования и дешифрования сообщений, мы тем самым существенно повышаем производительность протокола *TLS* и его криптостойкость, так как разработанное семейство алгоритмов обладает массой преимуществ: высокая скорость работы, простота реализации, минимальное потребление ресурсов пользователя, отличная масштабируемость.

3 ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ ЗАЩИЩЕННОГО КАНАЛА СВЯЗИ НА БАЗЕ СИММЕТРИЧНОЙ РЮКЗАЧНОЙ КРИПТОСИСТЕМЫ С ОБЩЕЙ ПАМЯТЬЮ

В главе приведены результаты экспериментального исследования предложенных средств организации защищенного канала связи. Получены скоростные характеристики разработанного семейства алгоритмов шифрования и дешифрования информации, выполнено *NIST*-тестирование.

3.1 Определение зависимости вероятности успешной реализации L^3-атаки от плотности укладки рюкзака

Для постановки необходимых экспериментов разработан стенд (рисунок 3.1).

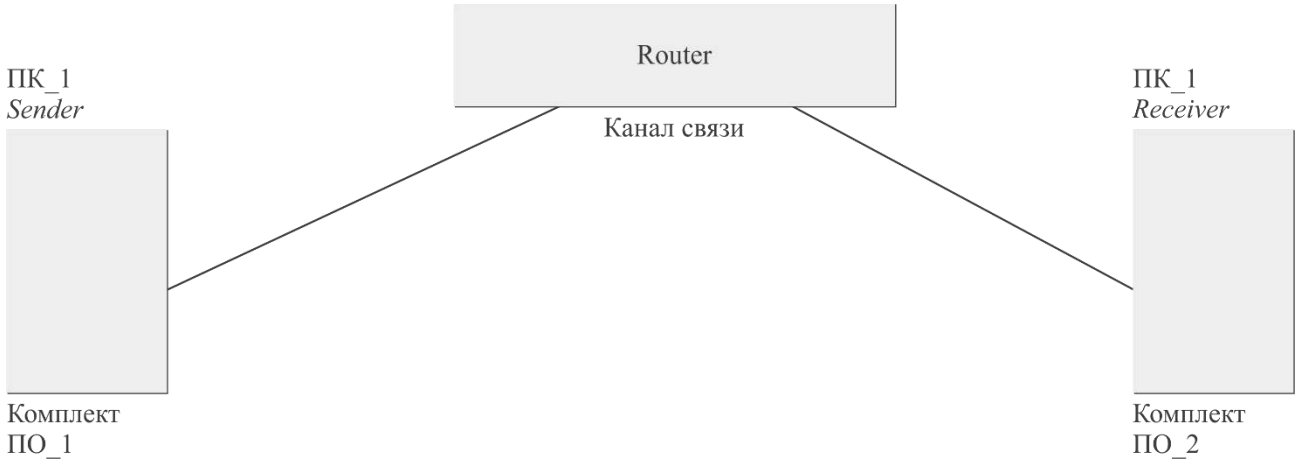

Рисунок 3.1 – Структурная схема лабораторной установки

Стенд состоит из двух персональных компьютеров, подключенных к одному каналу связи (маршрутизатору). Технические характеристики ПК №1 - *CPU*: *QuadCore Intel Core i*7-870, 3200 *MHz*; *RAM*: 16 *GB*; *HDD*: 1 *TB*; *GPU*: *Nvidia GTX* 970; *Network card*: 100 *MB/s*; ПК №2 - *CPU*: *AMD Phenom x4 940*, 2400 *MHz*; *RAM*: 8 *GB*; *HDD*: 500 *GB*; *GPU*: *Nvidia GTX* 470; *Network card*: 100 *MB/s*; Канал связи - *Router*: *Asus N18U*, пропускная способность 600 *MB/s*, поддержка гигабитной сети,

протоколы и стандарты: *IEEE*802.11*b*, *IEEE*802.11*g*, *IEEE*802.11*n*, *IPv*4, *IPv*6; витая пара категории 5*e*, стандарта 1000*BASE-T Ethernet*.

В комплект ПО №1 входит: приложение для генерации из выбранного массива документов общей памяти (*SM_generator*); приложение для шифрования и дешифрования сообщений с помощью разработанного семейства алгоритмов *CBC*-блочной симметричной рюкзачной криптосистемы с общей памятью и передачи выходных данных получателю (*Knapsack cryptosystem*).

В комплект ПО №2 входит: приложение для шифрования и дешифрования сообщений с помощью разработанного семейства алгоритмов *CBC*-блочной симметричной рюкзачной криптосистемы с общей памятью и приема исходящих от отправителя данных (*Knapsack cryptosystem*); приложение для реализации L^3-атаки на криптосистему (*LLL_system*); приложение для проведения *NIST*-тестирования (*NIST_system*); приложение для шифрования и дешифрования сообщений с помощью алгоритмов рюкзачной криптосистемы на супервозрастающих базисах (*s_growing_system*) и алгоритмов функции *DH_AES* из стандартного шифронабора *TLS* (*dh_aes_tls_system*).

Основные возможности вышеописанного ПО:

- Функция *establish_connection* (*dest_IP*, *dest_Port*): установка соединения по КС между ПК №1 отправителя и ПК №2 получателя, на вход функции поступают *IP*-адрес и порт ПК получателя.
- Функция *generate_SM* (*docs[ ]*): формирование общей памяти – последовательности $O = \{o_1, o_2, \ldots, o_B\}$ из выбранного массива документов отправителя.
- Функция *generate_vector_E* (*rand_seed*): генерация вектора $E = \{e_1, e_2, \ldots, e_B\}$ в соответствии с алгоритмом шифрования, показанного в п.2.4.
- Функция *generate_vector_K* (*vector_E*): формирование рюкзачной последовательности $\{k_1(\Psi_0),\ k_2(\Psi_0)\ , \ldots, k_g(\Psi_0)\}$ в соответствии с алгоритмом шифрования, показанного в п.2.4 и вектором *E*.

- Функция *encrypt_text* (*source_text, vector_K*): шифрование открытого текста *M* разбитого на блоки длины *G* в соответствии с алгоритмом *CBC*-блочного шифрования, показанного в п.2.4 и рюкзачным вектором $\{k_1(\Psi_0),\ k_2(\Psi_0)\ , \dots, k_g(\Psi_0)\}$.
- Функция *calculate_density* (*vector_K*): подсчет плотности укладки (2.3) полученной криптосистемы с общей памятью на основе последовательности $\{k_1(\Psi_0),\ k_2(\Psi_0)\ , \dots, k_g(\Psi_0)\}$.
- Функция *LLL_attack* (*encrypted_text*): реализация L^3-атаки на закрытый текст, полученный с помощью разработанной криптосистемы с общей памятью и вывод результатов о возможности ее проведения.
- Функция *decrypt_text* (*encrypted_text, vector_K*): дешифрование закрытого текста *S* разбитого на блоки длины *G* в соответствии с алгоритмом *CBC*-блочного дешифрования, показанного в п.2.4 и рюкзачным вектором $\{k_1(\Psi_0),\ k_2(\Psi_0)\ , \dots, k_g(\Psi_0)\}$.
- Функция *nist_testing* (*encrypted_text, result_1 … result_15*): реализация всех пятнадцати *NIST*-тестов на закрытый текст, полученный с помощью разработанной криптосистемы с общей памятью и вывод результатов их прохождения.
- Функция *s_growing_enc_dec* (*source_text, encrypted_text*): шифрование открытого текста с помощью алгоритмов рюкзачной криптосистемы на супервозрастающих базисах и дешифрование полученного закрытого.
- Функция *dh_aes_tls_enc_dec* (*source_text, encrypted_text*): шифрование открытого текста с помощью алгоритмов функции *DH_AES* из стандартного шифронабора *TLS* и дешифрование полученного закрытого.

Эксперимент 1.

Цель эксперимента – исследовать зависимости вероятности успешной реализации L^3-атаки от плотности укладки рюкзака ρ при различных объемах исходного текста.

Под количественной характеристикой криптостойкости будем понимать величину

$$W = \frac{\rho*(1-P)*\omega}{\rho_{max}}, \quad (3.1)$$

где ρ – плотность укладки рюкзака, P - вероятность успешной реализации L^3-атаки в спроектированной криптосистеме, ρ_{max} – предельная плотность ($\rho_{max} \approx 1{,}45$), ω – показатель статистической защищенности, где $\omega = 1$, если криптосистема успешно прошла все стандартные *NIST*-тесты, иначе $\omega = 0$.

Для проведения данного эксперимента случайным образом были отобраны 20 файлов, приведенных в таблице 3.1.

Таблица 3.1 – Исходная база документов

№ п/п	Тип файла	Размер, *Mb*
1	**.txt*	0.2
2	**.txt*	2.0
3	**.bmp*	8.0
4	**.gif*	1.0
5	**.exe*	3.0
6	**.mp3*	5.0
7	**.mp4*	2000.0
8	**.html*	0.4
9	**.rar*	1.0
10	**.7z*	2.0
11	**.avi*	750.0

Продолжение таблицы 3.1

№ п/п	Тип файла	Размер, *Mb*
12	**.3g*	107.0
13	**.doc*	2.4
14	**.docx*	3.7
15	**.exe*	7.4
16	**.xls*	0.8
17	**.xlsx*	4.2
18	**.zip*	3.0
19	**.jpg*	0.9
20	**.jpeg*	1.2

План проведения эксперимента:

Для каждого из файлов (таблица 3.1) выполнить:

1. Сформировать криптосистему на ПК №1: *generate_SM* (*docs[ ]*) - сформировать общую память $\{o_1, o_2, \ldots, o_B\}$; *generate_vector_E* (*rand_seed*) - сгенерировать вектор $E = \{e_1, e_2, \ldots, e_B\}$; *generate_vector_K* (*vector_E*) – сформировать рюкзачный вектор $\{k_1(\Psi_0),\ k_2(\Psi_0)\ , \ldots, k_g(\Psi_0)\}$.
2. Подсчитать значение плотности укладки: *calculate_density* (*vector_K*).
3. Зашифровать исходный текст: *encrypt_text* (*source_text*, *vector_K*).
4. Передать зашифрованный текст и вектор $E = \{e_1, e_2, \ldots, e_B\}$ получателю: *establish_connection* (*dest_IP*, *dest_Port*).
5. Принять на ПК №2 зашифрованный текст и вектор $E = \{e_1, e_2, \ldots, e_B\}$ от отправителя.
6. Сформировать криптосистему на ПК №2: *generate_SM* (*docs[ ]*) - сформировать общую память $\{o_1, o_2, \ldots, o_B\}$; *generate_vector_E* (*rand_seed*) - сгенерировать вектор $E = \{e_1, e_2, \ldots, e_B\}$; *generate_vector_K* (*vector_E*) – сформировать рюкзачный вектор $\{k_1(\Psi_0),\ k_2(\Psi_0)\ , \ldots, k_g(\Psi_0)\}$.
7. Дешифровать полученный текст: *decrypt_text* (*encrypted_text*, *vector_K*).

8. Реализовать L^3-атаку на полученный зашифрованный текст, сравнить полученный результат с результатом дешифрования. Если обнаружено совпадение, зарегистрировать случай прохождения атаки, обозначим эту величину как $P = amount\ (\rho)$. Повторить шаг 8 100 раз.

Полученные результаты представлены в приложении Б. На их основе построены сводные таблицы (таблицы 3.2-3.4) и графики зависимости (рисунки 3.2-3.4).

Таблица 3.2 – Зависимость W от плотности укладки рюкзака при $G = 64$ бита

№ п/п	Значение ρ	Значение P	Значение W
1	0,9989735	0,0120000	0,6806798
2	0,9946581	0,0130000	0,6770534
3	0,9940932	0,0130000	0,6766689
4	0,9933907	0,0140000	0,6755056
5	0,9925017	0,0145000	0,6745589
6	0,9913402	0,0170000	0,6720602
7	0,9937134	0,0135000	0,6760677
8	0,9928858	0,0140000	0,6751623
9	0,9936028	0,0130000	0,6763351
10	0,9907102	0,0185000	0,6706083

Рисунок 3.2 – График зависимости W от ρ при G = 64 бита

Таблица 3.3 – Зависимость W от плотности укладки рюкзака при G = 128 бит

№ п/п	Значение ρ	Значение P	Значение W
1	1,1961437	0,0060000	0,8199771
2	1,1941984	0,0075000	0,8174082
3	1,1980714	0,0045000	0,8225379
4	1,1991653	0,0040000	0,8237025
5	1,1910745	0,0095000	0,8136270
6	1,1978640	0,0050000	0,8219825
7	1,1974519	0,0055000	0,8212868
8	1,1968856	0,0060000	0,8204857
9	1,1975411	0,0050000	0,8217609
10	1,1937643	0,0080000	0,8166994

Рисунок 3.3 – График зависимости W от ρ при G = 128 бит

Таблица 3.4 – Зависимость W от плотности укладки рюкзака при G = 256 бит

№ п/п	Значение ρ	Значение P	Значение W
1	1,4419913	0,0020000	0,9924878
2	1,4427653	0,0010000	0,9940155
3	1,4429971	0,0010000	0,9941752
4	1,4416151	0,0020000	0,9922288
5	1,4415241	0,0020000	0,9921662
6	1,4427716	0,0010000	0,9940198
7	1,4429891	0,0010000	0,9941697
8	1,4419987	0,0020000	0,9924928
9	1,4425519	0,0010000	0,9938685
10	1,4427091	0,0010000	0,9939768

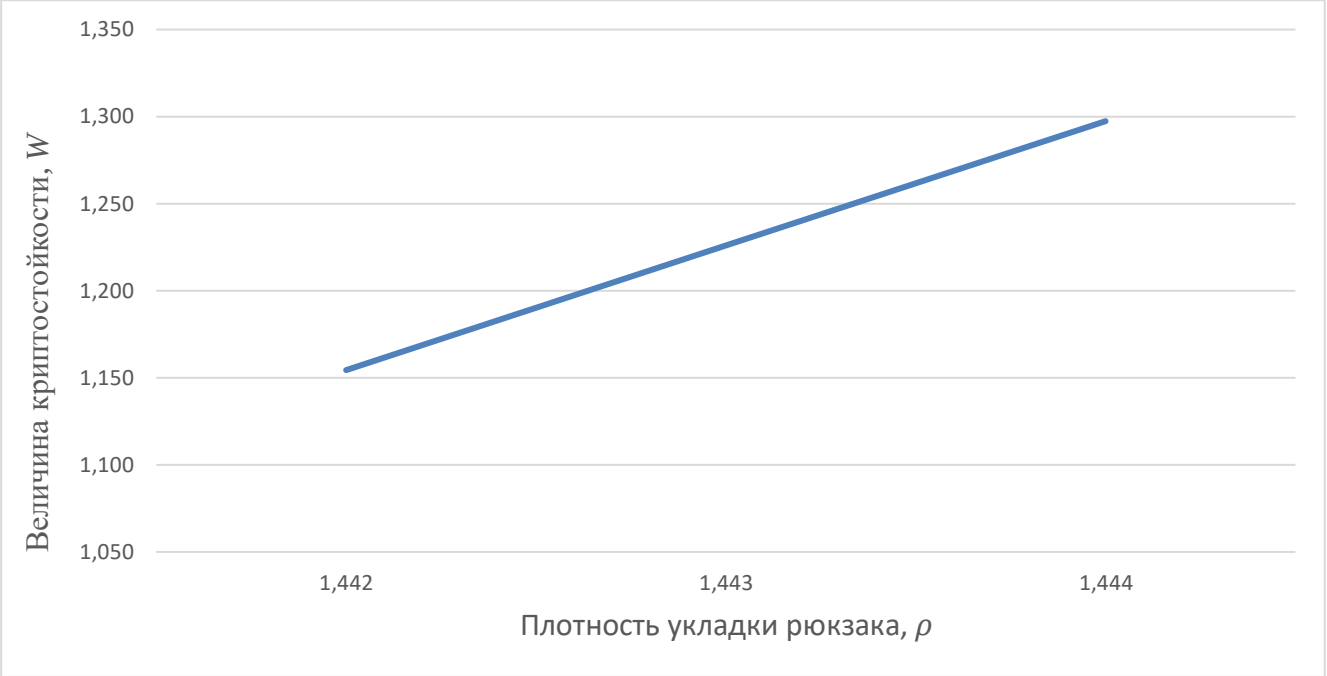

Рисунок 3.4 – График зависимости W от ρ при $G = 256$ бит

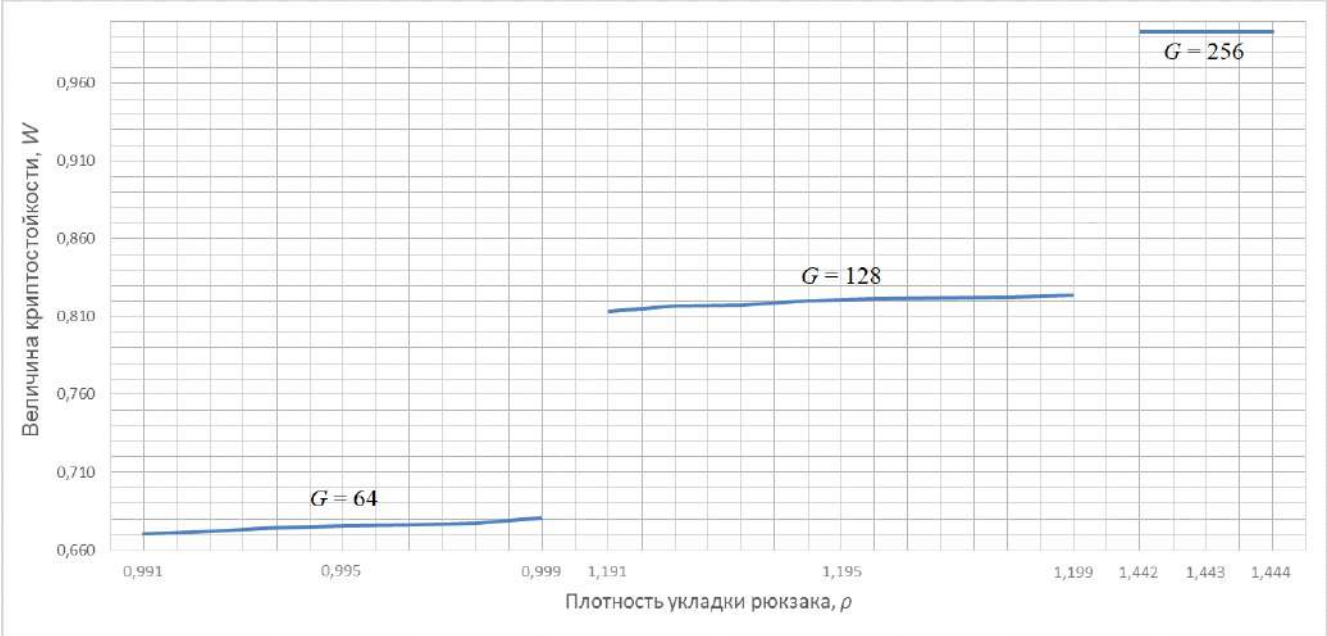

Рисунок 3.5 – Сводный график зависимости W от ρ

Итог экспериментального исследования позволяет сделать следующие выводы:

1. Предложенная *CBC*-блочная симметричная рюкзачная криптосистема с общей памятью обладает плотностью укладки не менее 0.9408, что в

соответствии с рекомендациями Костера – Одлыжко [32, 50, 57] способствует затруднению реализации L^3-атаки на нее.

2. Разработанная криптосистема с общей памятью при минимальном размере рюкзачного базиса в G = 64 бита имеет плотность укладки в пределах $0.9907 \leq \rho \leq 0.9989$, что удовлетворяет требованиям необходимого уровня криптостойкости.
3. При размерах рюкзачного базиса в G = 128 бит, плотность укладки принимает значения $1.1910 \leq \rho \leq 1.1991$, при размерах рюкзачного базиса в G = 256 бит, плотность укладки принимает значения $1.4415 \leq \rho \leq 1.4429$. При увеличении размера G повышается криптостойкость разработанной системы. Выбор длины блока для передачи необходимо осуществлять в зависимости от важности и объема передаваемых сведений, но не менее чем G = 64 бита.
4. В соответствии с рекомендациями Костера – Одлыжко количественная оценка криптостойкости для всех предыдущих вариантов рюкзачных криптосистем имеет значение $W = (0.9408 * (1 – 0.9)) / 1{,}45 = 0.06$, для предложенной криптосистемы с общей памятью минимальное значение W = 0,67, что говорит о значительно более высоком уровне криптостойкости по отношению к L^3-атаке.
5. При величине плотности укладки в пределах $0.9907 \leq \rho \leq 0.9989$ вероятность того, что успешно реализуется L^3-атака, составляет 1 - 2 % от общего числа попыток реализовать атаку, при $1.1910 \leq \rho \leq 1.1991$ составляет 0.4 – 0.9 % и при $1.4415 \leq \rho \leq 1.4429$ составляет 0.1 – 0.2 % от общего числа попыток.

Эксперимент 2.

Цель эксперимента – исследовать наличие статистической зависимости вероятности успешной реализации L^3-атаки от плотности укладки рюкзака ρ при различных объемах исходного текста.

Эксперимент был проведен для тех же 20 файлов, приведенных в таблице 3.1.

В план проведения первого эксперимента добавлены этапы:

9. Сформировать множество $\rho = \{ \rho_1, \rho_2, \ldots, \rho_{25}\}$ – множество плотностей укладок для сформированных в ходе эксперимента криптосистем с общей памятью.
10. Сформировать множество $P = \{ P_1, P_2, \ldots, P_{25}\}$ – множество вероятностей, когда в спроектированной криптосистеме успешно реализуется L^3-атака.
11. Подсчитать ковариацию и корреляцию для двух заданных множеств ρ и P.

Полученные результаты представлены в приложении В. На их основе построена сводная таблица числовых значений множеств ρ и P (таблица 3.5).

Таблица 3.5 – Значение множеств ρ и P

№ п/п	Значение элементов ρ	Значение элементов P
1	0,9989735	0,0120000
2	0,9946581	0,0130000
3	0,9940932	0,0130000
4	0,9933907	0,0140000
5	0,9925017	0,0145000
6	0,9913402	0,0170000
7	0,9985134	0,0135000
8	0,9928858	0,0140000
9	0,9936028	0,0130000
10	0,9907102	0,0185000
11	0,9981128	0,0130000
12	0,9942135	0,0130000
13	0,9939589	0,0135000
14	0,9947205	0,0140000
15	0,9981761	0,0120000
16	0,9923125	0,0145000

Продолжение таблицы 3.5

№ п/п	Значение элементов ρ	Значение элементов P
17	0,9949875	0,0130000
18	0,9949812	0,0130000
19	0,9914219	0,0170000
20	0,9948714	0,0130000
21	0,9989853	0,0120000
22	0,9947814	0,0130000
23	0,9989918	0,0110000
24	0,9997614	0,0100000
25	0,9998912	0,0100000

Для определения наличия статистической зависимости вероятности успешной реализации L^3-атаки от плотности укладки рюкзака ρ при различных объемах исходного текста был произведен подсчет ковариации для двух заданных множеств ρ и P. Зависимость между двумя случайными выборками существует в том случае, когда коэффициент корреляции равен или близок к единице по модулю.

Сначала вычисляем коэффициент корреляции по следующей формуле:

$$R_{\rho,P} = \frac{cov(\rho,P)}{\sigma_\rho \sigma_P}, \tag{3.2}$$

где $cov\ (\rho, P)$ – коэффициент ковариации между заданными множествами ρ и P, σ_ρ и σ_P – среднеквадратические отклонения случайных величин ρ и P.

Коэффициент ковариации характеризует степень линейной зависимости двух случайных величин ρ и P и вычисляется по формуле

$$cov(\rho,P) = \frac{1}{j}\sum_{i=1}^{j}(\rho_i - M_\rho)(P_i - M_P), \tag{3.3}$$

где M_ρ и M_P – математические ожидания и ρ_i, P_i – случайные величины из двух заданных множеств ρ и P.

В итоге получили коэффициент ковариации (таблица А.1):

M_ρ = 0.99523300.

M_P = 1.33800000.

cov (ρ, *P*) = 0.00047900.

Далее вычисляем коэффициент корреляции по формуле (3.2) для этого воспользуемся результатами, представленными в таблице А.1, дополним последнюю новыми столбцами (получим таблицу А.2), в которую запишем (предварительно вычислив) значения квадратов центрированных случайных величин $(\rho_i - M_\rho)^2$ и $(P_i - M_P)^2$.

σ_ρ^2 = 0.0000080.

σ_P^2 = 0.0366560.

R = 0.8960480.

Получаем, что между вероятностью успешной реализации L^3-атаки и плотностью укладки рюкзака при различных объемах исходного текста есть сильная статистическая зависимость (значение |R| ~ 0.90).

Построим диаграмму рассеяния (корреляционное поле) и график линии регрессии:

Диаграмма рассеяния — это графическое изображение соответствующих пар (ρ_i, P_i) в виде точек плоскости, в прямоугольных координатах с осями. Корреляционное поле является одним из графических представлений связанной (парной) выборки. В той же системе координат строится и график линии регрессии. Следует тщательно выбрать масштабы и начальные точки на осях, чтобы диаграмма была максимально наглядной.

На оси абсцисс разместим значения ρ_i, а на оси ординат значения P_i. Нанесем точки (ρ_1, P_1), (ρ_2, P_2), …, (ρ_{25}, P_{25}) на координатную плоскость. Получим диаграмму рассеяния (корреляционное поле), изображенное на рисунке 3.6. Линия регрессии показана красным цветом.

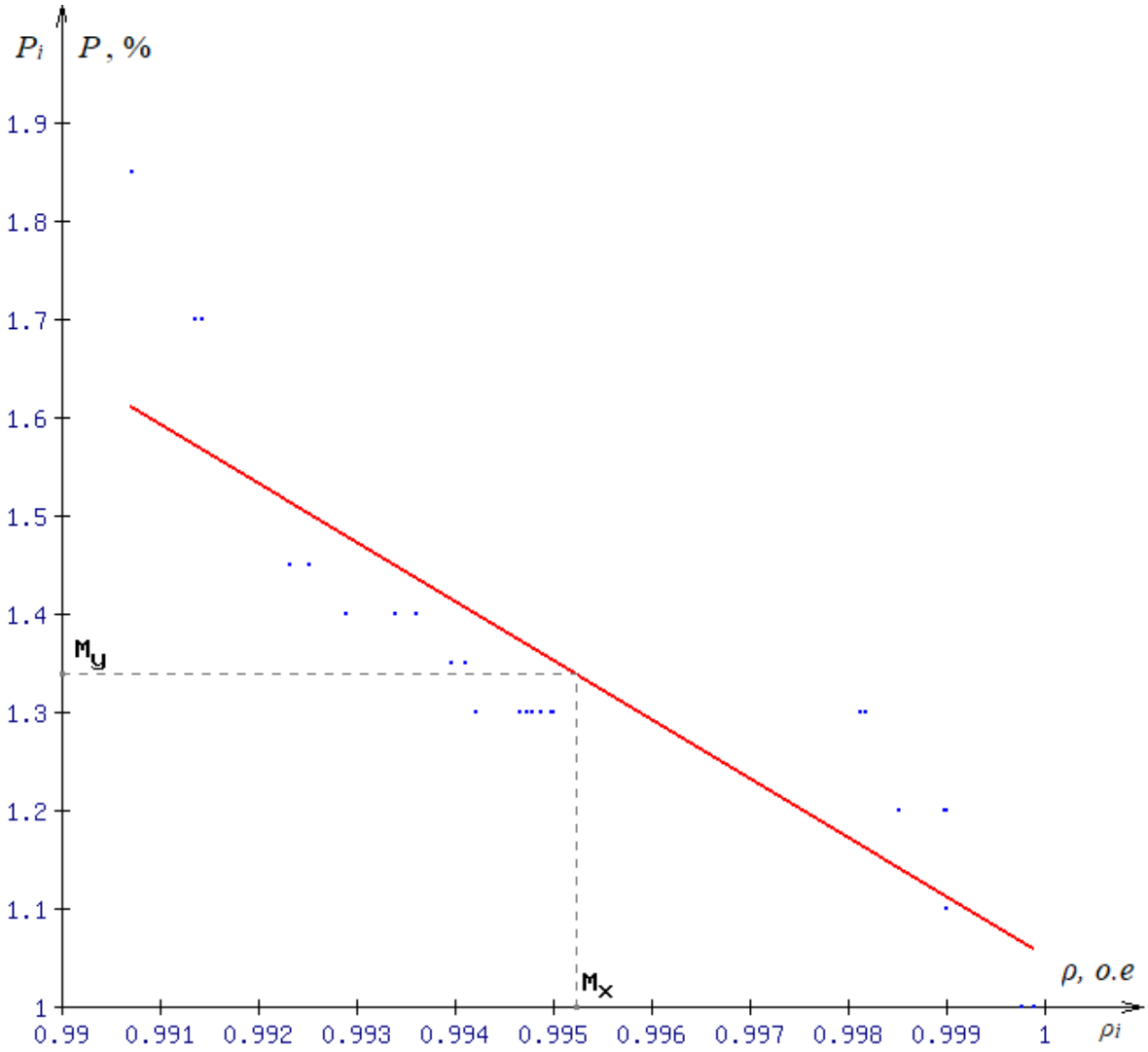

Рисунок 3.6 – График зависимости двух выборок P от ρ при G = 64 бита

Результаты эксперимента:

1. Между вероятностью успешной реализации L^3-атаки и плотностью укладки рюкзака при различных объемах исходного текста есть сильная статистическая зависимость (значение $|R| \sim 0.90$).
2. С возрастанием величины плотности укладки предложенного рюкзака уменьшается вероятность успешной реализации L^3-атаки в спроектированной *CBC*-блочной криптосистеме с общей памятью.

3.2 *NIST*-тестирование

Цель эксперимента – провести *NIST* – тестирование для практического доказательства высокого уровня криптостойкости предложенной криптосистемы. Результаты тестов направлены на подтверждение того факта, что последовательность, являющая результатом работы *CBC*-блочного шифра, похожа на случайную.

План проведения тестирования:

1. Сформировать криптосистему на ПК №2: *generate_SM* (*docs[ ]*) - сформировать общую память *{o_1, o_2, …, o_B}*; *generate_vector_E* (*rand_seed*) - сгенерировать вектор *E = {e_1, e_2, …, e_B}*; *generate_vector_K* (*vector_E*) – сформировать рюкзачный вектор $\{k_1(\Psi_0),\ k_2(\Psi_0)\ , \dots, k_g(\Psi_0)\}$.
2. Зашифровать исходный текст: *encrypt_text* (*source_text, vector_K*).
3. Провести *NIST* – тестирование на полученном закрытом тексте: *nist_testing* (*encrypted_text, result_1 … result_15*). Зарегистрировать результаты прохождения всех тестов, получить значение *p* для каждого теста.

Статистические тесты *NIST* — пакет статистических тестов [45], разработанный лабораторией информационных технологий (*ITL*), являющейся главной исследовательской организацией национального института стандартов и технологий (*NIST*). В его состав входят 15 статистических тестов, целью которых является определение меры случайности двоичных последовательностей, порождённых либо аппаратными, либо программными генераторами случайных чисел. Эти тесты основаны на различных статистических свойствах, присущих только случайным последовательностям.

Тест №1 - Частотный побитовый тест.

Суть данного теста заключается в определении соотношения между нулями и единицами во всей двоичной последовательности. Цель — выяснить, действительно ли число нулей и единиц в последовательности приблизительно одинаковы, как это можно было бы предположить в случае истинно случайной

бинарной последовательности. Если вычисленное в ходе теста значение $r < 0{,}01$, то данная двоичная последовательность не является истинно случайной. В противном случае последовательность носит случайный характер.

Тест №2 - Частотный блочный тест.

Суть теста — определение доли единиц внутри блока длиной G бит. Цель — выяснить действительно ли частота повторения единиц в блоке длиной G бит приблизительно равна $G/2$, как можно было бы предположить в случае абсолютно случайной последовательности. Вычисленное в ходе теста значение r должно быть не меньше 0,01. В противном случае ($r < 0{,}01$) двоичная последовательность не носит истинно случайный характер.

Тест №3 - Тест на последовательность одинаковых битов.

Суть состоит в подсчёте полного числа рядов в исходной последовательности. Цель данного теста — сделать вывод о том, действительно ли количество рядов, состоящих из единиц и нулей с различными длинами, соответствует их количеству в случайной последовательности. Если вычисленное в ходе теста значение $r < 0{,}01$, то данная двоичная последовательность не является истинно случайной. В противном случае она носит случайный характер.

Тест №4 - Тест на самую длинную последовательность единиц в блоке.

В данном тесте определяется самый длинный ряд единиц внутри блока длиной G бит. Цель — выяснить действительно ли длина такого ряда соответствует ожиданиям длины самого протяжённого ряда единиц в случае абсолютно случайной последовательности. Если высчитанное в ходе теста значение $r < 0{,}01$ полагается, что исходная последовательность не является случайной. В противном случае делается вывод о её случайности.

Тест №5 - Тест рангов бинарных матриц.

Здесь производится расчёт рангов непересекающихся подматриц, построенных из исходной двоичной последовательности. Целью этого теста является проверка на линейную зависимость подстрок фиксированной длины, составляющих первоначальную последовательность. В случае если вычисленное в

ходе теста значение $r < 0{,}01$, делается вывод о неслучайном характере входной последовательности бит. В противном случае считаем её абсолютно случайной.

Тест №6 - Спектральный тест.

Суть теста заключается в оценке высоты пиков дискретного преобразования Фурье исходной последовательности. Цель — выявление периодических свойств входной последовательности, например, близко расположенных друг к другу повторяющихся участков. Тем самым это явно демонстрирует отклонения от случайного характера исследуемой последовательности. Идея состоит в том, чтобы число пиков, превышающих пороговое значение в 95% по амплитуде, было значительно больше 5%. Если вычисленное в ходе теста значение $r < 0{,}01$, то данная двоичная последовательность не является абсолютно случайной. В противном случае она носит случайный характер.

Тест №7 - Тест на совпадение неперекрывающихся шаблонов.

В данном тесте подсчитывается количество заранее определенных шаблонов, найденных в исходной последовательности. Цель — выявить генераторы случайных или псевдослучайных чисел, формирующие слишком часто заданные непериодические шаблоны. Вычисленное в ходе теста значение r должно быть не меньше 0,01. В противном случае ($r < 0{,}01$), двоичная последовательность не является абсолютно случайной.

Тест №8 - Тест на совпадение перекрывающихся шаблонов.

Суть данного теста заключается в подсчете количества заранее определенных шаблонов, найденных в исходной последовательности. Вычисленное в ходе теста значение r должно быть не меньше 0,01. В противном случае ($r < 0{,}01$), двоичная последовательность не является абсолютно случайной.

Тест №9 - Универсальный статистический тест Маурера.

Здесь определяется число бит между одинаковыми шаблонами в исходной последовательности (мера, имеющая непосредственное отношение к длине сжатой последовательности). Цель теста — выяснить может ли данная последовательность быть значительно сжата без потерь информации. В случае если это возможно сделать, то она не является истинно случайной. В ходе теста вычисляется значение

r. Если $r < 0{,}01$, то полагается, что исходная последовательность не является случайной. В противном случае делается вывод о её случайности.

Тест №10 - Тест на линейную сложность.

В основе теста лежит принцип работы линейного регистра сдвига с обратной связью (*LFSR*). Цель — выяснить является ли входная последовательность достаточно сложной для того, чтобы считаться абсолютно случайной. Абсолютно случайные последовательности характеризуются длинными линейными регистрами сдвига с обратной связью. Если же такой регистр слишком короткий, то предполагается, что последовательность не является в полной мере случайной. В ходе теста вычисляется значение *r*. Если $r < 0{,}01$, то полагается, что исходная последовательность не является случайной. В противном случае делается вывод о её случайности.

Тест №11 - Тест на периодичность.

Данный тест заключается в подсчете частоты всех возможных перекрываний шаблонов длины *G* бит на протяжении исходной последовательности битов. Целью является определение действительно ли количество появлений 2^G перекрывающихся шаблонов длиной *G* бит, приблизительно такое же как в случае абсолютно случайной входной последовательности бит. Если вычисленное в ходе теста значение $r < 0{,}01$, то данная двоичная последовательность не является абсолютно случайной. В противном случае, она носит случайный характер.

Тест №12 - Тест приблизительной энтропии.

Как и в тесте на периодичность в данном тесте акцент делается на подсчёте частоты всех возможных перекрываний шаблонов длины *G* бит на протяжении исходной последовательности битов. Цель теста — сравнить частоты перекрывания двух последовательных блоков исходной последовательности с длинами *G* и *G+1* с частотами перекрывания аналогичных блоков в абсолютно случайной последовательности. Вычисляемое в ходе теста значение *r* должно быть не меньше 0,01. В противном случае ($r < 0{,}01$), двоичная последовательность не является абсолютно случайной.

Тест №13 - Тест кумулятивных сумм.

Тест заключается в максимальном отклонении (от нуля) при произвольном обходе, определяемым кумулятивной суммой заданных цифр в последовательности. Цель данного теста — определить является ли кумулятивная сумма частичных последовательностей, возникающих во входной последовательности, слишком большой или слишком маленькой по сравнению с ожидаемым поведением такой суммы для абсолютно случайной входной последовательности. Если вычисленное в ходе теста значение $r < 0{,}01$, то входная двоичная последовательность не является абсолютно случайной. В противном случае она носит случайный характер.

Тест №14 - Тест на произвольные отклонения.

Суть данного теста заключается в подсчёте числа циклов, имеющих строгое количество посещений при произвольном обходе кумулятивной суммы. Цель данного теста — определить отличается ли число посещений определенного состояния внутри цикла от аналогичного числа в случае абсолютно случайной входной последовательности. Принимается решение о степени случайности исходной последовательности в соответствии со следующим правилом: если вычисленное в ходе теста значение $r < 0{,}01$, то входная двоичная последовательность не является абсолютно случайной. В противном случае она носит случайный характер.

Тест №15 - Другой тест на произвольные отклонения.

В этом тесте подсчитывается общее число посещений определенного состояния при произвольном обходе кумулятивной суммы. Целью является определение отклонений от ожидаемого числа посещений различных состояний при произвольном обходе. Если вычисленное в ходе теста значение $r < 0{,}01$, то входная двоичная последовательность не является абсолютно случайной. В противном случае она носит случайный характер.

Для проверки стойкости к статистическим атакам на вход функции *nist_testing* (*encrypted_text, result_1 ... result_15*) были переданы следующие три

вида зашифрованных текстов и получены следующие результаты (таблица 3.6 и рисунки 3.7-3.8):

1. Зашифрованный текст №1: получен с помощью *CBC*-блочного шифрования открытого текста размером 2 МБ, состоящего из одного и того же символа.

2. Зашифрованный текст №2: получен с помощью *CBC*-блочного шифрования открытого текста размером 9 МБ, состоящего из различных символов.

3. Зашифрованный текст №3: получен с помощью *CBC*-блочного шифрования открытого текста размером 1 МБ, полученный с выхода генератора псевдослучайны чисел.

Таблица 3.6 – Результаты *NIST* – тестирования предложенной криптосистемы

Название теста	Полученное значение r / требуемое			Значение ω (результат теста)
	Текст №1	Текст №2	Текст №3	
Частотный побитовый тест	0,48 / 0,01	0,49 /0,01	0,49 / 0,01	1 (пройден)
Частотный блочный тест	0,13 / 0,01	0,17 / 0,01	0,15 / 0,01	1 (пройден)
Тест на последовательность одинаковых битов	0,67 / 0,01	0,64 / 0,01	0,61 / 0,01	1 (пройден)
Тест на самую длинную последовательность единиц в блоке	0,70 / 0,01	0,73 / 0,01	0,76 / 0,01	1 (пройден)
Тест рангов бинарных матриц	0,24 / 0,01	0,25 / 0,01	0,27 / 0,01	1 (пройден)
Спектральный тест	0,21 / 0,01	0,19 / 0,01	0,22 / 0,01	1 (пройден)
Тест на совпадение неперекрывающихся шаблонов	0,05 / 0,01	0,05 / 0,01	0,06 / 0,01	1 (пройден)
Тест на совпадение перекрывающихся шаблонов	0,09 / 0,01	0,07 / 0,01	0,09 / 0,01	1 (пройден)
Универсальный статистический тест Маурера	0,15 / 0,01	0,16 / 0,01	0,13 / 0,01	1 (пройден)
Тест на линейную сложность	0,37 / 0,01	0,34 / 0,01	0,36 / 0,01	1 (пройден)
Тест на периодичность	0,25 / 0,01	0,21 / 0,01	0,23 / 0,01	1 (пройден)
Тест приблизительной энтропии	0,49 / 0,01	0,46 / 0,01	0,50 / 0,01	1 (пройден)
Тест кумулятивных сумм	0,21 / 0,01	0,21 / 0,01	0,23 / 0,01	1 (пройден)
Тест на произвольные отклонения	0,09 / 0,01	0,07 / 0,01	0,11 / 0,01	1 (пройден)
Другой тест на произвольные отклонения	0,04 / 0,01	0,04 / 0,01	0,05 / 0,01	1 (пройден)

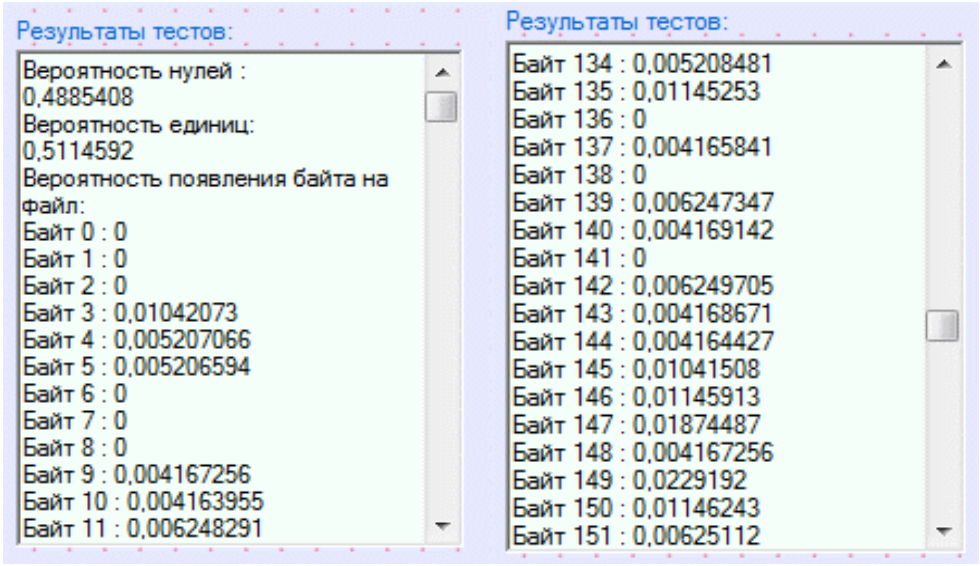

Рисунок 3.7 – Пример результатов *NIST*-тестов №1 и №3 для текста №2

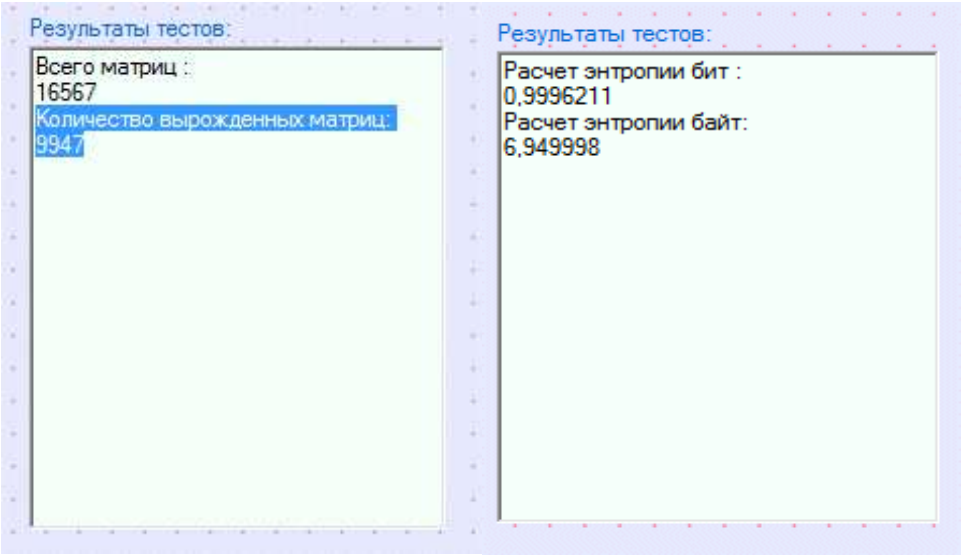

Рисунок 3.8 – Пример результатов *NIST*-тестов №5 и №12 для текста №2

Итог статистического исследования позволяет сделать следующие выводы:

1. Предложенная *CBC*-блочная криптосистема показала свои отличные результаты в прохождении *NIST* – тестов в соответствии с требуемой методикой. Во всех случаях зашифрованные тексты успешно прошли статистические тесты, что говорит о стойкости криптосистемы к частотному анализу и типовым атакам.
2. Результаты тестов подтвердили тот факт, что последовательность, являющая результатом работы *CBC*-блочного шифра, похожа на случайную, что в свою очередь говорит о высоком уровне

криптостойкости предложенной системы с общей памятью как по сравнению с предыдущими версиями симметричных рюкзачных криптосистем, так и симметричных систем в общем.

3.3 Сравнительное скоростное тестирование разработанной симметричной рюкзачной криптосистемы с известными алгоритмами

Цель эксперимента – необходимо практически оценить скорости работы семейства *CBC*-блочных алгоритмов [8], предложенную криптосистему подвергнуть экспериментальному прогону по следующему направлению – тестирование на скорость работы шифрования и дешифрования, сравнение с соответствующими скоростями работы функции *DH_AES* в стандартном шифронаборе *TLS* и рюкзачных криптосистем на основе супервозрастающих базисов.

Для проведения данного эксперимента были отобраны 10 файлов (сжимаемых и несжимаемых), приведенных в таблице 3.7. Для каждого из трех типов алгоритмов и для каждого типа файла проведено по 30 контрольных экспериментов.

Таблица 3.7 – Исходная база документов для скоростного тестирования

№ п/п	Тип файла	Размер, *Mb*
1	**.txt*	0.2
2	**.txt*	20.0
3	**.bmp*	80.0
4	**.gif*	10.0
5	**.exe*	30.0
6	**.mp3*	50.0
7	**.mp4*	2000.0

Продолжение таблицы 3.7

№ п/п	Тип файла	Размер, *Mb*
8	**.html*	0.4
9	**.rar*	10.0
10	**.7z*	20.0

План проведения эксперимента:

Для каждого из сжимаемых и несжимаемых файлов (таблица 3.9) выполнить:

1. Зашифровать исходный текст с помощью алгоритмов рюкзачной криптосистемы с общей памятью: *encrypt_text* (*source_text, vector_K*). Зарегистрировать время шифрования открытого текста.
2. Дешифровать полученный текст: *decrypt_text* (*encrypted_text, vector_K*). Зарегистрировать время дешифрования закрытого текста.
3. Зашифровать исходный текст с помощью алгоритмов рюкзачной криптосистемы на супервозрастающих базисах: *s_growing_enc_dec* (*source_text, encrypted_text*). Зарегистрировать время шифрования открытого текста.
4. Дешифровать полученный текст: *s_growing_enc_dec* (*source_text, encrypted_text*). Зарегистрировать время дешифрования закрытого текста.
5. Зашифровать исходный текст с помощью алгоритмов функции *DH_AES* из стандартного шифронабора *TLS*: *dh_aes_tls_enc_dec* (*source_text, encrypted_text*). Зарегистрировать время шифрования открытого текста.
6. Дешифровать полученный текст: *dh_aes_tls_enc_dec* (*source_text, encrypted_text*). Зарегистрировать время дешифрования закрытого текста.

На основе полученных результатов построены сводные таблицы (таблицы 3.8-3.9) и сравнительные гистограммы (в процентах относительно худшего результата рисунки А.1-А.10 и рисунки 3.9-3.10).

Таблица 3.8 – Результаты эксперимента по шифрованию файлов

№ п/п	Тип файла	Рюкзак с общей памятью, сек	Рюкзак на супервозрастающих базисах, сек	Функция *DH_AES* в *TLS*, сек
1	**.txt* (0,2 МБ)	1,2	1,4	1,2
2	**.txt* (20,0 МБ)	16,0	22,0	17,0
3	**.bmp* (80,0 МБ)	31,0	40,0	62,0
4	**.gif* (10,0 МБ)	7,0	9,0	9,0
5	**.exe* (30,0 МБ)	21,0	26,0	20,0
6	**.mp3* (50,0 МБ)	27,0	31,0	27,0
7	**.mp4* (2000,0 МБ)	95,0	147,0	99,0
8	**.html* (0,4 МБ)	1,0	3,0	1,0
9	**.rar* (10,0 МБ)	8,0	11,0	8,0
10	**.7z* (20,0 МБ)	13,0	20,0	15,0

Таблица 3.9 – Результаты эксперимента по дешифрованию файлов

№ п/п	Тип файла	Рюкзак с общей памятью, сек	Рюкзак на супервозрастающих базисах, сек	Функция *DH_AES* в *TLS*, сек
1	**.txt* (0,2 МБ)	1,7	2,0	1,7
2	**.txt* (20,0 МБ)	23,2	31,9	24,7
3	**.bmp* (80,0 МБ)	45,0	58,0	89,9
4	**.gif* (10,0 МБ)	10,2	13,1	13,1
5	**.exe* (30,0 МБ)	30,5	37,7	29,0
6	**.mp3* (50,0 МБ)	39,2	45,0	39,2
7	**.mp4* (2000,0 МБ)	137,8	213,2	143,6
8	**.html* (0,4 МБ)	1,5	4,4	1,5
9	**.rar* (10,0 МБ)	11,6	16,0	11,6
10	**.7z* (20,0 МБ)	18,9	29,0	21,8

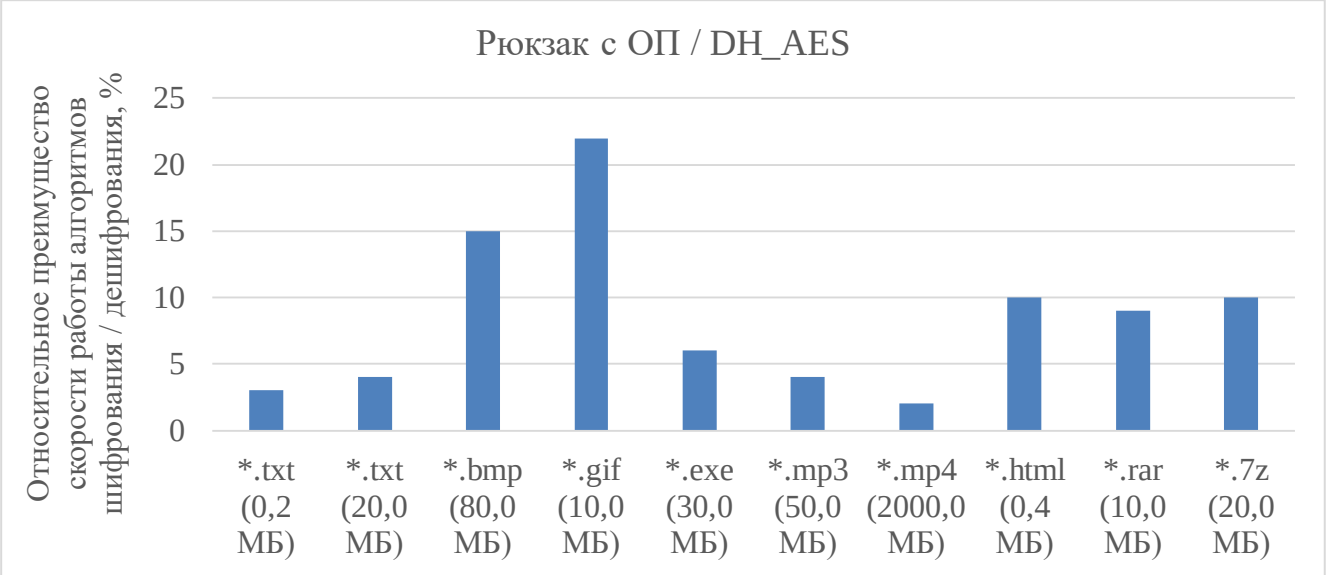

Рисунок 3.9 – Относительное преимущество скорости работы алгоритмов шифрования / дешифрования рюкзака с ОП к *DH_AES*

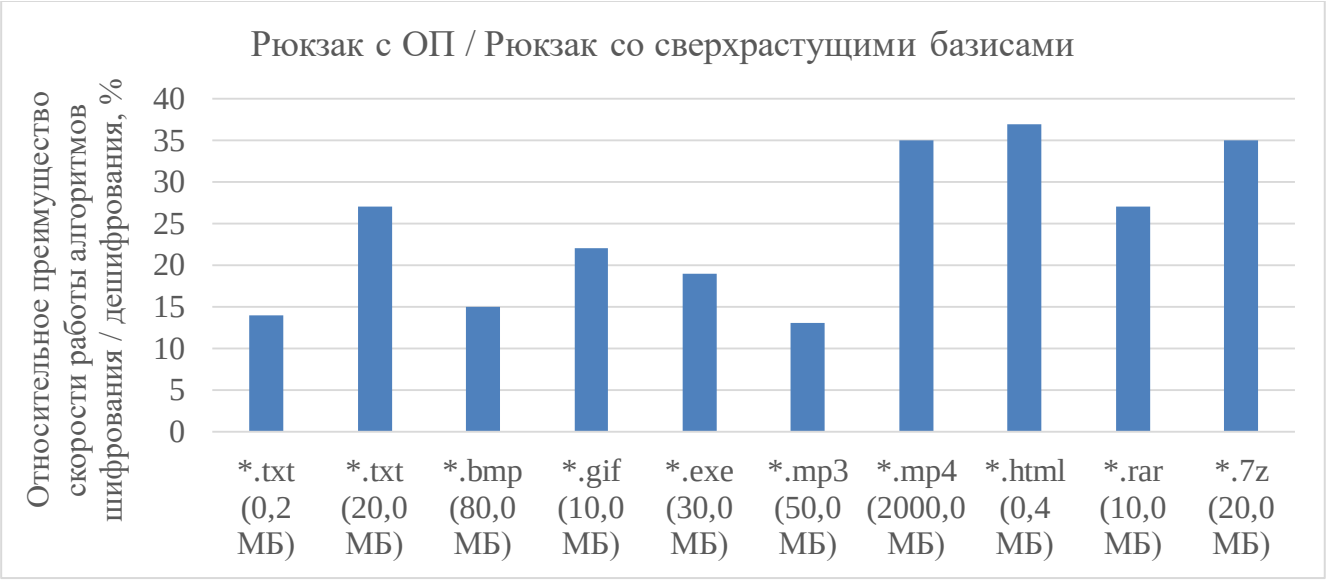

Рисунок 3.10 – Относительное преимущество скорости работы алгоритмов шифрования / дешифрования рюкзака с ОП к рюкзаку со сверхрастущими базисами

Итог сравнительного скоростного исследования позволяет сделать следующие выводы:

1. Спроектированное семейство *CBC*-блочных алгоритмов криптосистемы с общей памятью работает в среднем на 7-9% быстрее, чем функция

DH_AES в стандартном шифронаборе *TLS* и на 25-28% быстрее, чем рюкзачная криптосистема, в основе которой лежит супервозрастающий базис не зависимо от типа и объема исходных данных.

2. Полученные скоростные характеристики позволяют делать выводы о возможности применения разработанного семейства алгоритмов в композициях с другими стандартами, такими как *AES*, *DES* и *GOST1989* и для создания нового дополненного шифронабора в протоколе *TLS*. Также в частности, для встраивания этих алгоритмов в протоколы передачи данных *https*, где конструкцию шифрования / дешифрования *AES* можно соответственно заменить на композиции $AES(F_k(M))$ / $F_k^{-1}(AES^{-1}(S))$.

Выводы к главе 3

1. Предложенная *CBC*-блочная симметричная рюкзачная криптосистема с общей памятью обладает плотностью укладки не менее 0.9408, что в соответствии с рекомендациями Костера – Одлыжко [32, 50, 57] способствует затруднению реализации L^3-атаки на нее.

2. Разработанная криптосистема с общей памятью при минимальном размере рюкзачного базиса в G = 64 бита имеет плотность укладки в пределах $0.9907 \leq \rho \leq 0.9989$, что удовлетворяет требованиям необходимого уровня криптостойкости.

3. При размерах рюкзачного базиса в G = 128 бит, плотность укладки принимает значения $1.1910 \leq \rho \leq 1.1991$, при размерах рюкзачного базиса в G = 256 бит, плотность укладки принимает значения $1.4415 \leq \rho \leq 1.4429$. При увеличении размера G повышается криптостойкость разработанной системы. Выбор длины блока для передачи необходимо осуществлять в зависимости от важности и объема передаваемых сведений, но не менее чем G = 64 бита.

4. В соответствии с рекомендациями Костера – Одлыжко количественная оценка криптостойкости для всех предыдущих вариантов [32, 50, 57] рюкзачных криптосистем имеет значение W = (0.9408 * (1 – 0.9)) / 1,45 = 0.06, для предложенной криптосистемы с общей памятью минимальное значение W = 0,67,

что говорит о значительно более высоком уровне криптостойкости по отношению к L^3-атаке.

5. При величине плотности укладки в пределах $0.9907 \leq \rho \leq 0.9989$ вероятность того, что успешно реализуется L^3-атака, составляет 1 - 2 % от общего числа попыток реализовать атаку, при $1.1910 \leq \rho \leq 1.1991$ составляет 0.4 – 0.9 % и при $1.4415 \leq \rho \leq 1.4429$ составляет 0.1 – 0.2 % от общего числа попыток.

6. Между вероятностью успешной реализации L^3-атаки и плотностью укладки рюкзака при различных объемах исходного текста есть сильная статистическая зависимость (значение $|R| \sim 0.90$).

7. С возрастанием величины плотности укладки предложенного рюкзака уменьшается вероятность успешной реализации L^3-атаки в спроектированной *CBC*-блочной криптосистеме с общей памятью.

8. Предложенная *CBC*-блочная криптосистема показала свои отличные результаты в прохождении *NIST* – тестов в соответствии с требуемой методикой. Во всех случаях зашифрованные тексты успешно прошли статистические тесты, что говорит о стойкости криптосистемы к частотному анализу и типовым атакам.

9. Результаты тестов подтвердили тот факт, что последовательность, являющая результатом работы *CBC*-блочного шифра, похожа на случайную, что в свою очередь говорит о высоком уровне криптостойкости предложенной системы с общей памятью как по сравнению с предыдущими версиями симметричных рюкзачных криптосистем, так и симметричных систем в общем.

10. Спроектированное семейство *CBC*-блочных алгоритмов криптосистемы с общей памятью работает в среднем на 7-9% быстрее, чем функция *DH_AES* в стандартном шифронаборе *TLS* и на 25-28% быстрее, чем рюкзачная криптосистема, в основе которой лежит супервозрастающий базис не зависимо от типа и объема исходных данных.

11. Полученные скоростные характеристики позволяют делать выводы о возможности применения разработанного семейства алгоритмов в композициях с другими стандартами, такими как *AES*, *DES* и *GOST1989* и для создания нового дополненного шифронабора в протоколе *TLS*. Также в частности, для встраивания

этих алгоритмов в протоколы передачи данных *https*, где конструкцию шифрования / дешифрования AES можно соответственно заменить на композиции $AES(F_k(M))$ / $F_k^{-1}(AES^{-1}(S))$.

4 ПРАКТИЧЕСКАЯ РАЗРАБОТКА И ВНЕДРЕНИЕ СРЕДСТВ ОРГАНИЗАЦИИ ЗАЩИЩЕННОГО КАНАЛА СВЯЗИ

В главе предложена практическая реализация алгоритмов, приведенных в предыдущих главах по экспериментальному исследованию разработанных средств организации защищенного канала связи в телекоммуникационных сетях *TCP/IP*. Показаны их основные возможности и функционал.

4.1 Приложение для генерации из выбранного массива документов общей памяти

В состав реализованного ПО входит: приложение для генерации из выбранного массива документов общей памяти (*SM_generator*); основное приложение для шифрования и дешифрования сообщений с помощью разработанного семейства алгоритмов *CBC*-блочной симметричной рюкзачной криптосистемы с общей памятью и передачи выходных данных получателю (*Knapsack cryptosystem*).

В качестве одного из вариантов внедрения разработанных средств организации защищённого КС разработано клиент-серверное приложение для защиты авторского медиаконтента на основе семейства *CBC*-блочных алгоритмов симметричной рюкзачной криптосистемы с общей памятью (*media_knapsack_system*) в рамках работы по выигранному гранту (конкурсу) УМНИК-2015.

Все программное обеспечение реализовано с помощью средств свободно распространяемых сред программирования *Embarcadero rad studio XE* и *Google Android SDK*.

Основные возможности приложения для генерации из выбранного массива документов общей памяти (рисунок 4.1):

- Функция *clue_docs* (*source_docs [ ]*): склейка двух или более исходных документов отправителя в один для генерации на его основе общей памяти $O = \{o_1, o_2, \ldots, o_B\}$ между отправителем и получателем.
- Функция *generate_SM* (*clue_docs [ ]*): формирование общей памяти – последовательности $O = \{o_1, o_2, \ldots, o_B\}$ из выбранного (склеенного) массива документов отправителя.

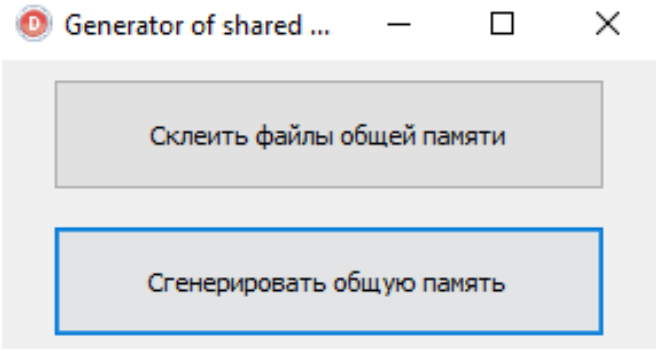

Рисунок 4.1 – Основой интерфейс ПО для генерации общей памяти

Основной принцип работы приложения для генерации из выбранного массива документов общей памяти:

- По кнопке «Склеить файлы общей памяти» ПО предлагает пользователю выбран два или более файлов (рисунок 4.2) любого вида (любого расширения) и на его основе получить склеенный файл (рисунок 4.3) формата **.clu* для дальнейшей генерации последовательности $O = \{o_1, o_2, \ldots, o_B\}$.
- По кнопке «Сгенерировать общую память» ПО предлагает пользователю выбран файл или склейку файлов (рисунок 4.4) на основе которой, в соответствии с алгоритмом шифрования, показанного в п.2.4 формируется массив общей памяти в виде файла (рисунок 4.5) с расширением **.shm*.

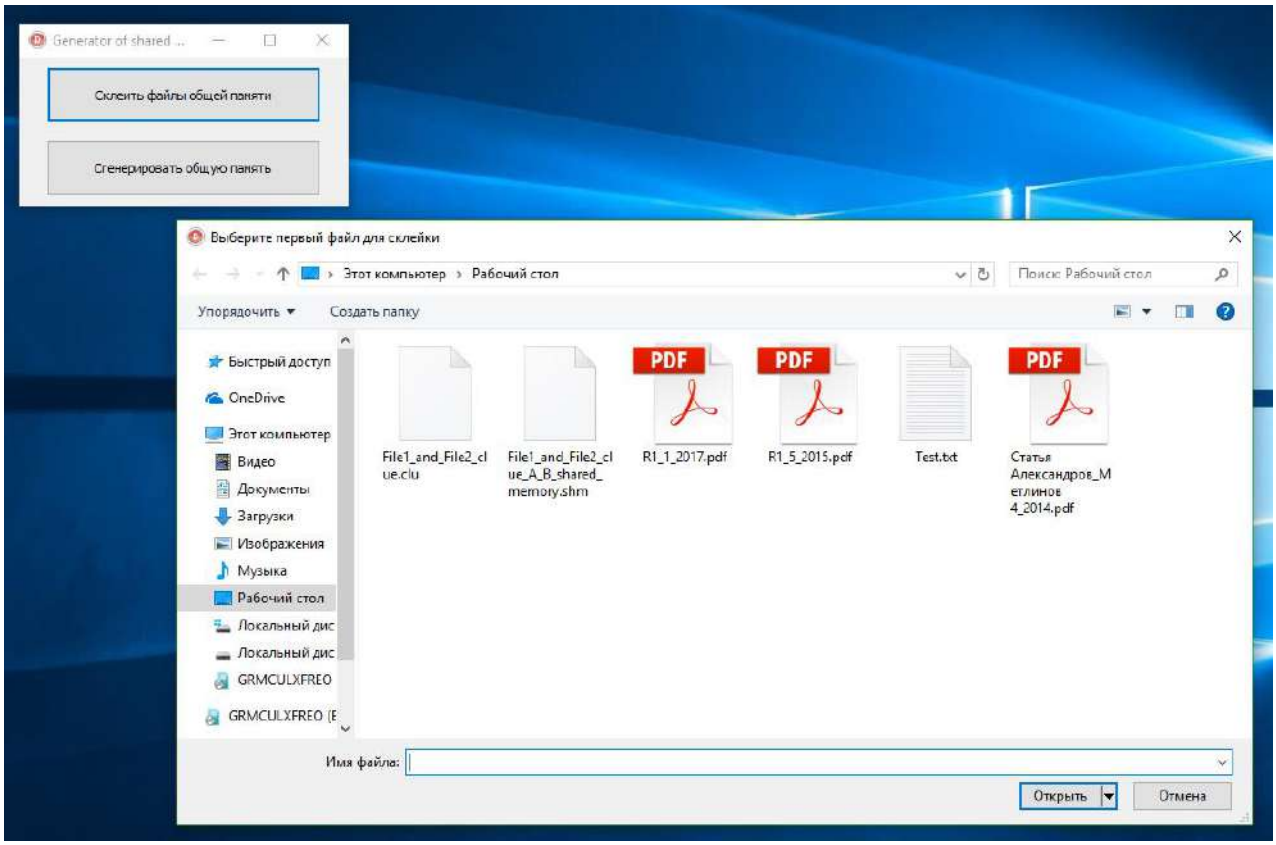

Рисунок 4.2 – Диалог выбора файлов для склейки

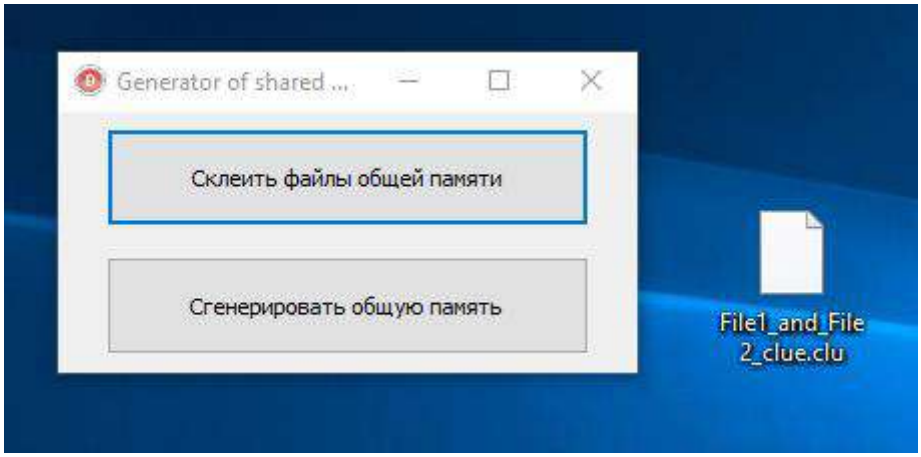

Рисунок 4.3 – Полученный файл *.*clu* для генерации общей памяти

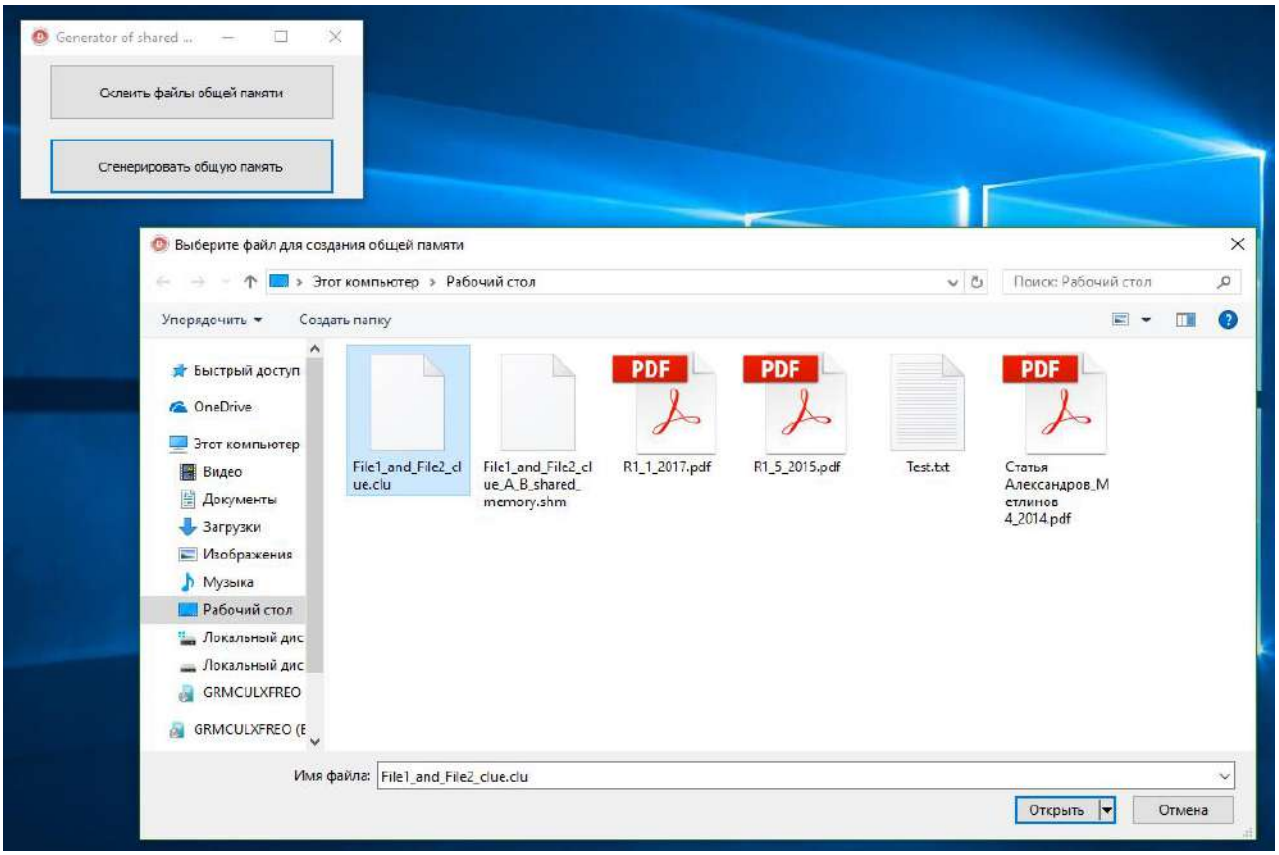

Рисунок 4.4 – Диалог выбора исходных файлов для генерации общей памяти

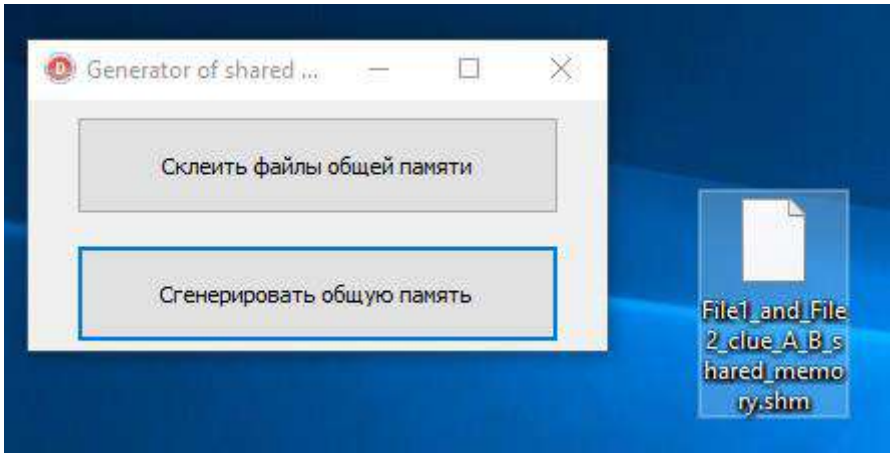

Рисунок 4.5 – Полученный массив общей памяти в виде файл *.*shm*

4.2 Приложение для шифрования и дешифрования сообщений

Основные возможности приложения для шифрования и дешифрования сообщений (рисунок 4.6):

- Функция *establish_connection* (*dest_IP, dest_Port*): установка соединения по КС между отправителем (*Sender*) и получателем (*Receiver*), на вход функции поступают *IP*-адрес и порт ПК получателя.
- Функция *get_SM* (*docs[ ]*): получение общей памяти – последовательности $O = \{o_1, o_2, \ldots, o_B\}$ из заранее подготовленного приложением (*SM_generator*) массива документов.
- Функция *generate_vector_E* (*rand_seed*): генерация вектора $E = \{e_1, e_2, \ldots, e_B\}$ в соответствии с алгоритмом шифрования, показанного в п.2.4.
- Функция *calculate_sum* (*shared_memory, vector_E*): вычисление всех возможных сумм $\{\Psi\}$ произведений неповторяющихся элементов общей памяти и элементов вектора *E* в соответствии с алгоритмом шифрования, показанного в п.2.4.
- Функция *sort_sum* (*vector_sum*): сортировка последовательности всех возможных сумм $\{\Psi\}$ по возрастанию в соответствии с алгоритмом шифрования, показанного в п.2.4.
- Функция *choose_element_sum* (*sum_0*): выбор одной суммы Ψ_0 как серединный элемент отсортированной последовательности в соответствии с алгоритмом шифрования, показанного в п.2.4.
- Функция *generate_vector_K* (*sum_0, vector_E*): формирование рюкзачной последовательности $\{k_1(\Psi_0),\ k_2(\Psi_0)\ , \ldots, k_g(\Psi_0)\}$ в соответствии с алгоритмом шифрования, показанного в п.2.4 и вектором *E*.
- Функция *encrypt_text* (*source_text, vector_K*): шифрование открытого текста *M* разбитого на блоки длины *G* в соответствии с алгоритмом *CBC*-блочного шифрования, показанного в п.2.4 и рюкзачным вектором $\{k_1(\Psi_0),\ k_2(\Psi_0)\ , \ldots, k_g(\Psi_0)\}$.
- Функция *calculate_density* (*vector_K*): подсчет плотности укладки (2.3) полученной криптосистемы с общей памятью на основе последовательности $\{k_1(\Psi_0),\ k_2(\Psi_0)\ , \ldots, k_g(\Psi_0)\}$.

- Функция *decrypt_text* (*encrypted_text*, *vector_K*): дешифрование закрытого текста *S* разбитого на блоки длины *G* в соответствии с алгоритмом *CBC*-блочного дешифрования, показанного в п.2.4 и рюкзачным вектором $\{k_1(\Psi_0),\ k_2(\Psi_0)\ , \dots , k_g(\Psi_0)\}$.
- Функция *calculate_time* (*alg_enc*, *alg_dec*): подсчет и вывод времени работы алгоритмов шифрования и дешифрования в соответствии с разработанным семейством алгоритмов *CBC*-блочного шифрования и дешифрования, показанных в п.2.4.

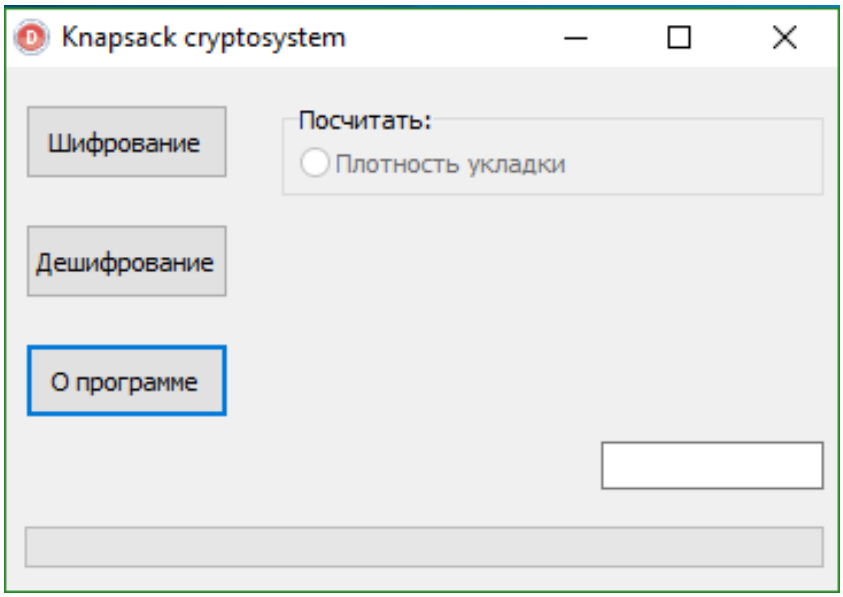

Рисунок 4.6 – Основой интерфейс ПО для шифрования и дешифрования

Основной принцип работы приложения для шифрования и дешифрования сообщений:

- По кнопке «Шифрование» ПО предлагает пользователю ввести *IP*-адрес и порт получателя, на ПК которого запущено аналогичное приложение. Затем просит выбрать файл (открытый текст) для шифрования (рисунок 4.7), выбрать общую память (рисунок 4.8) и преобразует исходный текст в соответствии с алгоритмом *CBC*-блочного шифрования, показанного в п.2.4, получая зашифрованный файл формата **.cpt*, ключевой вектор *E* в виде файла *key.kpt* и все отправляет получателю (рисунок 4.9).

- По кнопке «Дешифрование» ПО предлагает пользователю выбрать зашифрованный файл (рисунок 4.10), который необходимо расшифровать, затем файл общей памяти (рисунок 4.11) и ключевой файл вектора *E* (рисунок 4.12). ПО преобразует закрытый текст в соответствии с алгоритмом *CBC*-блочного дешифрования, показанного в п.2.4, получая файл исходного открытого текста (рисунок 4.13).
- По кнопке «Посчитать плотность укладки» выводится значение плотности укладки для сформированного рюкзачного вектора (рисунок 4.14) и время работы алгоритмов шифрования и дешифрования.

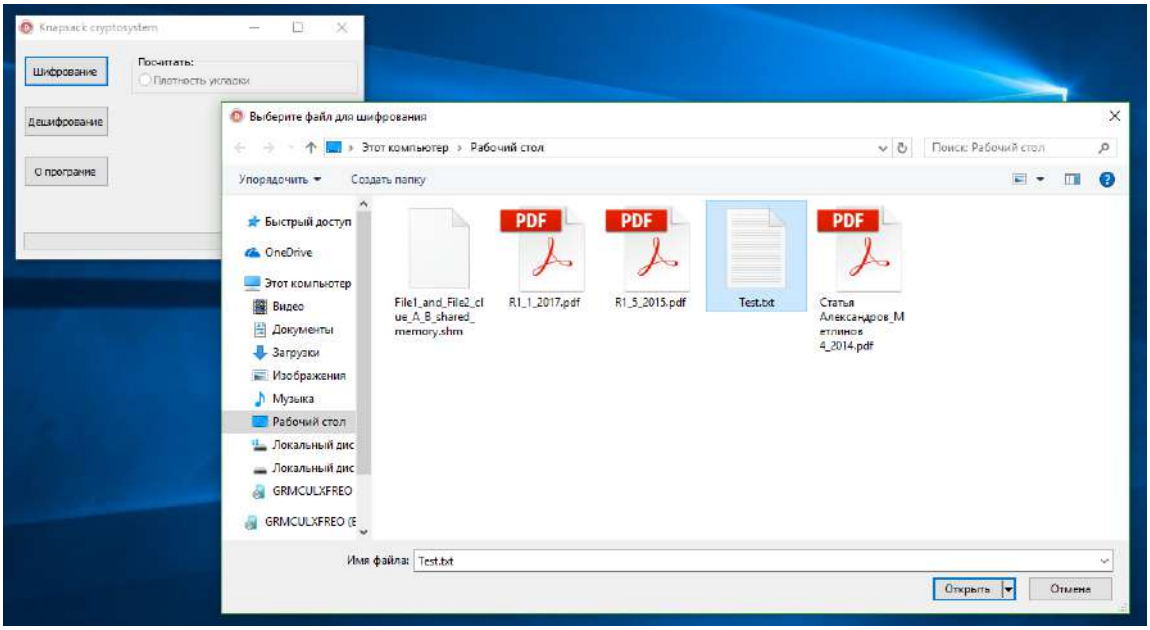

Рисунок 4.7 – Диалог выбора файла для шифрования

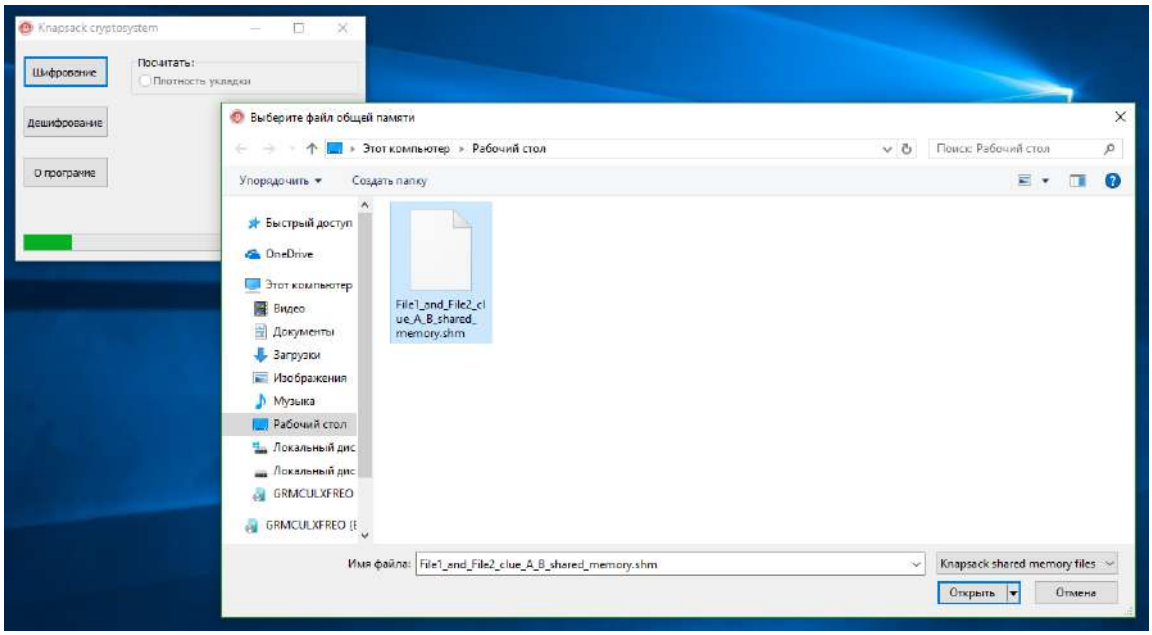

Рисунок 4.8 – Диалог выбора файла общей памяти

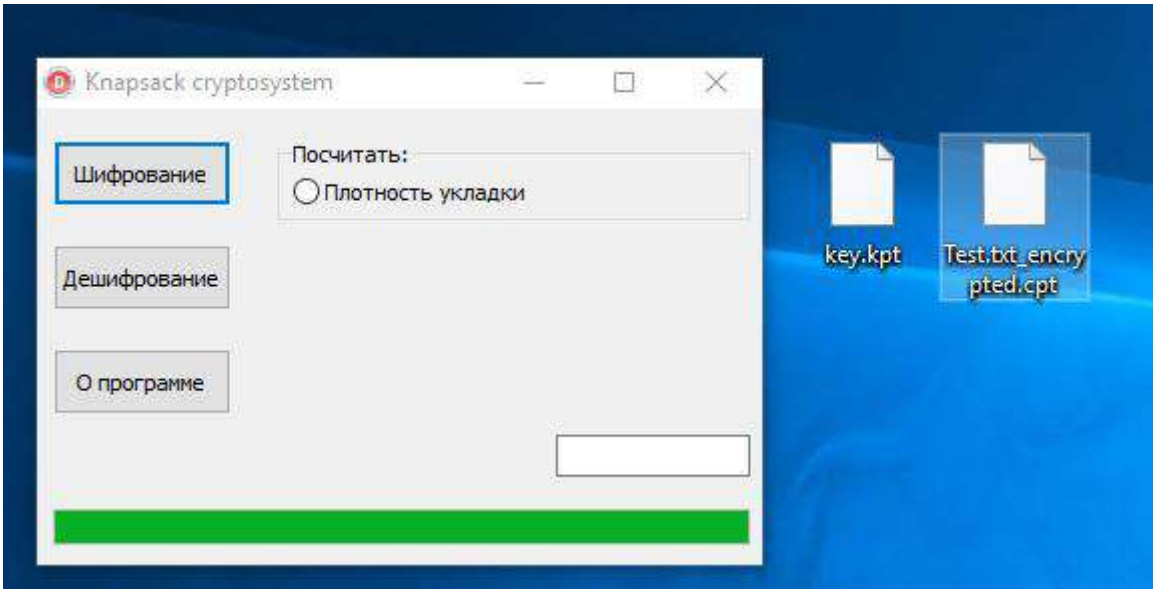

Рисунок 4.9 – Полученные зашифрованный и ключевой файлы

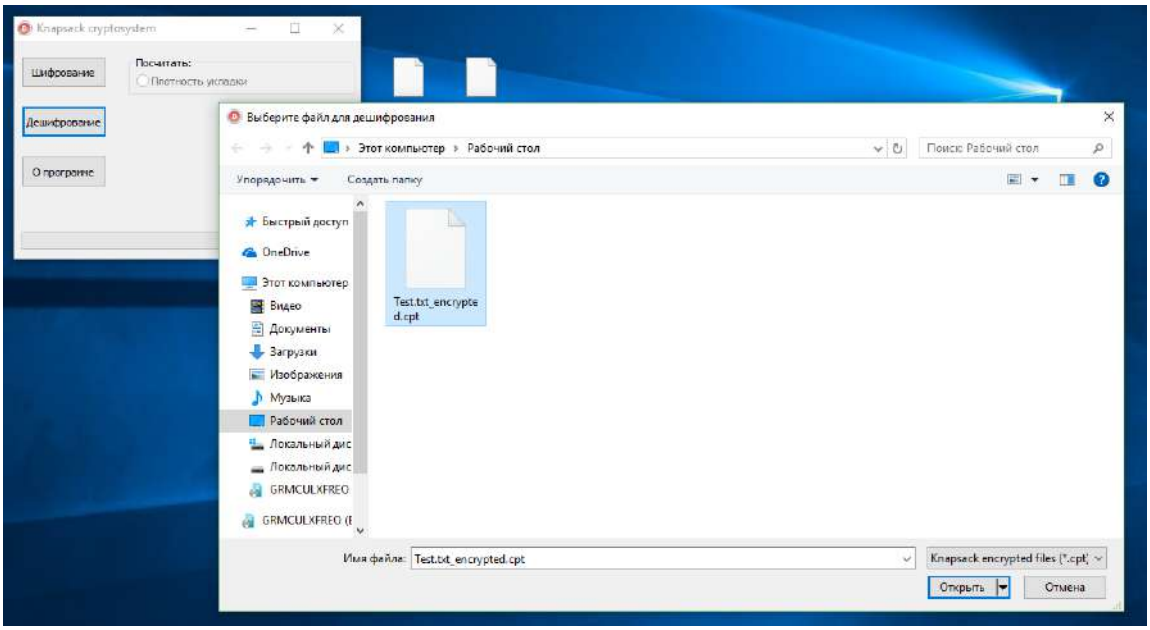

Рисунок 4.10 – Диалог выбора файла для дешифрования

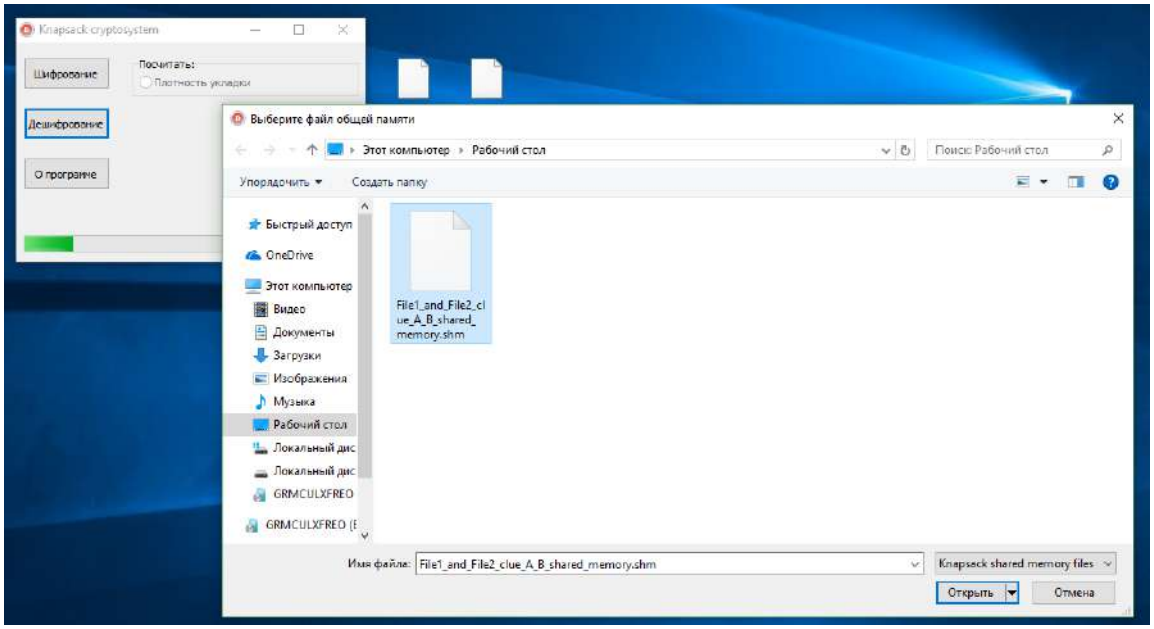

Рисунок 4.11 – Диалог выбора файла общей памяти для дешифрования

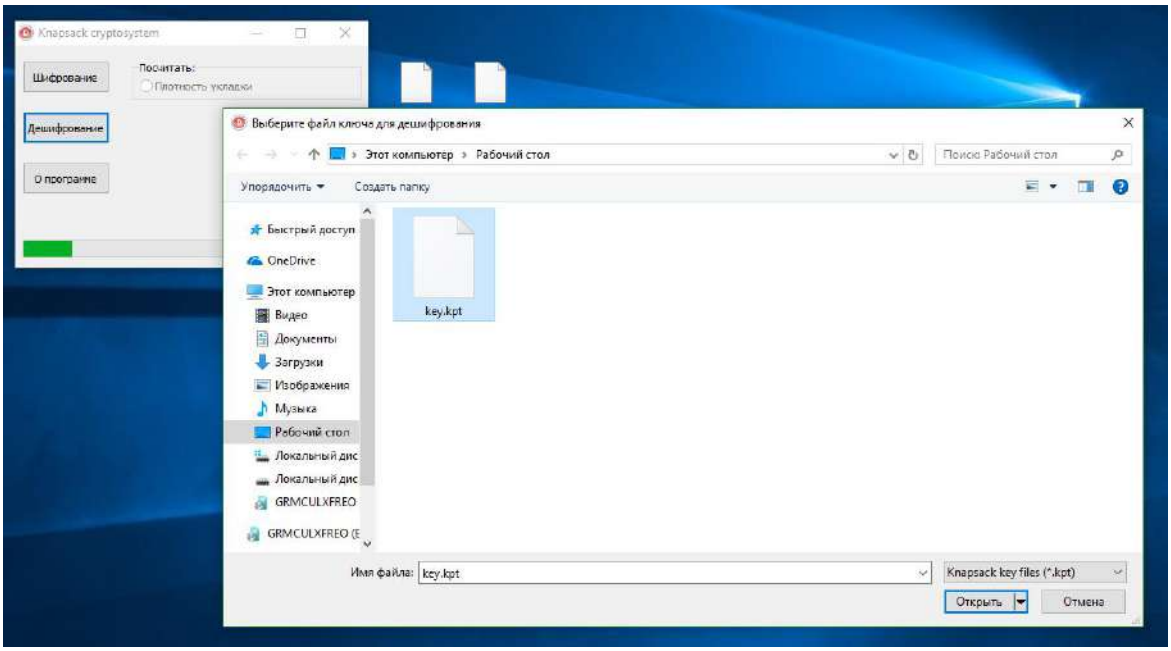

Рисунок 4.12 – Диалог выбора ключевого файла для дешифрования

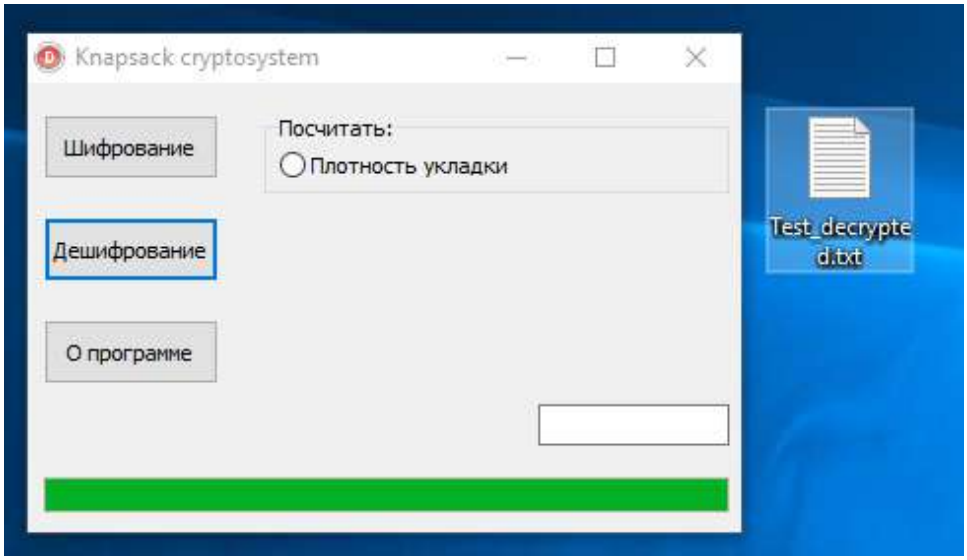

Рисунок 4.13 – Полученный дешифрованный (исходный) файл

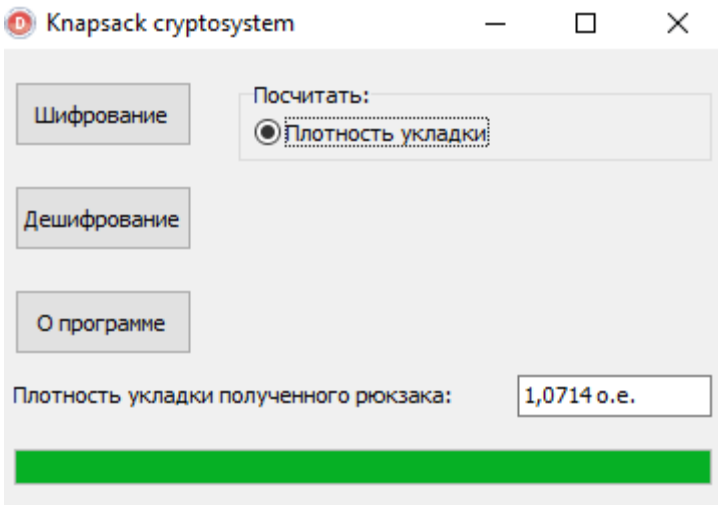

Рисунок 4.14 – Расчет плотности укладки полученного рюкзачного вектора

Исходные листинги кода данных программных обеспечений с комментариями представлены в приложениях Г и Д.

4.3 Клиент-серверное приложение для защиты авторского медиаконтента

Основные возможности клиент-серверного приложения для защиты авторского медиаконтента:

- Функция *establish_connection* (*server_IP*, *server_Port*, *API_key*): установка соединения с сервером, где хранится купленный авторский медиаконтент в зашифрованном виде и авторизация пользователя.
- Функция *get_list_content* (*source_items [ ]*): получение списка доступного (купленного) для данного авторизованного пользователя авторского медиаконтента.
- Функция *get_item* (*source_items [ ]*): потоковое получение конкретного файла авторского медиаконтента, его дешифровка на устройстве пользователя и открытие (воспроизведение).

Приложение состоит из двух частей: серверная часть (рисунки 4.15 – 4.16), разворачивается и работает у автора (владельца) медиаконтента и клиентская часть (рисунки 4.17 – 4.20), поставляется пользователю, который приобрел лицензию на данный просмотр.

Основной функционал и особенности серверной части:

- регистрация, модификация и удаление пользователей, которые приобрели лицензию на просмотр конкретного медиаконтента;
- генерация данных аутентификации (*API-key*) пользователей, для их авторизации в клиентской части приложения;
- привязка доступного медиаконтента к конкретному пользователю;
- внесение, модификация и удаление медиаконтента с сервера;
- весь медиаконтент на сервере находится в зашифрованном состоянии.

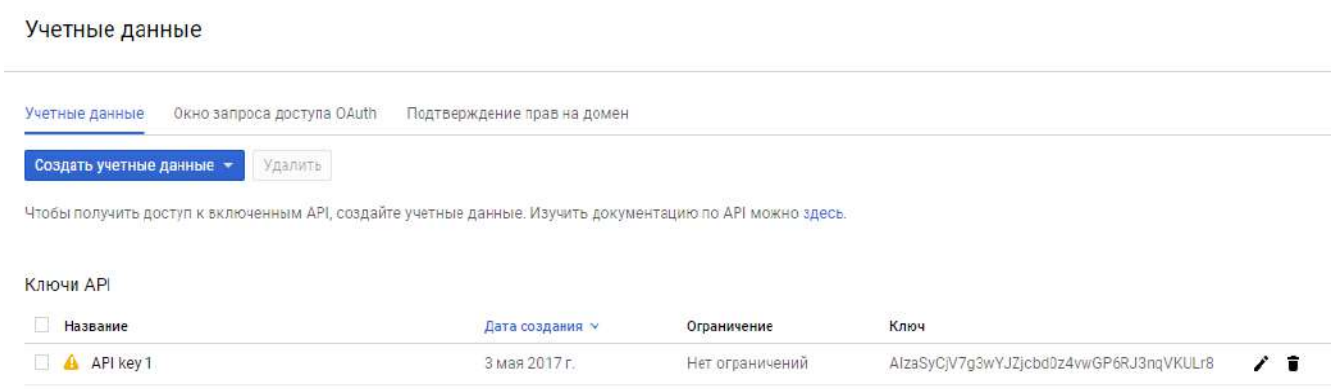

Рисунок 4.15 – Работа с учетными записями пользователей

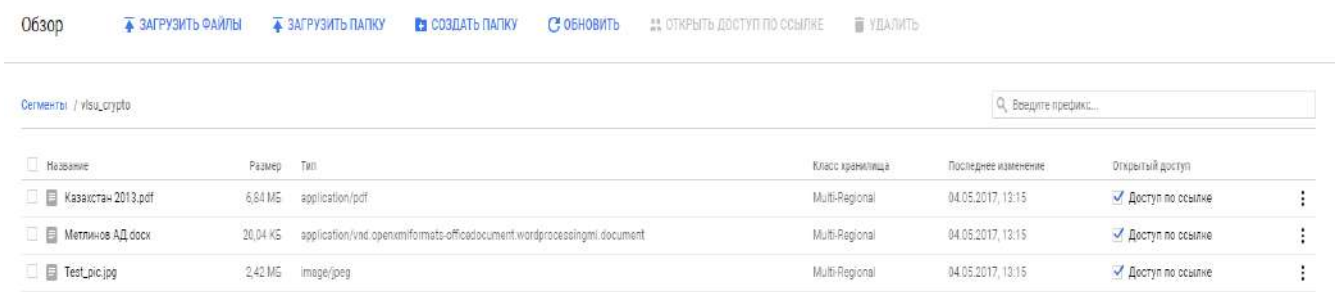

Рисунок 4.16 – Работа с доступным медиаконтентом для пользователя

Основной функционал и особенности клиентской части:

- регистрация и авторизация пользователя в приложении для конкретного поставщика авторского медиаконтента;
- получение данных аутентификации (*API-key*), для авторизации в клиентской части приложения;
- просмотр доступного медиаконтента, его загрузка и воспроизведение на устройстве конечного пользователя;
- -весь медиаконтент на клиентской части дешифруется с помощью разработанных алгоритмов симметричной рюкзачной криптосистемы с общей памятью.

Рисунок 4.17 – Страница авторизации пользователя

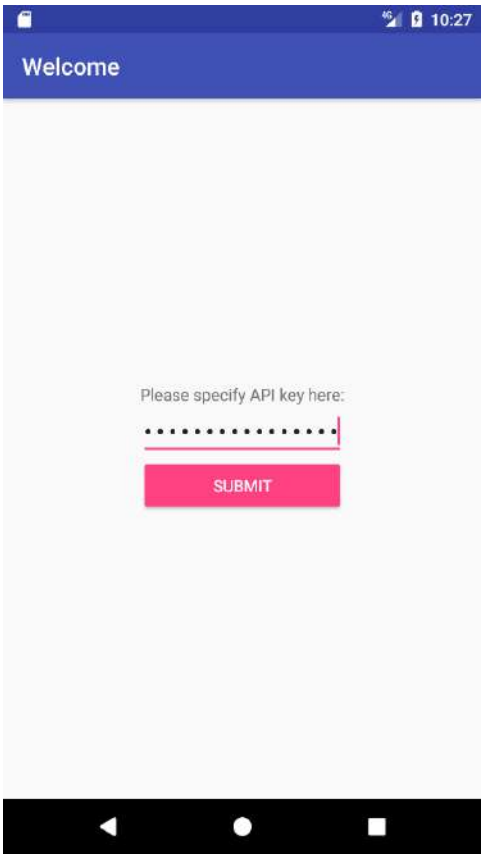

Рисунок 4.18 – Ввод полученных данных авторизации

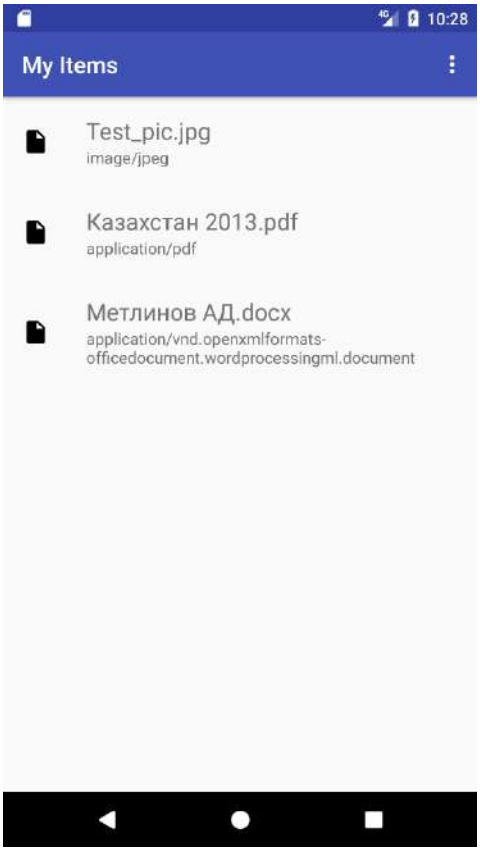

Рисунок 4.19 – Получение списка доступного зашифрованного медиаконтента

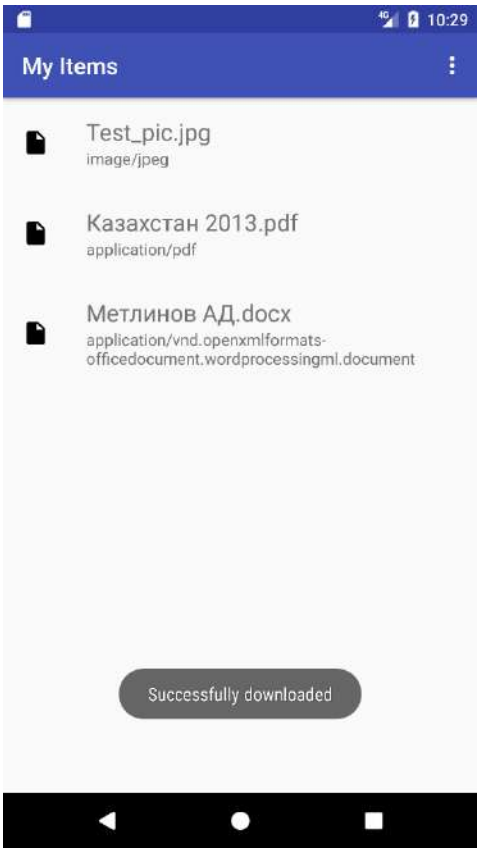

Рисунок 4.20 – Загрузка и дешифровка доступного медиаконтента

4.4 Внедрение, лицензирование и коммерциализация

Разработано информационное и программное обеспечение комплекса *CBC*-блочной симметричной рюкзачной криптосистемы с общей памятью, включающее: программный комплекс *CBC*-блочной симметричной рюкзачной криптологической системы для вариации с плотностью укладки больше единицы при G = 128 бит (свидетельство о государственной регистрации программы для ЭВМ №2014614981); программный тестовый комплекс для симметричной рюкзачной криптосистемы (свидетельство о государственной регистрации программы для ЭВМ №2014614937); программный модуль генератора общей памяти для симметричной рюкзачной криптосистемы (свидетельство о государственной регистрации программы для ЭВМ №2015616165). Таким образом на все разработанные алгоритмы получены необходимые свидетельства о регистрации, представленные в приложении Е.

Результаты работы данного диссертационного исследования внедрены на следующих предприятиях:

- ООО «Русский мастер», Владимирская область;
- ООО «Дивания», Владимирская область;
- ИП Щерба А.Ю., город Владимир.

В ООО «Русский мастер» апробированы и используются следующие результаты диссертационного исследования:

1. Алгоритмы шифрования, дешифрования, хеширования конфиденциальной информации, а также алгоритмы генерации и передачи ключевой информации.
2. Реализованная на основе предложенной криптосистемы с общей памятью система безопасного хранилища и передачи по открытым каналам связи конфиденциальной информации *Trans Kept&Sec* 1.3.
3. Авторские методики тестирования полученной защищенной среды передачи конфиденциальной информации: скоростная методика оценки работы защищенной среды; оценки объема общей памяти, минимально необходимого для ухода от L^3-атаки и т.д.

Применение вышеописанных результатов в ООО «Русский мастер» позволило безопасно хранить данные предприятия, которые составляют его коммерческую тайну, а также безопасно их передавать между подразделениями, не опасаясь их хищения; снизить материальные затраты на активные средства по защите этих данных, облегчить и ускорить работу подразделений предприятия.

В ООО «ДИВАНиЯ» апробированы и используются следующие результаты диссертационного исследования:

1. Протокол *TLS* с модифицированными алгоритмами аутентификации пользователей (удаленных узлов) и сервера, шифрования, дешифрования, хеширования информационным ресурсов предприятия.
2. Реализованная на основе предложенной модификации протокола *TLS* система безопасного удаленного доступа к ценным информационным ресурсам предприятия *Divaniya_SecAccess* 1.2.

Применение вышеописанных результатов в ООО «ДИВАНиЯ» позволило организовать безопасный удаленный доступ и переду ценных информационных ресурсов предприятия между подразделениями; снизить до минимума материальные затраты на активные средства по защите этих данных, ускорить работу с этими данными в 3 раза. Снизить количество инцидентов по взлому серверов с этими данными на 95%, ускорить работу удаленных пользователей с серверами в 2 раза, время на аутентификацию каждого пользователя сокращено с минуты до десяти секунд.

В ИП Щерба А.Ю апробированы и используются следующие результаты диссертационного исследования:

1. Алгоритмы шифрования, дешифрования, хеширования конфиденциальной информации, составляющей коммерческую тайну ИП.
2. Реализованная на основе предложенной криптосистемы с *pre-shared memory* система безопасного хранилища конфиденциальной информации *Trans Kept&Sec* 1.3.1 (*build_for_Tsherba*).

Применение вышеописанных результатов в ИП Щерба А.Ю позволило безопасно хранить данные индивидуального предпринимателя, которые составляют его коммерческую тайну и максимально снизить материальные затраты на активные средства по защите этих данных от внешних злоумышленников.

Данные программные комплексы внедрены на правах бесплатной ограниченной лицензии для рекламы и раскрутки приложения по защите данных, составлявших коммерческую тайну на малых и средних предприятиях, а также у индивидуальных предпринимателей. Внедрение этих приложений позволило практически до нуля сократить количество инцидентов по несанкционированному доступу к информации, составляющей коммерческую тайну и значительно сократить расходы по защите информации на предприятиях.

Копии актов внедрения результатов диссертационного исследования на данных предприятия представлены в приложении Е.

Выводы к главе 4

1. Практически реализовано семейство алгоритмов (шифрования, дешифрования и хеширования информации) для повышения производительности и криптостойкости симметричных рюкзачных криптосистем в защищенных КС сетей *TCP/IP*, которые отличаются от предыдущих версий рюкзачных криптосистем (симметричных систем в общем) наличием общей памяти, высоким уровнем плотности укладки рюкзака и отказом от супервозрастающих базисов в пользу линейно - рекуррентных последовательностей.

2. Аналогов разработанных программных продуктов на данный момент не найдено или вообще не существует. Подобная разработка представляется впервые. Основные преимущества разработанного ПО: высокая скорость работы, низкие требования к ресурсам пользовательского ПК, высокая криптостойкость системы, криптосистема прошла проверку на *NIST*-тесты, низкие требования к пропускной способности интернет - канала конечного пользователя (в большей степени требователен сам контент, имеющий большой объем).

3. Внедрение этих приложений на предприятия позволило практически до нуля сократить количество инцидентов по несанкционированному доступу к информации, составляющей их коммерческую тайну и значительно сократить расходы по защите информации на предприятиях.

ЗАКЛЮЧЕНИЕ

Основные результаты диссертационного исследования:

1. Каналы связи сетей *TCP/IP* не имеют встроенных средств защиты, механизмы обеспечения информационной безопасности реализованы на сеансовом уровне *OSI* и имеют множество уязвимостей. Основные угрозы и атаки в каналах связи направлены на перехват и имперсонацию сообщений – нарушение конфиденциальности и целостности передаваемых данных. Незначительное количество атак относится к атакам на доступность узлов канала и их подмену.

2. Перспективным подходом является использование криптостойкого шифрования *TCP/IP*-потока с использованием протоколов *SSL* и *TLS*, которые обеспечивая криптозащиту, имеют недостаточное быстродействие и относительно низкую криптостойкость. Для повышения быстродействия и криптостойкости КС перспективным может являться использование средств симметричной рюкзачной криптографии.

3. Разработано семейство алгоритмов симметричных рюкзачных криптосистем, которые отличаются наличием общей памяти между узлом-отправителем и узлом-получателем, высоким уровнем плотности укладки рюкзака, а также отказом от супервозрастающих базисов в пользу линейно - рекуррентных последовательностей.

4. Предложена модификация симметричной рюкзачной криптосистемы с общей памятью семейством *CBC*-блочных алгоритмов шифрования и дешифрования информации, что еще в большей мере повышает криптостойкость: при минимальном размере рюкзачного базиса в 64 бита плотность укладки рюкзака лежит в пределах $0.9907 \le \rho \le 0.9989$, что в соответствии с рекомендациями Костера – Одлыжко значительно затрудняет реализацию L^3-атаки (ее вероятность не более 1 - 2 %).

5. Экспериментально выявлено, что между вероятностью успешной реализации L^3-атаки и плотностью укладки рюкзака при различных объемах исходного текста есть сильная статистическая зависимость (значение $|R| \sim 0.90$).

6. Предложенная *CBC*-блочная криптосистема показала свои отличные результаты в прохождении *NIST* – тестов в соответствии с требуемой методикой. Во всех случаях зашифрованные тексты успешно прошли статистические тесты, что говорит о стойкости криптосистемы к частотному анализу и типовым атакам. Результаты тестов подтвердили тот факт, что последовательность, являющая результатом работы *CBC*-блочного шифра, похожа на случайную, что в свою очередь говорит о высоком уровне криптостойкости предложенной системы с общей памятью как по сравнению с предыдущими версиями симметричных рюкзачных криптосистем, так и симметричных систем в общем.

7. Внедрение разработанной криптосистемы в фазы работы стандартного протокола *TLS* (аутентификации клиента и сервера, создания кода аутентификации сообщений и работы симметричных блочных алгоритмов шифрования и дешифрования сообщений) позволяет существенно повысить эффективность защищенного КС сетей *TCP/IP*, канал работает в среднем на 7-9% быстрее, чем использующий *DH_AES* в стандартном *TLS* и на 25-28% быстрее, чем рюкзачная криптосистема, в основе которой лежит супервозрастающий базис независимо от типа и объема исходных данных.

8. Полученные скоростные характеристики позволяют делать выводы о возможности применения разработанного семейства алгоритмов в композициях с другими стандартами, такими как *AES*, *DES* и *GOST1989* и для создания нового дополненного шифронабора в протоколе *TLS*. Также в частности, для встраивания этих алгоритмов в протоколы передачи данных *https*, где конструкцию шифрования / дешифрования *AES* можно соответственно заменить на композиции $AES(F_k(M))$ / $F_k^{-1}(AES^{-1}(S))$.

9. Практически реализовано семейство алгоритмов (шифрования, дешифрования и хеширования информации) для повышения быстродействия и криптостойкости симметричных рюкзачных криптосистем в защищенных КС сетей *TCP/IP*. Аналогов разработанных программных продуктов на данный момент не найдено или вообще не существует. Подобная разработка представляется впервые. Основные преимущества разработанного ПО: высокая скорость работы, низкие

требования к ресурсам пользовательского ПК, высокая криптостойкость системы, криптосистема прошла проверку на *NIST*-тесты, низкие требования к пропускной способности интернет - канала конечного пользователя (в большей степени требователен сам контент, имеющий большой объем). Внедрение этих приложений на предприятия позволило практически до нуля сократить количество инцидентов по несанкционированному доступу к информации, составляющей их коммерческую тайну и значительно сократить расходы по защите информации на предприятиях.

СПИСОК ПРИНЯТЫХ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

DH – алгоритм Диффи-Хеллмана;

DH_AES – шифронабор протокола *TLS*, состоящий из алгоритма Диффи-Хеллмана и алгоритма *AES;*

IPS – производительность протокола;

NIST – национальный институт стандартов и технологий США;

SSL – уровень защищённых сокетов;

TLS – протокол защиты транспортного уровня;

VPN – виртуальная частная сеть;

TCP/IP – сетевая модель передачи данных, представленных в цифровом виде;

OSI – базовая эталонная модель взаимодействия открытых систем;

CBC – режим сцепления блоков шифротекста;

IETF – инженерный совет интернета;

MAC – код аутентификации сообщений;

HMAC – хеш-функция кода аутентификации сообщений;

MITM – атака «человек по середине»;

ЗИ – защита информации;

ИБ – информационная безопасность;

ОП – общая память;

ИР – информационные ресурсы;

ИС – информационная система;

КМХ – криптосистема Меркла-Хеллмана;

ТС – телекоммуникационная сеть;

КС – канал связи;

РС – рабочая станция;

СВТ – средства вычислительной техники;

ПК – персональный компьютер;

ПО – программное обеспечение;

ЭВМ – электронно-вычислительная машина;

ρ – плотность укладки рюкзака;

O – общая память между отправителем и получателем;

E – случайный двоичный вектор между отправителем и получателем;

m – порядок линейно - рекуррентных последовательностей;

M – исходное сообщение (отрытый текст);

k – криптографический (закрытый) ключ;

S – передаваемое (зашифрованное) сообщение;

F и F^{-1} – алгоритмы шифрования и дешифрования сообщений соответственно;

N – количество блоков в открытом сообщении;

G – количество бит в одном блоке;

B – количество документов общей памяти;

$\bigoplus$ – операция сложения в выбранном поле Галуа GF_p;

$\{\Psi\}$ – все возможные суммы произведений неповторяющихся элементов общей памяти и элементов вектора E;

H – хеш-функция;

W – количественная характеристика криптостойкости;

P - вероятность успешной реализации L^3-атаки в спроектированной криптосистеме;

ρ_{max} – предельная плотность ($\rho_{max} \approx 1{,}45$);

ω – показатель статистической защищенности;

r – полученное значение *NIST*-теста.

СПИСОК ОПРЕДЕЛЕНИЙ

В настоящем диссертационном исследовании применяются следующие термины с соответствующими определениями:

- $\boldsymbol{L^3}$**–атака –** это полиноминальная по сложности атака (в основе которой лежит алгоритм Грама-Шмидта) на рюкзачные системы, успешно взламывающей их с плотностью укладки менее 0.9408.
- **Атака Шамира –** это атака на пару «открытый / закрытый ключ», с помощью которой в 1982 году Ади Шамир раскрыл рюкзачную систему Меркла-Хеллмана, базирующуюся на асимметричном шифровании с лазейкой.
- **Асимметричные ключи** – это ключи, используемые в асимметричных алгоритмах (шифрование, ЭЦП); являются ключевой парой, поскольку состоят из двух ключей. Закрытый ключ — ключ, известный только своему владельцу. Только сохранение пользователем в тайне своего закрытого ключа гарантирует невозможность подделки злоумышленником документа и цифровой подписи от имени заверяющего. Открытый ключ — ключ, который может быть опубликован и используется для проверки подлинности подписанного документа, а также для предупреждения мошенничества со стороны заверяющего лица в виде отказа его от подписи документа. Открытый ключ подписи вычисляется, как значение некоторой функции от закрытого ключа, но знание открытого ключа не дает возможности определить закрытый ключ.
- **Вычислительная сложность алгоритма** - это функция, определяющая зависимость объёма работы, выполняемой некоторым алгоритмом, от размера входных данных.
- **Задача об укладке рюкзака** – это одна из *NP*-полных задач комбинаторной оптимизации. Название своё получила от максимизационной задачи укладки как можно большего числа нужных

вещей в рюкзак при условии, что общий объём (или вес) всех предметов, способных поместиться в рюкзак, ограничен.

- **Канал связи** – это система технических средств и среда распространения сигналов для передачи сообщений (не только данных) от источника к получателю (и наоборот). Канал связи, понимаемый в узком смысле (тракт связи), представляет только физическую среду распространения сигналов, например, физическую линию связи.
- **Канал передачи данных** – это система, которая определяется наличием минимум двух каналов связи, обеспечивающих передачу сигнала во взаимопротивоположных направлениях.
- **Ключ** – это секретная информация, используемая алгоритмом при шифровании / дешифровании сообщений, постановке и проверке цифровой подписи. При использовании одного и того же алгоритма результат шифрования зависит от ключа.
- **Криптографический протокол** - это абстрактный или конкретный протокол, включающий набор алгоритмов. В основе протокола лежит набор правил, регламентирующих использование преобразований и алгоритмов в информационных процессах.
- **Модель Долев – Яо** – это модель, широко используемая для описания среды, в которой происходит обмен шифрованными сообщениями.
- **Теория Шеннона –** это раздел прикладной математики, радиотехники (теория обработки сигналов) и информатики, относящийся к измерению количества информации, её свойства и устанавливающий предельные соотношения для систем передачи данных. Как и любая математическая теория, теория, оперирует математическими моделями, а не реальными физическими объектами (источниками и каналами связи). Использует, главным образом, математический аппарат теории вероятностей и математической статистики. Основные разделы теории информации — кодирование источника (сжимающее кодирование) и канальное (помехоустойчивое) кодирование. Теория информации тесно связана с

информационной энтропией, коммуникационными системами, криптографией и другими смежными дисциплинами.

- **Общая память –** это общая база документов (сообщений), которая имеется у отправителя и получателя. Используется в алгоритмах шифрования и дешифрования передаваемой информации.
- **Симметричная криптосистема** (симметричное шифрование, симметричные шифры) – это способ шифрования, в котором для шифрования и дешифрования применяется один и тот же криптографический ключ.
- **Симметричные ключи** – это ключи, используемые в симметричных алгоритмах (шифрование, выработка кодов аутентичности). Главное свойство симметричных ключей: для выполнения как прямого, так и обратного преобразования (шифрование / дешифрование) необходимо использовать один и тот же ключ (либо же ключ для обратного преобразования легко вычисляется из ключа для прямого преобразования, и наоборот). С одной стороны, это обеспечивает более высокую конфиденциальность сообщений, с другой стороны, создаёт проблемы распространения ключей в системах с большим количеством пользователей.
- **Жадный алгоритм** – это алгоритм, заключающийся в принятии локально оптимальных решений на каждом этапе, допуская, что конечное решение также окажется оптимальным.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1 Александров А.В., Метлинов А.Д. «Алгоритмические и статистические свойства разреженной рюкзачной криптосистемы с общей памятью» // Изв. вузов. Приборостроение. 2017. Т. 60, № 1;

2 Ананий В. Глава 3. Метод грубой силы: Задача о рюкзаке // Алгоритмы: введение в разработку и анализ. — М.: «Вильямс», 2006. — С. 160-163, ISBN 5-8459-0987-2;

3 Александров А.В., Метлинов А.Д. К вопросу об особенностях реализации симметричной рюкзачной криптосистемы с общей памятью и плотностью укладки больше единицы // XXXIII Всероссийская НТК «Проблемы эффективности и безопасности функционирования сложных технических и информационных систем», г. Серпухов, сборник научных трудов, часть 4, с. 161-167, 2014;

4 Александров А.В., Метлинов А.Д. О реализации и свойствах хеш-функций, основанных на XOR-свертке блоков симметричной рюкзачной криптосистемы с общей памятью // V Всероссийская научно-техническая конференция ИКВО НИУ ИТМО «Проблема комплексного обеспечения информационной безопасности и совершенствование образовательных технологий подготовки специалистов силовых структур», г. Санкт-Петербург, с. 13-16, 2014;

5 Александров А.В., Метлинов А.Д. О симметричной рюкзачной криптографической системе, устойчивой к L^3-атакам // Сборник научных трудов X международной научно-технической конференции «Перспективные технологии в средствах передачи информации - ПТСПИ-2013», 2013, т.1, с. 167-169;

6 Александров А.В., Метлинов А.Д. О симметричных рюкзачных криптографических системах с разреженной плотностью укладки // Пятая международная научная конференция «Современные методы и проблемы теории операторов и гармонического анализа и их приложения V» в г. Ростов-на-Дону, издательский центр ДГТУ, с. 150-151, 2015;

7 Александров А.В., Метлинов А.Д. Симметричная рюкзачная криптосистема с общей памятью и высокой плотностью укладки // Изв. вузов. Приборостроение, Т. 58, № 5. с. 344—350, 2015;

8 Александров А.В., Метлинов А.Д. Симметричная рюкзачная криптосистема с общей памятью и плотностью укладки больше единицы // Журнал «Проблемы информационной безопасности. Компьютерные системы», №4 2014, с. 58-65;

9 Александров А.В., Метлинов А.Д. Симметричная рюкзачная криптографическая система, устойчивая к L^3-атакам // Сборник научных трудов I Международной научно-практической конференции «Информационная безопасность в свете Стратегии Казахстан - 2050», 2013, с 425-430;

10 Александров А.В., Метлинов А.Д., Зимников А.С. О семействе рюкзачных блочных шифров с общей памятью и плотностью укладки больше единицы и хеш-функций на их основе // Сборник научных трудов II Международной научно-практической конференции «Информационная безопасность в свете Стратегии Казахстан - 2050», 2014, с. 31-35;

11 Гольдштейн Б.С. Системы коммутации: Учебник для ВУЗов. 2-е изд. - СПб.: БХВ - Санкт-Петербург, 2004. - 314 с, ISBN 5-8206-0108-4;

12 Гончарок М. Х., Крюков Ю. С. Построение системы защиты информации в цифровых АТС и выбор класса защищенности // Защита информации. Конфидент - 2004 № 2. - с. 2-7;

13 Григорик И. High Performance Browser Networking. O'Reilly Media, 2013, ISBN 978-1-4493-4476-4;

14 Иванов М.А. Криптографические методы защиты информации в компьютерных системах и сетях. М.: КУДИЦ-ОБРАЗ, 2001, 368с, ISBN 5-93378-021-9;

15 Корнышев Ю.Н., Романцов В.М., Стовбун Г.В. Сигнализация на телефонных сетях: Учебн. пособие / Украинская Государственная Академия связи им. А.С.Попова. Одесса, 1996. - 64 с;

16 Метлинов А.Д. О скоростных особенностях и NIST-тестировании симметричной рюкзачной криптографической системы с общей памятью // Сборник трудов XI Международной научной конференции «Перспективные технологии в средствах передачи информации», 2015;

17 Метлинов А.Д. Передача сообщений на основе схемы SMT LSS // Инновации в науке, Аэтерна, Уфа, сборник научных трудов, том 1, с. 3-6, 2014;

18 Метлинов А.Д. Скоростные характеристики симметричной рюкзачной криптосистемы с общей памятью и плотностью укладки больше единицы. NIST - тестирование // Сборник научных трудов III Международной научно-практической конференции «Информационная безопасность в свете Стратегии Казахстан - 2050», с. 247-252, 2015;

19 Мурин Д. М. Модификация метода Лагариаса–Одлыжко для решения обобщённой задачи о рюкзаке и систем задач о рюкзаках, ПДМ, 2013, № 2, 91–100;

20 Панасенко С. Алгоритмы шифрования: СПб, БХВ, 2008. 563 с, ISBN 978-5-9775-0319-8;

21 Слоэн Н. Дж. А. «Коды, исправляющие ошибки, и криптография» - в сб. Математический цветник. – М.: Мир, 1983, с 432-472, ISBN 5-477-00180-1;

22 Черемушкин А.В. Криптографические протоколы: основные свойства и уязвимости. М.: 2009, 36с, ISBN 978-5-7695-5748-4;

23 Шеннон К. Работы по теории информации и кибернетике. // ИИЛ, Москва 1963, 829с, ISBN 5-7417-0197-3;

24 Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си // М.: "Триумф", 2002, ISBN 5-89392-055-4;

25 Ященко. Введение в криптографию. Новые математические дисциплины. // МЦНМО Санкт-Петербург, 2001, 288с, ISBN 5-318-00443-1;

26 Agarwal S., Cramer R., and de Haan R. Asymptotically Optimal Two-Round Perfectly Secure Message Transmission CRYPTO 2006: 394-408, ISBN 978-3-540-70935-0;

27 ANSI X3.106, «American National Standard for Information Systems-Data Link Encryption», American National Standards Institute, 1983, ISBN 92-871-0022-5;

28 Berlekamp E. R., McEliece R. J. and van Tilborg H. C. A., On the inherent intractability of certain coding problems, IEEE Trans. Inf. Theory, vol. IT-24, no. 3, pp. 384-386, May 1978, ISBN 3-540-43328-7;

29 Berners-Lee, T., Fielding, R., and H. Fristik, «Hypertext Transfer Protocol — HTTP/1.0», RFC-1945, Май 1996, ISBN 3-540-64700-7;

30 Bleichenbacher D., «Chosen Ciphertext Attacks against Protocols Based on RSA Encryption Standard PKCS #1» in Advances in Cryptology — CRYPTO'98, LNCS vol. 1462, p. 1-12, 1998, ISBN 978-3-540-64892-5;

31 CITT. Recommendation X.509: «The Directory — Authentication Framework». 1988;

32 Coster M. J., Joux A., LaMacchia B. A., et al. Improved low-density subset sum algorithms // Computational Complexity. 1992. No. 2. P. 111–128, ISBN 3-528-06619-9;

33 Desmedt Y., Wang Y. and Burmester M. A Complete Characterization of Tolerable Adversary Structures for Secure Point-to-Point Transmissions Without Feedback. ISAAC 2005: 277-287, ISBN 978-3-540-30935-2;

34 Diffie W., Hellman M. Hiding in-formation and signatures in trapdoor knapsacks // Information Theory, IEEE Transactions, 1978, P. 525-530;

35 Diffie W. and Hellman M. E., «New Directions in Cryptography», IEEE Transactions on Information Theory, V. IT-22, n. 6, Jun 1977, p. 74-84;

36 Dolev D., Dwork C., Waarts O., Yung M. Perfectly Secure Message Transmission. J. ACM 40(1): pp.17- 47 (1993), ISBN 978-3-540-43553-2;

37 Dolev D., Dwork C., Naor M. Nonmalleable cryptography, SIAM Journal on Computing, 30 (2):, 2003 727–784;

38 Dolev D. The Byzantine generals strike again, Journal of Algorithms, 3 (1):, 1982 14–30;

39 Dolev D., Yao A. On the Security of Public Key Protocols // IEEE Transact. on Inform. Theory. 1983. Vol. 29, N 2. P. 198—208, ISBN 5-8459-0847-7;

40 Franklin M., Wright R. Secure communication in minimal connectivity models // J. Cryptology. 2000. Vol. 13, N 1. P. 9—30, ISBN 978-3-540-92294-0;

41 Frier A., Karlton P., and Kocher P., «The SSL 3.0 Protocol», Netscape Communications Corp., Nov 18, 1996, ISBN 3-89376-105-5;

42 Hickman, Kipp, «The SSL Protocol», Netscape Communications Corp., Feb 9, 1995, ISBN 0-201-10150-5;

43 Hirt M. and Maurer U. Player Simulation and General Adversary Structures in Perfect Multiparty Computation. J. Cryptology 13(1): 31-60 (2000), ISBN 978-3-540-67517-4;

44 Housley R., Ford W., Polk W. и D. Solo, «Internet Public Key Infrastructure: Part I: X.509 Certificate and CRL Profile», RFC-2459, Январь 1999;

45 Kaliski B., «The MD2 Message Digest Algorithm», RFC-1319, Апрель 1992;

46 Karp R. M. Reducibility among combinatorial problems // Complexity of Computer Computations: Proc. of a Symp. on the Complexity of Computer Computations, the IBM Research Symposia Series. NY: Plenum Press, 1972. P. 85–103, ISBN 978-3-540-68274-5;

47 Krawczyk, H., Bellare, M., and R. Canetti, «HMAC: Keyed-Hashing for Message Authentication», RFC-2104, Февраль 1997, ISBN 0-13-014249-2;

48 Kurosawa K. General Error Decodable Secret Sharing Scheme and Its Application. Cryptology ePrint Archive Report/ 2009. P. 263, ISBN 978-3-642-17618-0;

49 Kurosawa Kaoru, Suzuki Kazuhiro, Almost Secure (1-Round, n-Channel) Message Transmission Scheme, Information Theoretic Security, Lecture Notes in Computer Science, Volume 4883. Springer-Verlag Berlin Heidel-berg, 2009, p. 99, ISBN 978-3-642-10229-5;

50 Lagarias J. C., Odlyzko A. M. Solving low-density subset problems, Proc. 24th Annual IEEE Symp. on Found, of Corp. Science, pp. 1-10, 1983, ISBN 978-3-540-54620-7;

51 Lai X., «On the Design and Security of Block Ciphers», ETH Series in Information Processing, v. 1, Konstanz: Hartung-Gorre Verlag, 1992, ISBN 0-387-97930-1;

52 MacWilliams F.J. and Sloanem N.J.A. The Theory of Error-correcting Codes, North-Holland, 1981, ISBN 0-444-85193-3;

53 Merkle R.C., Hellman M.E. Hiding information and signatures in trapdoor knapsacks, IEEE Transactions on Information Theory, IT-24, pp. 525-530, 1978, ISBN 978-3-540-16076-2;

54 NIST FIPS PUB 180-1, «Secure Hash Standard», National Institute of Standards and Technology, U.S. Department of Commerce, Work in Progress, May 31, 1994;

55 NIST FIPS PUB 186, «Digital Signature Standard», National Institute of Standards and Technology, U.S. Department of Commerce, May 18, 1994;

56 Odlyz Hamlin, N., Krishnamoorthy, B., and Webb, W. A Knapsack-Like Code Using Recurrence Sequence Representations. Fibonacci Quarterly, 1(53), P. 24-33;

57 Odlyzko A. M. and Lagarias J. C. Solving Low-Density Subset Sum Problems // J. Association Computing Machinery. 1985. V. 32. No.1. P. 229–246;

58 Ogata W., Kurosawa K., Stinson D. Optimum Secret Sharing Scheme Secure against Cheating. SIAM J. Discrete Math. 20(1): 79-95 (2006), ISBN 978-3-642-20464-7;

59 Postel J., «Протокол управления передачей (TCP)», RFC 793, Сентябрь 1981, ISBN 0-13-212571-4;

60 Postel J., and J. Reynolds, «Telnet Option Specifications», STD-8, RFC-855, Май 1993, ISBN 0-932376-88-6;

61 Postel J., and J. Reynolds, «Telnet Protocol Specifications», STD-8, RFC-854, Май 1993, ISBN 1-55558-164-1;

62 Postel J., and J. Reynolds, «File Transfer Protocol», STD-9, RFC-959, October 1985, ISBN 0-932376-88-6;

63 Rivest, R., «A Description of the RC2(r) Encryption Algorithm», RFC 2268, Январь 1998, ISBN 0-14-024432-8;

64 R. Rivest, A. Shamir, and L. M. Adleman, «A Method for Obtaining Digital Signatures and Public-Key Cryptosystems», Communications of the ACM, v. 21, n. 2, Feb 1978, pp. 120-126;

65 RSA Laboratories, «PKCS #1: RSA Encryption Standard», version 1.5, Ноябрь 1993;

66 RSA Laboratories, «PKCS #6: RSA Extended Certificate Syntax Standard», version 1.5, Ноябрь 1993;

67 RSA Laboratories, «PKCS #7: RSA Cryptographic Message Syntax Standard», version 1.5, Ноябрь 1993;

68 Shannon C. Communication theory of secrecy systems // Bell System Techn. J. 1949. Vol. 28, № 4. P. 656—715, ISBN 0-7803-0434-9:

69 Shamir A. How to share a secret // Communication of ACM. 1979. Vol. 22, N 11. P. 612—613, ISBN 5-89392-055-4;

70 Srinivansan R., Sun Microsystems, RFC-1832: XDR: External Data Representation Standard, Август 1995, ISBN 0-13-825001-4;

71 Srinathan K., Narayanan A., Pandu Rangan C. Optimal Perfectly Secure Message Transmission. CRYPTO 2004: 545-561, ISBN 978-3-540-77026-8;

72 Tompa M. and Woll H. How to share a secret with cheaters, Journal of Cryptology 1 (1988), 133-138, ISBN 978-3-540-43861-8;

73 Tuchman W., «Hellman Presents No Shortcut Solutions To DES», IEEE Spectrum, v. 16, n. 7, Июль 1979, p. 40-41, ISBN 0-471-59756-2;

74 Yang Q., Desmedt Y. Cryptanalysis of Secure Message Transmission Protocols with Feedback // ICITS. 2009. P. 159—176, ISBN 978-3-642-14495-0.

Приложение А. Сводные результаты статистического и сравнительного скоростного экспериментов

В приложении представлены сводные результаты эксперимента по исследованию наличия статистической зависимости вероятности успешной реализации L^3-атаки от плотности укладки рюкзака ρ при различных объемах исходного текста (таблицы А.1 – А.2), а также результаты сравнительного скоростного тестирования разработанной симметричной рюкзачной криптосистемы с известными алгоритмами (рисунки А.1 – А.10).

Таблица А.1 - Вычисление коэффициента ковариации

№	Величина ρ	Величина P	Центрированная величина ρ	Центрированная величина P	Произведение центрированных величин
	ρ_i	P_i	$(\rho_i - M_\rho)$	$(P_i - M_P)$	$(\rho_i - M_\rho) \bullet (P_i - M_P)$
1	0.9907102	0,0120000	-0.0045200	-0.3380000	0.0015300
2	0.9913402	0,0130000	-0.0038900	-0.3380000	0.0013200
3	0.9914219	0,0130000	-0.0038100	-0.2380000	0.0009100
4	0.9923125	0,0140000	-0.0029200	-0.1380000	0.0004000
5	0.9925017	0,0145000	-0.0027300	-0.1380000	0.0003800
6	0.9928858	0,0170000	-0.0023500	-0.1380000	0.0003200
7	0.9933907	0,0135000	-0.0018400	-0.0380000	0.0000700
8	0.9936028	0,0140000	-0.0016300	-0.0380000	0.0000600
9	0.9939589	0,0130000	-0.0012700	-0.0380000	0.0000500
10	0.9940932	0,0185000	-0.0011400	-0.0380000	0.0000400
11	0.9942135	0,0130000	-0.0010200	-0.0380000	0.0000400
12	0.9946581	0,0130000	-0.0005800	-0.0380000	0.0000200
13	0.9947205	0,0135000	-0.0005100	-0.0380000	0.0000200
14	0.9947814	0,0140000	-0.0004500	-0.0380000	0.0000200
15	0.9948714	0,0120000	-0.0003600	-0.0380000	0.0000100
16	0.9949812	0,0145000	-0.0002500	0.0120000	0.0000000
17	0.9949875	0,0130000	-0.0002500	0.0120000	0.0000000

Продолжение таблицы А.1

№	Величина ρ	Величина P	Центрированная величина ρ	Центрированная величина P	Произведение центрированных величин
	ρ_i	P_i	$(\rho_i - M_\rho)$	$(P_i - M_P)$	$(\rho_i - M_\rho) \bullet (P_i - M_P)$
18	0.9981128	0,0130000	0.0028800	0.0620000	0.0001800
19	0.9981761	0,0170000	0.0029400	0.0620000	0.0001800
20	0.9985134	0,0130000	0.0032800	0.0620000	0.0002000
21	0.9989735	0,0120000	0.0037400	0.1120000	0.0004200
22	0.9989853	0,0130000	0.0037500	0.1120000	0.0004200
23	0.9989918	0,0110000	0.0037600	0.3620000	0.0013600
24	0.9997614	0,0100000	0.0045300	0.3620000	0.0016400
25	0.9998912	0,0100000	0.0046600	0.5120000	0.0023800

Таблица А.2 - Вычисление коэффициента корреляции

№	Величина ρ	Величина P	Центрированная величина ρ и ее квадрат		Центрированная величина P и ее квадрат	
	ρ_i	P_i	$(\rho_i - M_\rho)$	$(\rho_i - M_\rho)^2$	$(P_i - M_P)$	$(P_i - M_P)^2$
1	0.9907102	0,0120000	-0.0045200	0.0000200	-0.3380000	0.1142400
2	0.9913402	0,0130000	-0.0038900	0.0000200	-0.3380000	0.1142400
3	0.9914219	0,0130000	-0.0038100	0.0000100	-0.2380000	0.0566400
4	0.9923125	0,0140000	-0.0029200	0.0000100	-0.1380000	0.0190400
5	0.9925017	0,0145000	-0.0027300	0.0000100	-0.1380000	0.0190400
6	0.9928858	0,0170000	-0.0023500	0.0000100	-0.1380000	0.0190400
7	0.9933907	0,0135000	-0.0018400	0.0000000	-0.0380000	0.0014400
8	0.9936028	0,0140000	-0.0016300	0.0000000	-0.0380000	0.0014400
9	0.9939589	0,0130000	-0.0012700	0.0000000	-0.0380000	0.0014400
10	0.9940932	0,0185000	-0.0011400	0.0000000	-0.0380000	0.0014400
11	0.9942135	0,0130000	-0.0010200	0.0000000	-0.0380000	0.0014400
12	0.9946581	0,0130000	-0.0005800	0.0000000	-0.0380000	0.0014400
13	0.9947205	0,0135000	-0.0005100	0.0000000	-0.0380000	0.0014400
14	0.9947814	0,0140000	-0.0004500	0.0000000	-0.0380000	0.0014400

Продолжение таблицы А.2

№	Величина ρ	Величина P	Центрированная величина ρ и ее квадрат		Центрированная величина P и ее квадрат	
	ρ_i	P_i	$(\rho_i - M_\rho)$	$(\rho_i - M_\rho)^2$	$(P_i - M_P)$	$(P_i - M_P)^2$
15	0.9948714	0,0120000	-0.0003600	0.0000000	-0.0380000	0.0014400
16	0.9949812	0,0145000	-0.0002500	0.0000000	0.0120000	0.0001400
17	0.9949875	0,0130000	-0.0002500	0.0000000	0.0120000	0.0001400
18	0.9981128	0,0130000	0.0028800	0.0000100	0.0620000	0.0038400
19	0.9981761	0,0170000	0.0029400	0.0000100	0.0620000	0.0038400
20	0.9985134	0,0130000	0.0032800	0.0000100	0.0620000	0.0038400
21	0.9989735	0,0120000	0.0037400	0.0000100	0.1120000	0.0125400
22	0.9989853	0,0130000	0.0037500	0.0000100	0.1120000	0.0125400
23	0.9989918	0,0110000	0.0037600	0.0000100	0.3620000	0.1310400
24	0.9997614	0,0100000	0.0045300	0.0000200	0.3620000	0.1310400
25	0.9998912	0,0100000	0.0046600	0.0000200	0.5120000	0.2621400

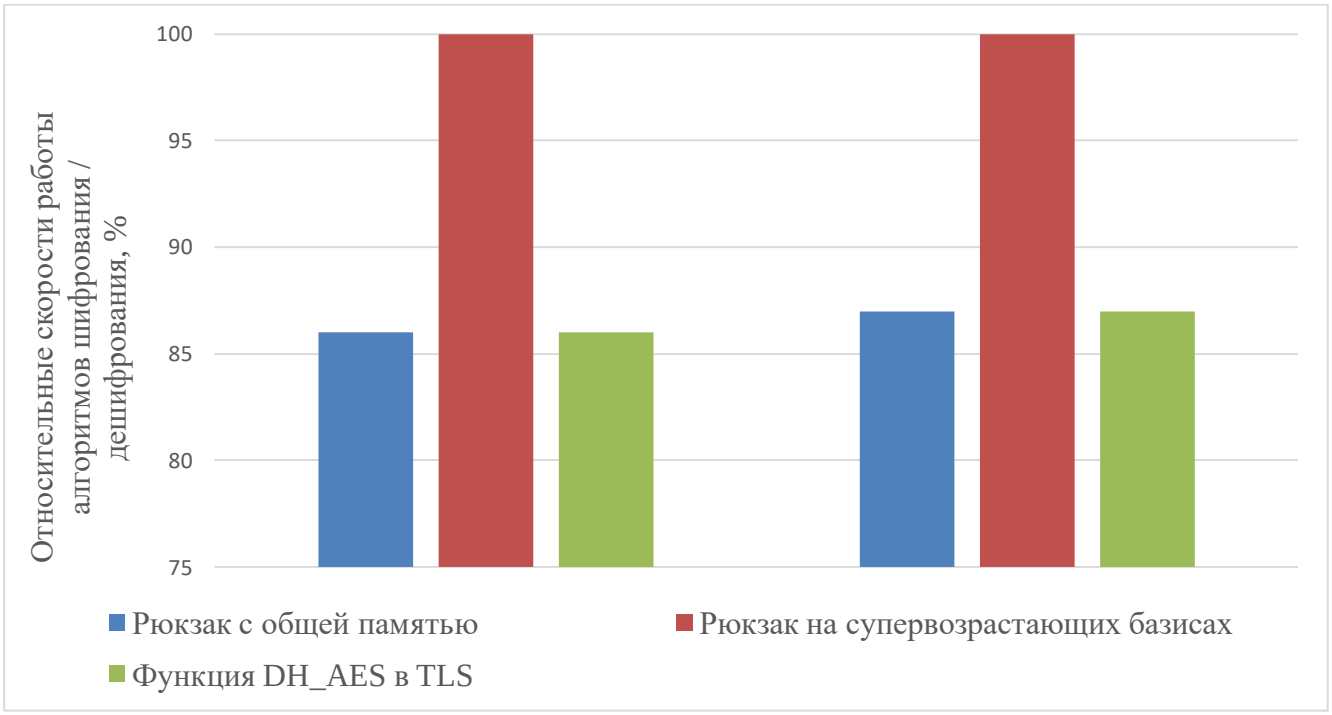

Рисунок А.1 – Сравнение скоростей работы алгоритмов для файла №1

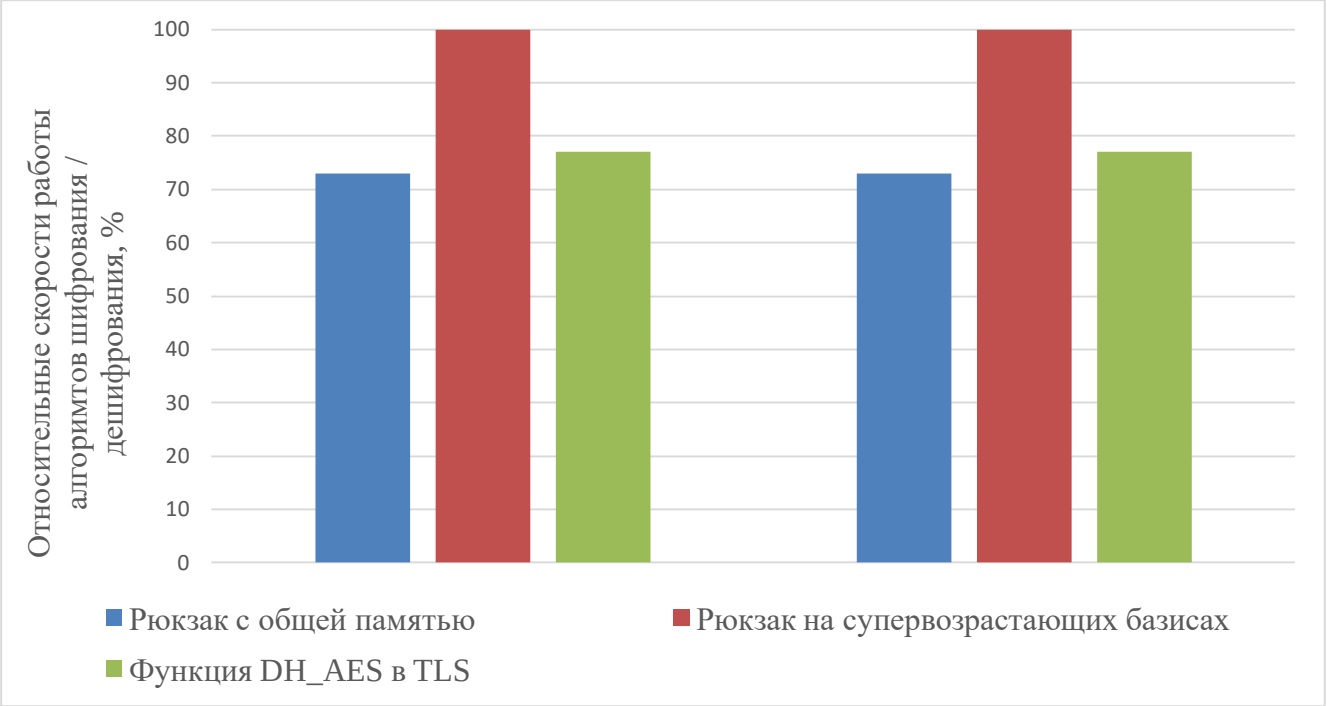

Рисунок А.2 – Сравнение скоростей работы алгоритмов для файла №2

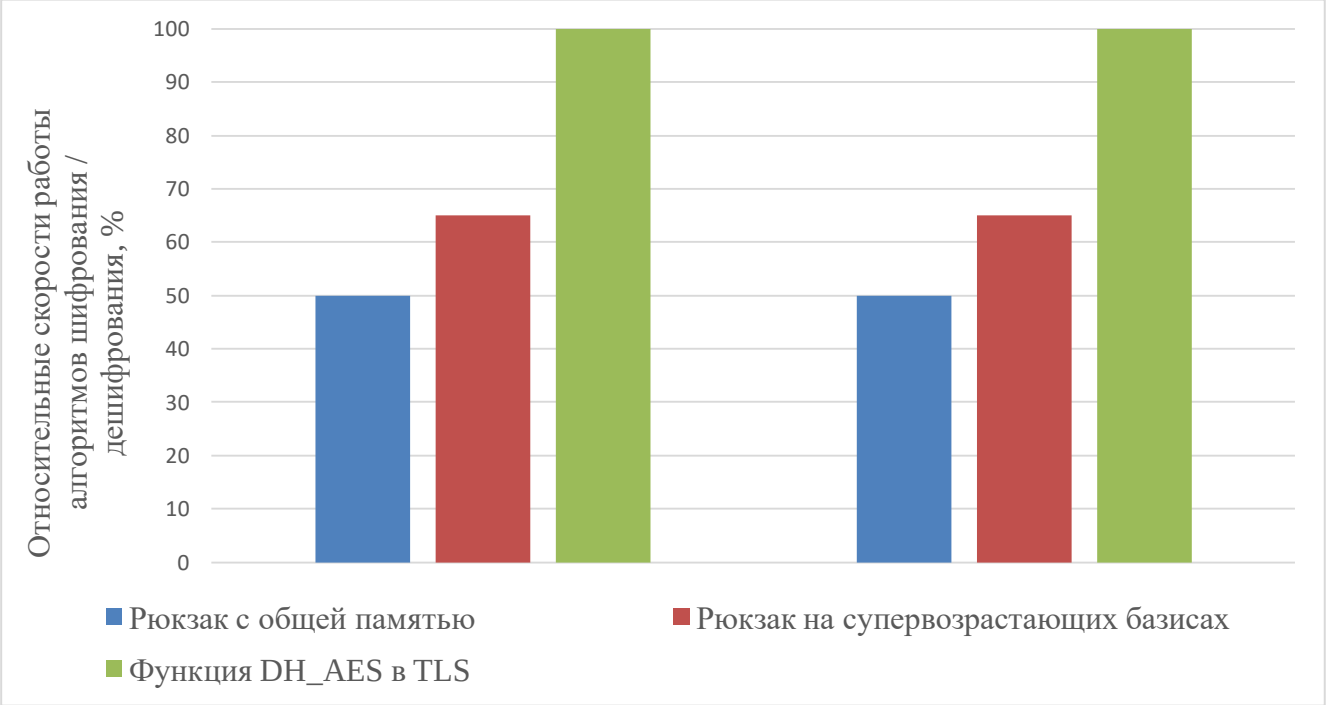

Рисунок А.3 – Сравнение скоростей работы алгоритмов для файла №3

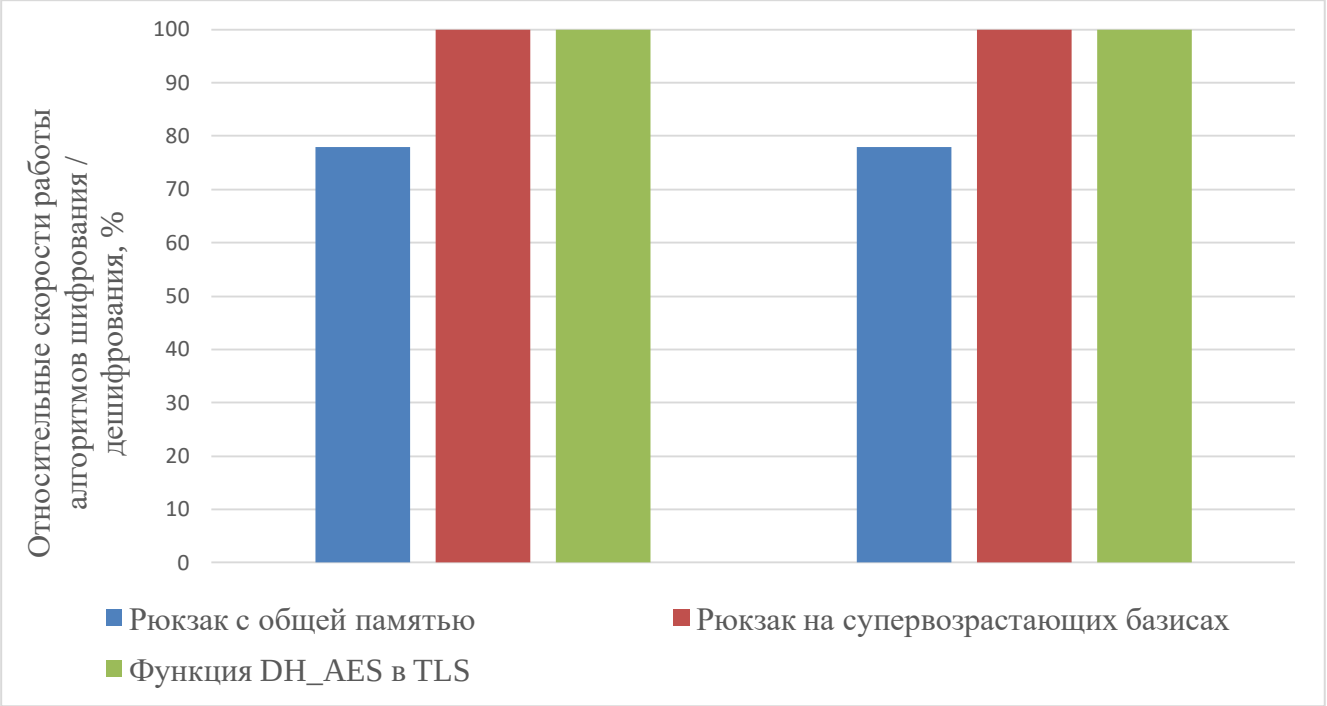

Рисунок А.4 – Сравнение скоростей работы алгоритмов для файла №4

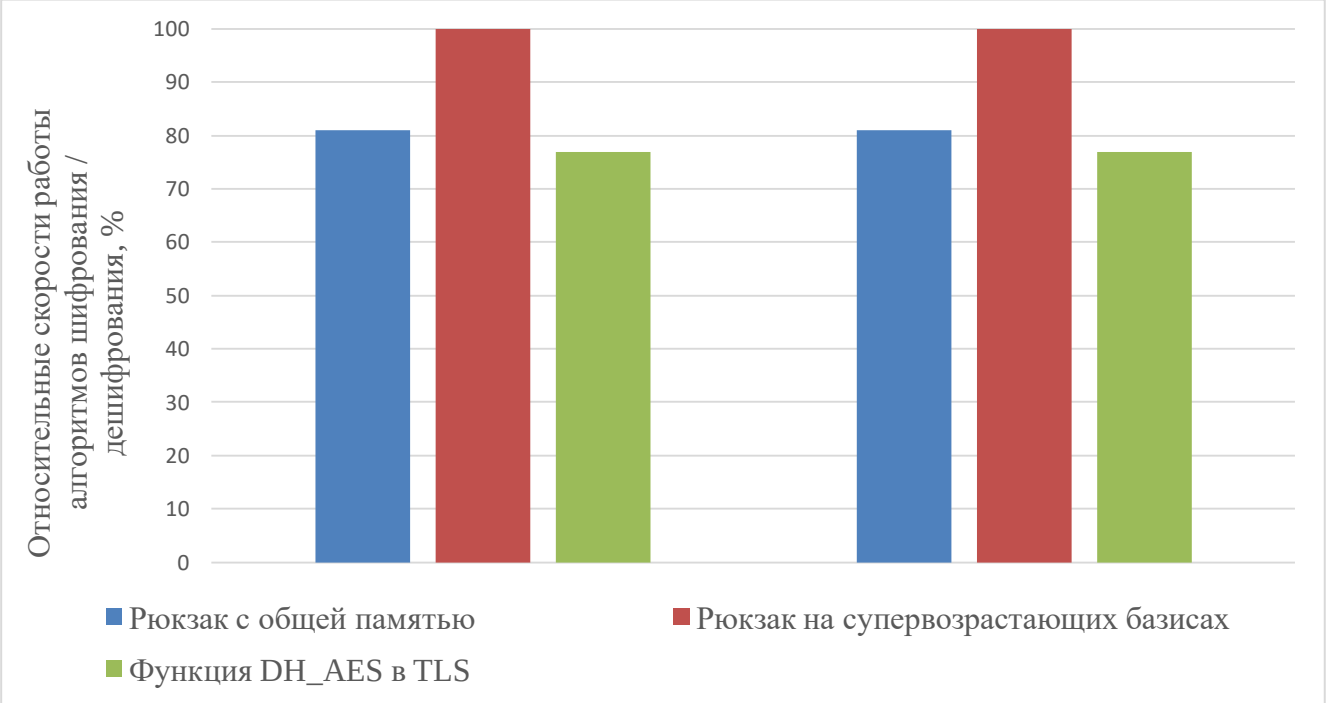

Рисунок А.5 – Сравнение скоростей работы алгоритмов для файла №5

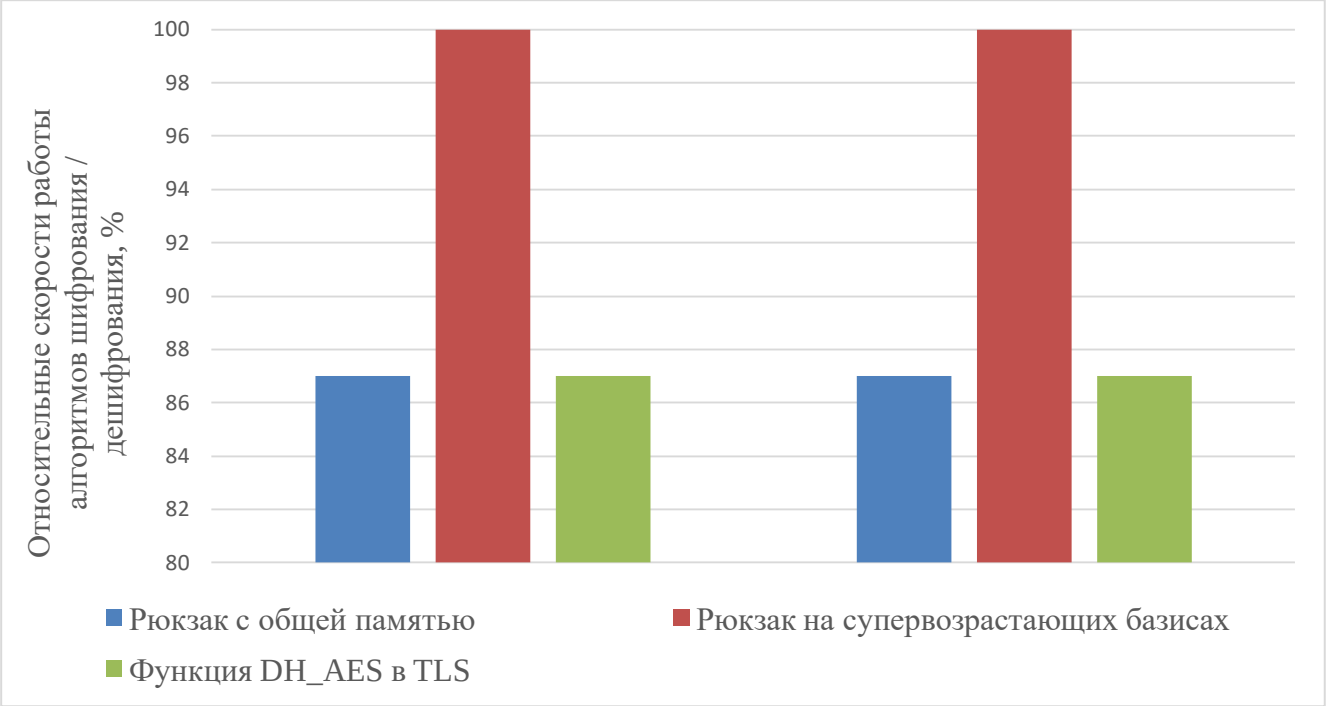

Рисунок А.6 – Сравнение скоростей работы алгоритмов для файла №6

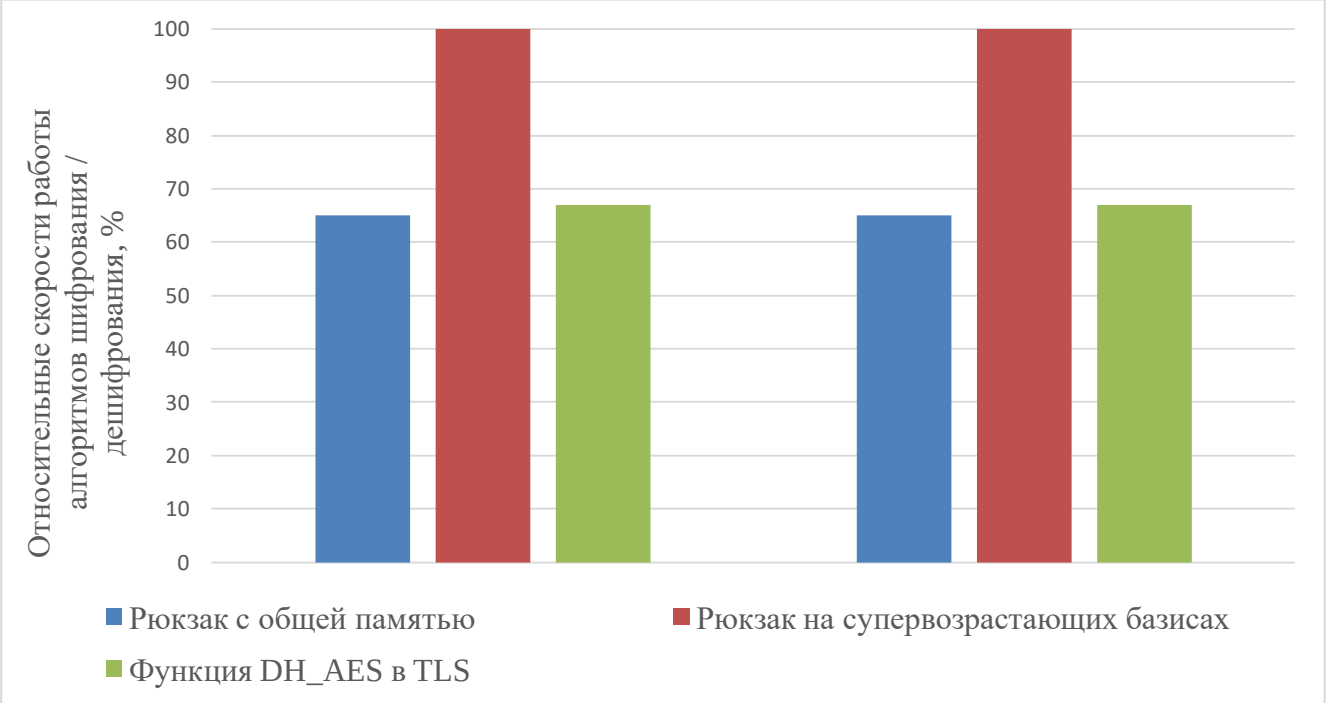

Рисунок А.7 – Сравнение скоростей работы алгоритмов для файла №7

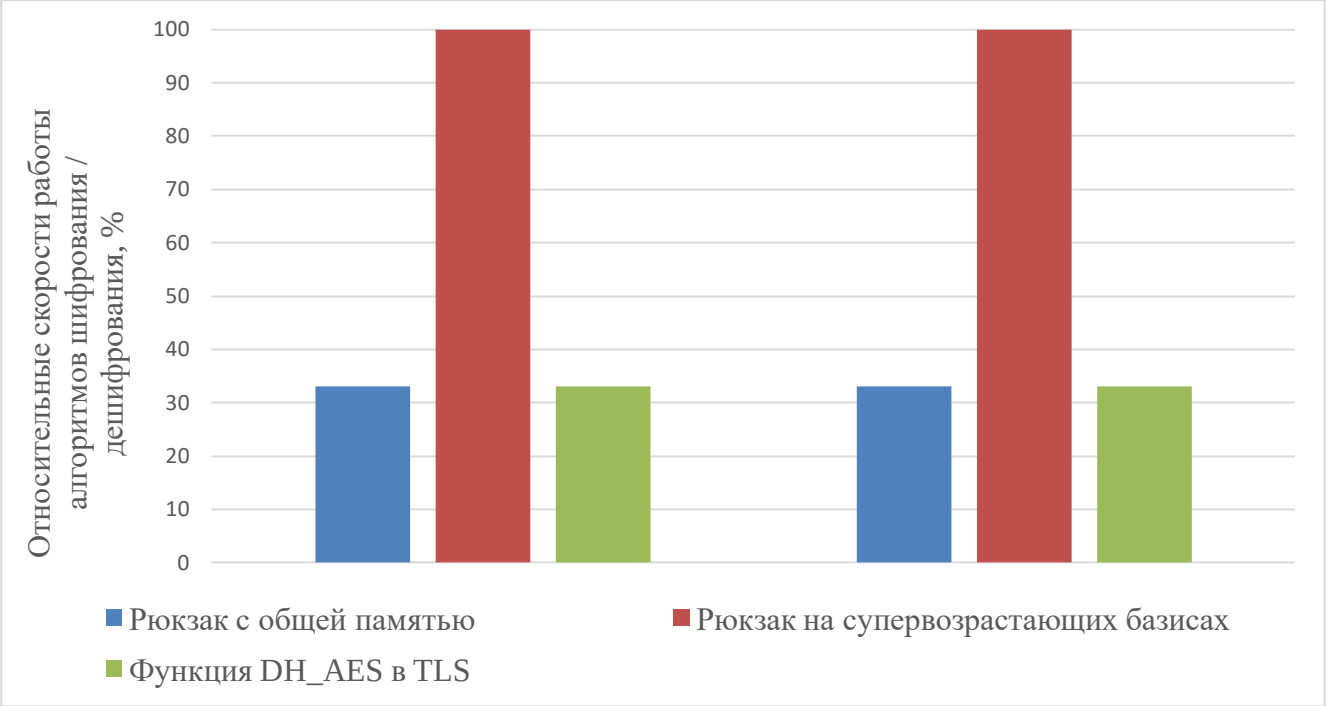

Рисунок А.8 – Сравнение скоростей работы алгоритмов для файла №8

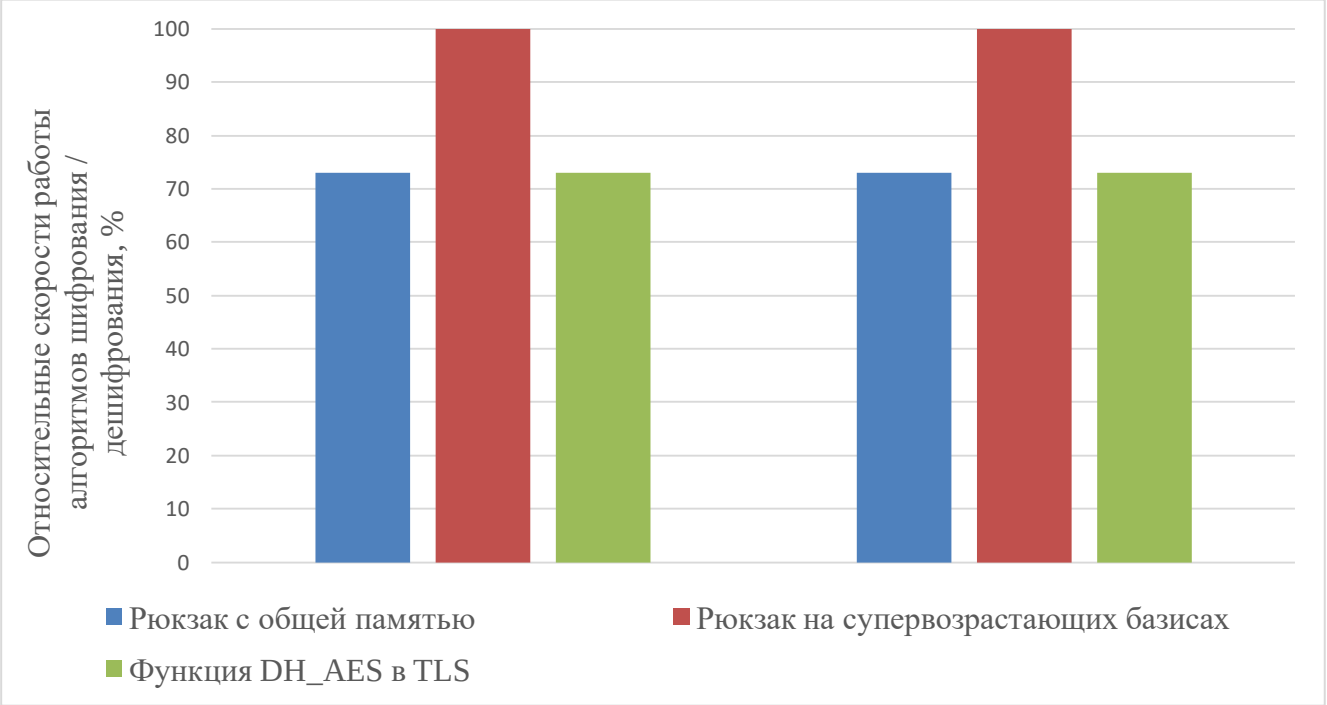

Рисунок А.9 – Сравнение скоростей работы алгоритмов для файла №9

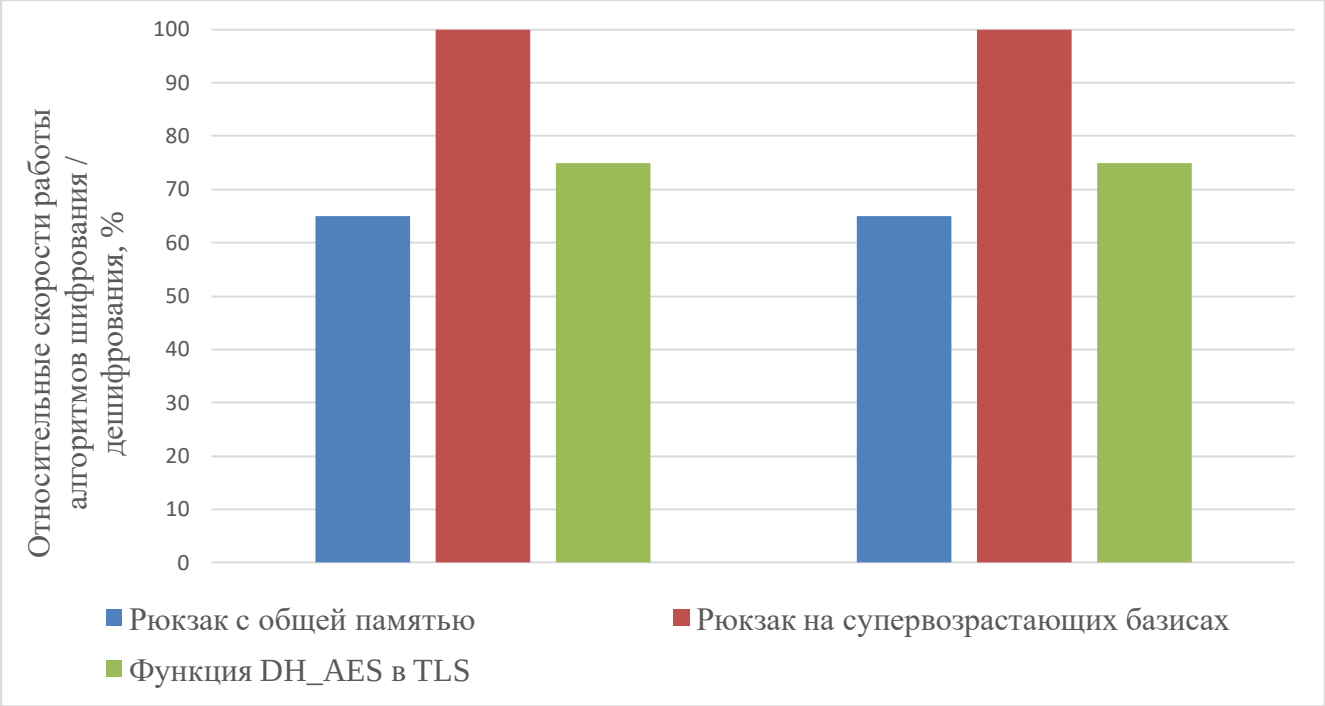

Рисунок А.10 – Сравнение скоростей работы алгоритмов для файла №10

Приложение Б. Экспериментальная зависимость вероятности успешной реализации L^3-атаки от плотности укладки рюкзака

В приложении представлены результаты эксперимента по исследованию зависимости вероятности успешной реализации L^3-атаки (таблица Б.1) от плотности укладки рюкзака ρ при различных объемах исходного текста (рисунки Б.1 – Б.10).

Таблица Б.1 - Выявление вероятности L^3-атаки от плотности укладки рюкзака

Номер	E = {e1, e2, … , eB}	Сообщение S	L3-атака "да"	L3-атака "нет"	Время атаки, мин	Значение ρ
1	11111111111111111111111111111111111111111111101110111111011111011110	140593	2 из 100	98 из 100	1	0.9989735
		517998	0 из 100	100 из 100	3	
		997503	2 из 100	98 из 100	17	
		330504	1 из 100	99 из 100	2	
		198510	2 из 100	98 из 100	3	
		4112284	2 из 100	98 из 100	8	
		94028916	0 из 100	100 из 100	140	
		210861	1 из 100	99 из 100	1	
		40874	2 из 100	98 из 100	3	
		206405	1 из 100	99 из 100	4	
		11925263513	1 из 100	99 из 100	210	
		18607676768	0 из 100	100 из 100	340	
		1031871	1 из 100	99 из 100	9	
		783490	1 из 100	99 из 100	11	
		935366	1 из 100	99 из 100	15	
		179248	2 из 100	98 из 100	7	
		5112284	1 из 100	99 из 100	10	
		612791	1 из 100	99 из 100	5	
		176251	1 из 100	99 из 100	1	
		54308	2 из 100	98 из 100	2	
2	11011111111011011011011011111111111101101111101110111010111011110	140593	2 из 100	98 из 100	1	0.9946581
		517998	0 из 100	100 из 100	3	
		997503	2 из 100	98 из 100	17	
		330504	1 из 100	99 из 100	2	
		198510	2 из 100	98 из 100	3	
		4112284	2 из 100	98 из 100	8	
		94028916	0 из 100	100 из 100	140	
		210861	1 из 100	99 из 100	1	
		40874	1 из 100	99 из 100	3	
		206405	1 из 100	99 из 100	4	
		11925263513	2 из 100	98 из 100	210	
		18607676768	0 из 100	100 из 100	340	
		1031871	1 из 100	99 из 100	9	
		783490	2 из 100	98 из 100	11	
		935366	2 из 100	98 из 100	15	
		179248	2 из 100	98 из 100	7	
		5112284	1 из 100	99 из 100	10	
		612791	1 из 100	99 из 100	5	
		176251	1 из 100	99 из 100	1	
		54308	2 из 100	98 из 100	2	

Продолжение таблицы Б.1

Номер	E = {e1, e2, … , eB}	Сообщение S	L3-атака "да"	L3-атака "нет"	Время атаки, мин	Значение ρ
3	1101111111101101110111011111111101101101101101010111010110101 1010	140593	2 из 100	98 из 100	1	0.9940932
		517998	0 из 100	100 из 100	3	
		997503	2 из 100	98 из 100	17	
		330504	1 из 100	99 из 100	2	
		198510	2 из 100	98 из 100	3	
		4112284	2 из 100	98 из 100	8	
		94028916	0 из 100	100 из 100	140	
		210861	1 из 100	99 из 100	1	
		40874	1 из 100	99 из 100	3	
		206405	1 из 100	99 из 100	4	
		11925263513	2 из 100	98 из 100	210	
		18607676768	0 из 100	100 из 100	340	
		1031871	1 из 100	99 из 100	9	
		783490	2 из 100	98 из 100	11	
		935366	2 из 100	98 из 100	15	
		179248	2 из 100	98 из 100	7	
		5112284	1 из 100	99 из 100	10	
		612791	1 из 100	99 из 100	5	
		176251	1 из 100	99 из 100	1	
		54308	2 из 100	98 из 100	2	
4	1101111111101101010111011111111011011011001010100010101101010010	140593	2 из 100	98 из 100	1	0.9933907
		517998	1 из 100	99 из 100	3	
		997503	2 из 100	98 из 100	17	
		330504	1 из 100	99 из 100	2	
		198510	2 из 100	98 из 100	3	
		4112284	2 из 100	98 из 100	8	
		94028916	1 из 100	99 из 100	140	
		210861	1 из 100	99 из 100	1	
		40874	1 из 100	99 из 100	3	
		206405	1 из 100	99 из 100	4	
		11925263513	2 из 100	98 из 100	210	
		18607676768	0 из 100	100 из 100	340	
		1031871	1 из 100	99 из 100	9	
		783490	2 из 100	98 из 100	11	
		935366	2 из 100	98 из 100	15	
		179248	2 из 100	98 из 100	7	
		5112284	1 из 100	99 из 100	10	
		612791	1 из 100	99 из 100	5	
		176251	1 из 100	99 из 100	1	
		54308	2 из 100	98 из 100	2	
5	1101111111101101010011010110011011011011001010100010101101010000	140593	2 из 100	98 из 100	1	0.9925017
		517998	1 из 100	99 из 100	3	
		997503	2 из 100	98 из 100	17	
		330504	1 из 100	99 из 100	2	
		198510	2 из 100	98 из 100	3	
		4112284	2 из 100	98 из 100	8	
		94028916	1 из 100	99 из 100	140	
		210861	1 из 100	99 из 100	1	
		40874	1 из 100	99 из 100	3	
		206405	1 из 100	99 из 100	4	
		11925263513	2 из 100	98 из 100	210	
		18607676768	0 из 100	100 из 100	340	
		1031871	1 из 100	99 из 100	9	
		783490	2 из 100	98 из 100	11	
		935366	2 из 100	98 из 100	15	
		179248	2 из 100	98 из 100	7	
		5112284	1 из 100	99 из 100	10	
		612791	2 из 100	98 из 100	5	
		176251	1 из 100	99 из 100	1	
		54308	2 из 100	98 из 100	2	

Продолжение таблицы Б.1

Номер	E = {e1, e2, … , eB}	Сообщение S	L3-атака "да"	L3-атака "нет"	Время атаки, мин	Значение ρ
6	1101111111101101010011010110011011011011001010100010000000000000	140593	2 из 100	98 из 100	1	0.9913402
		517998	2 из 100	98 из 100	3	
		997503	2 из 100	98 из 100	17	
		330504	1 из 100	99 из 100	2	
		198510	2 из 100	98 из 100	3	
		4112284	2 из 100	98 из 100	8	
		94028916	1 из 100	99 из 100	140	
		210861	3 из 100	97 из 100	1	
		40874	1 из 100	99 из 100	3	
		206405	1 из 100	99 из 100	4	
		11925263513	2 из 100	98 из 100	210	
		18607676768	1 из 100	99 из 100	340	
		1031871	1 из 100	99 из 100	9	
		783490	2 из 100	98 из 100	11	
		935366	2 из 100	98 из 100	15	
		179248	2 из 100	98 из 100	7	
		5112284	1 из 100	99 из 100	10	
		612791	2 из 100	98 из 100	5	
		176251	1 из 100	99 из 100	1	
		54308	3 из 100	97 из 100	2	
7	1100001111101101010000010110011011011011001010100010000000000000	140593	2 из 100	98 из 100	1	0.9985134
		517998	0 из 100	100 из 100	3	
		997503	2 из 100	98 из 100	17	
		330504	2 из 100	98 из 100	2	
		198510	2 из 100	98 из 100	3	
		4112284	2 из 100	98 из 100	8	
		94028916	0 из 100	100 из 100	140	
		210861	1 из 100	99 из 100	1	
		40874	2 из 100	98 из 100	3	
		206405	1 из 100	99 из 100	4	
		11925263513	1 из 100	99 из 100	210	
		18607676768	0 из 100	100 из 100	340	
		1031871	1 из 100	99 из 100	9	
		783490	1 из 100	99 из 100	11	
		935366	1 из 100	99 из 100	15	
		179248	2 из 100	98 из 100	7	
		5112284	2 из 100	98 из 100	10	
		612791	2 из 100	98 из 100	5	
		176251	1 из 100	99 из 100	1	
		54308	2 из 100	98 из 100	2	
8	1000001001101101010000010110011000011011001010100010000000000000	140593	2 из 100	98 из 100	1	0.99288584
		517998	1 из 100	99 из 100	3	
		997503	2 из 100	98 из 100	17	
		330504	1 из 100	99 из 100	2	
		198510	2 из 100	98 из 100	3	
		4112284	1 из 100	99 из 100	8	
		94028916	1 из 100	99 из 100	140	
		210861	1 из 100	99 из 100	1	
		40874	1 из 100	99 из 100	3	
		206405	1 из 100	99 из 100	4	
		11925263513	2 из 100	98 из 100	210	
		18607676768	0 из 100	100 из 100	340	
		1031871	1 из 100	99 из 100	9	
		783490	2 из 100	98 из 100	11	
		935366	2 из 100	98 из 100	15	
		179248	2 из 100	98 из 100	7	
		5112284	1 из 100	99 из 100	10	
		612791	2 из 100	98 из 100	5	
		176251	1 из 100	99 из 100	1	
		54308	2 из 100	98 из 100	2	

Продолжение таблицы Б.1

Номер	E = {e1, e2, … , eB}	Сообщение S	L3-атака "да"	L3-атака "нет"	Время атаки, мин	Значение ρ
9	10000010010000000000000101100000000000000000000000000000000000000000	140593	2 из 100	98 из 100	1	0.9936028
		517998	1 из 100	99 из 100	3	
		997503	2 из 100	98 из 100	17	
		330504	1 из 100	99 из 100	2	
		198510	1 из 100	99 из 100	3	
		4112284	2 из 100	98 из 100	8	
		94028916	1 из 100	99 из 100	140	
		210861	1 из 100	99 из 100	1	
		40874	1 из 100	99 из 100	3	
		206405	1 из 100	99 из 100	4	
		11925263513	2 из 100	98 из 100	210	
		18607676768	0 из 100	100 из 100	340	
		1031871	1 из 100	99 из 100	9	
		783490	2 из 100	98 из 100	11	
		935366	1 из 100	99 из 100	15	
		179248	2 из 100	98 из 100	7	
		5112284	1 из 100	99 из 100	10	
		612791	1 из 100	99 из 100	5	
		176251	1 из 100	99 из 100	1	
		54308	2 из 100	98 из 100	2	
10	10000010010000001110000101100000110000011111100000000000111100000	140593	2 из 100	98 из 100	1	0.9907102
		517998	2 из 100	98 из 100	3	
		997503	2 из 100	98 из 100	17	
		330504	2 из 100	98 из 100	2	
		198510	2 из 100	98 из 100	3	
		4112284	2 из 100	98 из 100	8	
		94028916	1 из 100	99 из 100	140	
		210861	2 из 100	98 из 100	1	
		40874	1 из 100	99 из 100	3	
		206405	1 из 100	99 из 100	4	
		11925263513	2 из 100	98 из 100	210	
		18607676768	2 из 100	98 из 100	340	
		1031871	1 из 100	99 из 100	9	
		783490	2 из 100	98 из 100	11	
		935366	3 из 100	97 из 100	15	
		179248	2 из 100	98 из 100	7	
		5112284	2 из 100	98 из 100	10	
		612791	2 из 100	98 из 100	5	
		176251	1 из 100	99 из 100	1	
		54308	3 из 100	97 из 100	2	

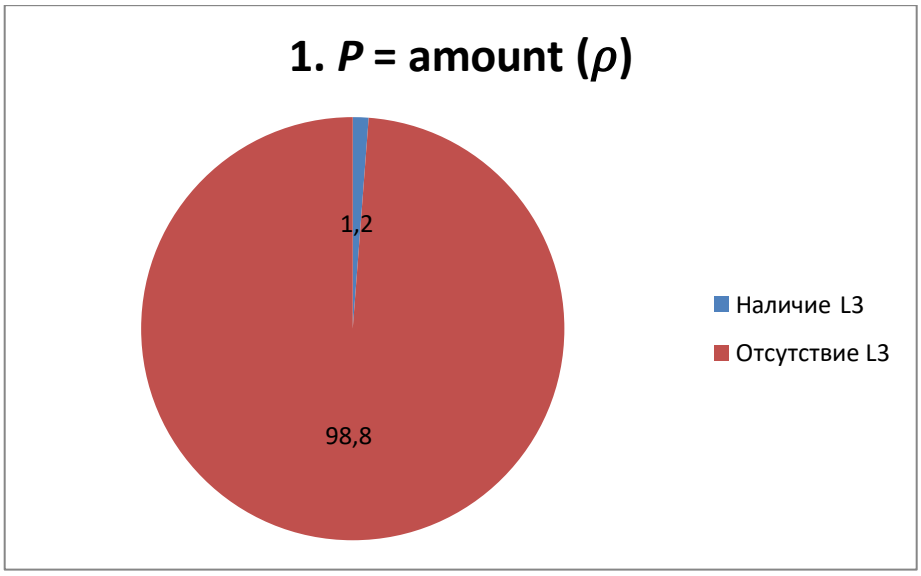

Рисунок Б.1 – Зависимость $P = amount\ (\rho)$ для случая №1 (вектора E №1)

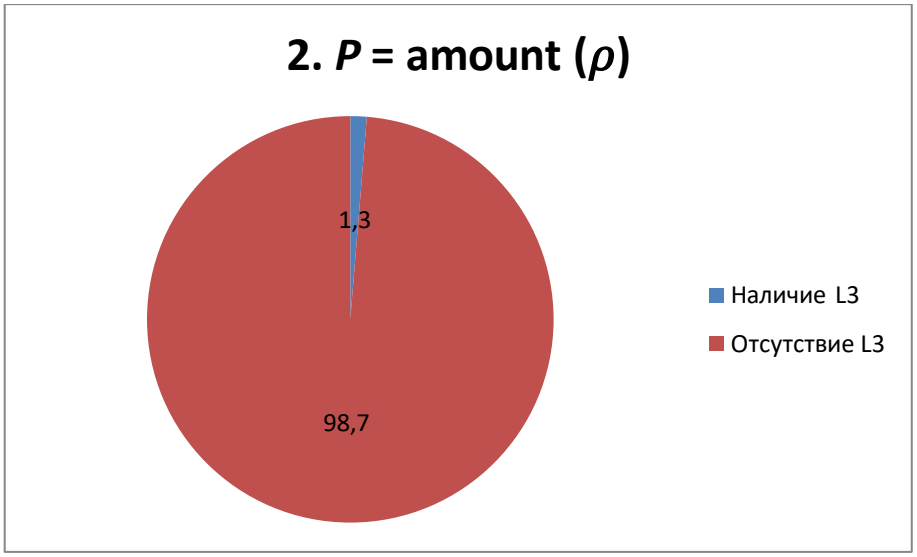

Рисунок Б.2 – Зависимость $P = amount\ (\rho)$ для случая №2 (вектора E №2)

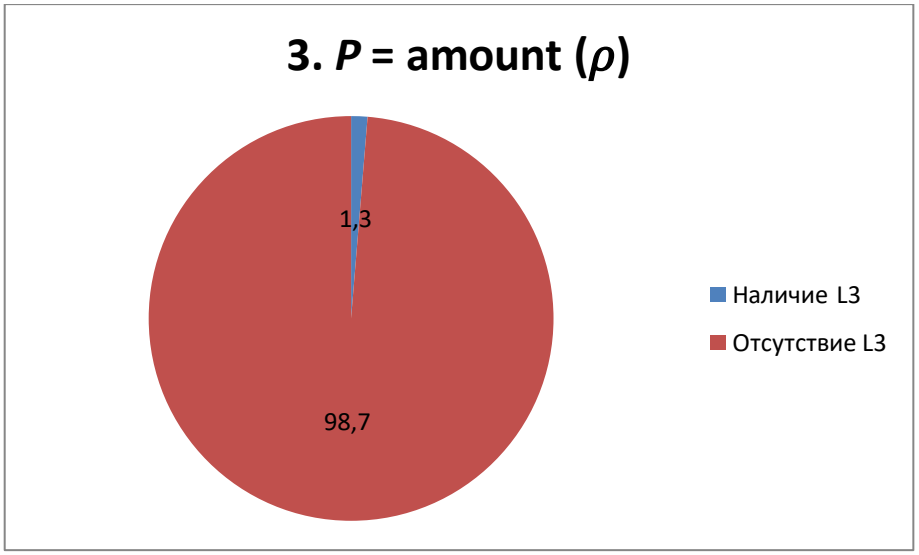

Рисунок Б.3 – Зависимость $P = amount\ (\rho)$ для случая №3 (вектора E №3)

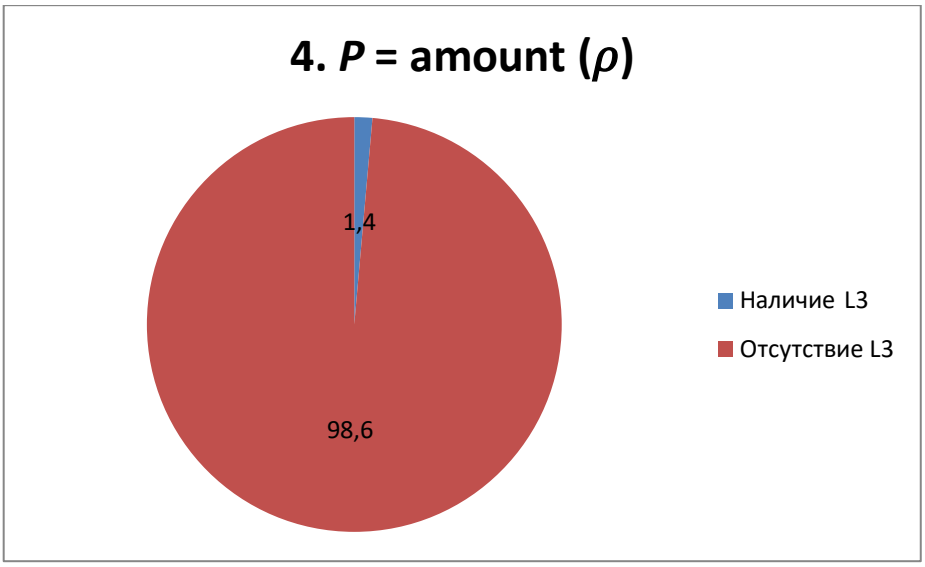

Рисунок Б.4 – Зависимость P = *amount* (ρ) для случая №4 (вектора E №4)

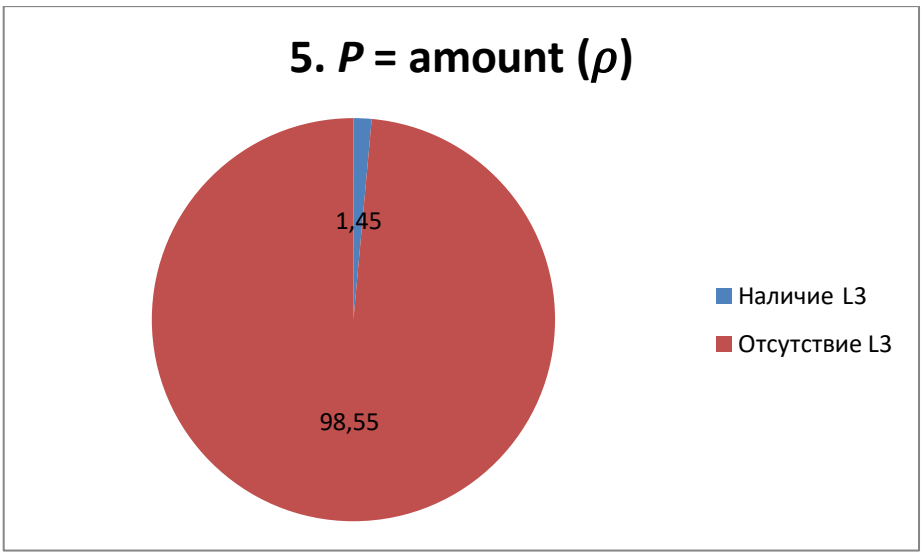

Рисунок Б.5 – Зависимость P = *amount* (ρ) для случая №5 (вектора E №5)

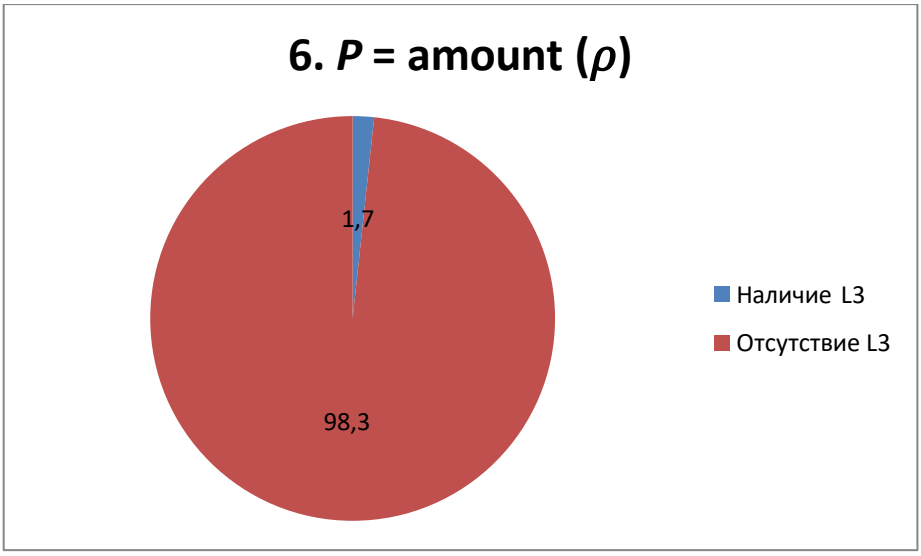

Рисунок Б.6 – Зависимость P = *amount* (ρ) для случая №6 (вектора E №6)

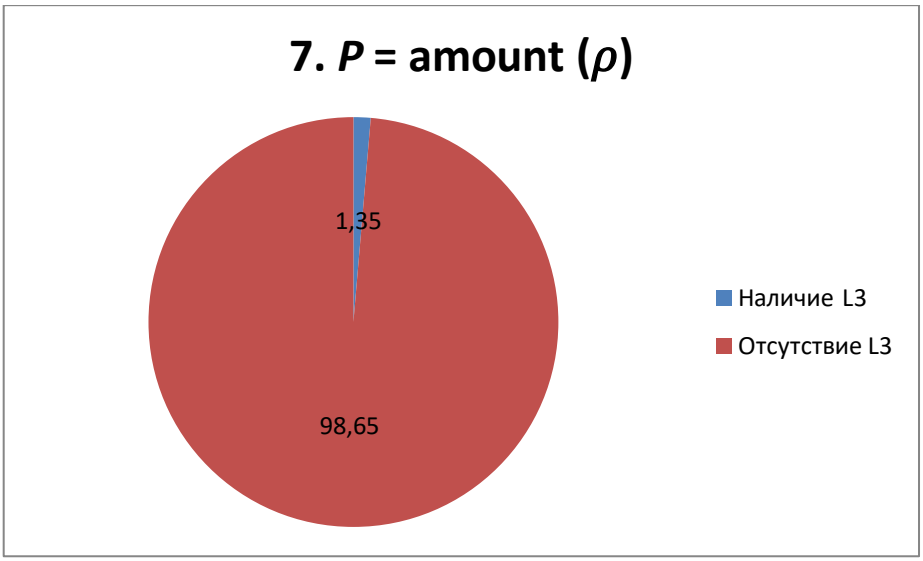

Рисунок Б.7 – Зависимость P = *amount* (ρ) для случая №7 (вектора E №7)

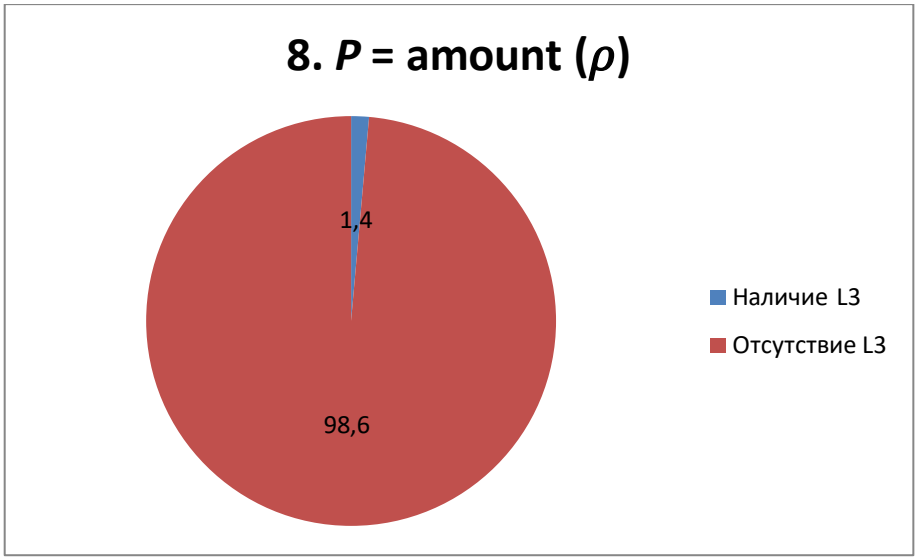

Рисунок Б.8 – Зависимость P = *amount* (ρ) для случая №8 (вектора E №8)

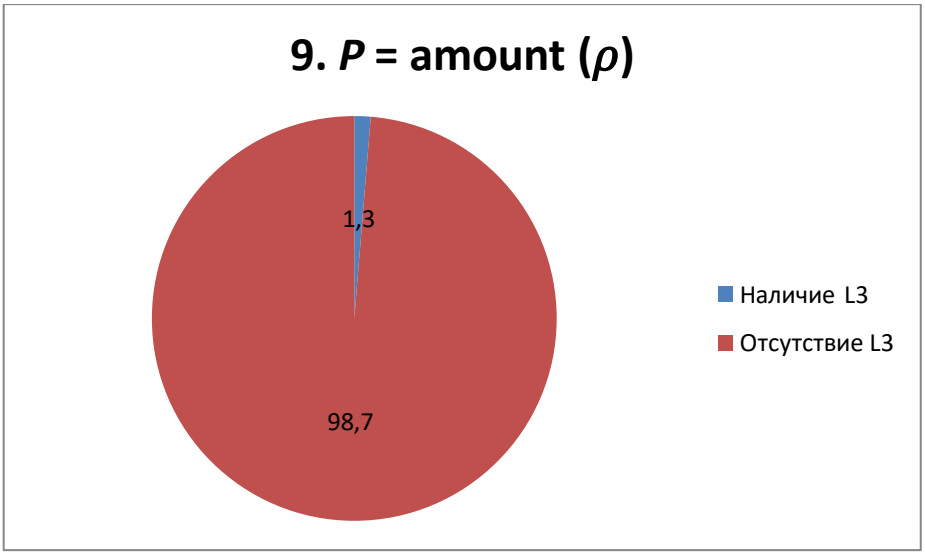

Рисунок Б.9 – Зависимость P = *amount* (ρ) для случая №9 (вектора E №9)

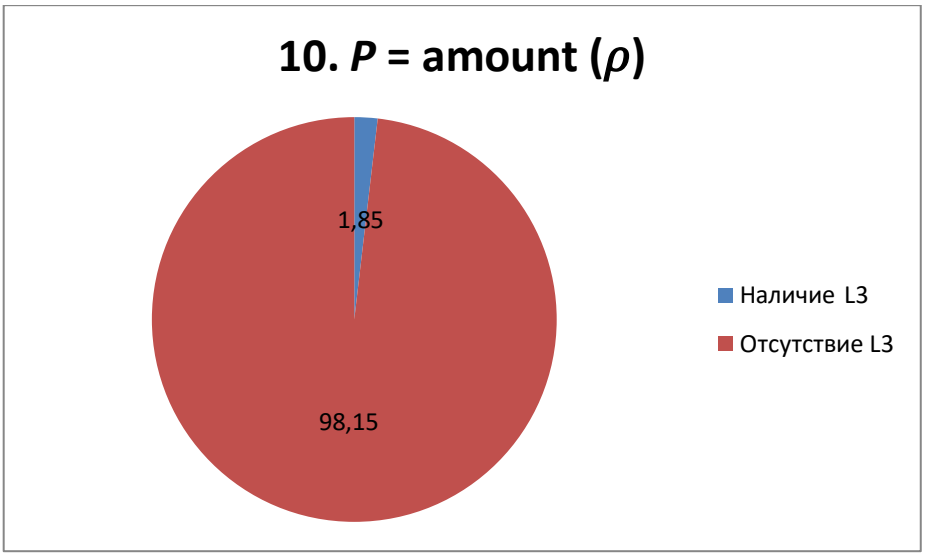

Рисунок Б.10 – Зависимость *P = amount (ρ)* для случая №10 (вектора *E* №10)

Приложение В. Статистическая зависимость вероятности успешной реализации L^3-атаки от плотности укладки рюкзака

Результаты экспериментальных данных по подсчету плотностей укладок для различных рюкзаков с общей памятью, полученных с помощью разработанного семейства алгоритмов шифрования и дешифрования (таблица В.1) и подсчету вероятностей успешной реализации L^3-атаки от плотности укладки рюкзака ρ при различных объемах исходного текста (таблица В.2).

Таблица В.1 - Подсчет плотностей укладки рюкзаков с общей памятью

Номер	Вектор E = {e1, e2, … , eB}	Значение ρ
1	11111111111111111111111111111111111111111111011101111101111011110	0.9989735
2	11011111111011011101110111111111110110111110111011101011111011110	0.9946581
3	11011111111011011101110111111111011011011011010101110101101011010	0.9940932
4	11011111111011010101110111111111011011001010100010101101010010	0.9933907
5	11011111111011010100110101100110111011010010101000101011010000	0.9925017
6	11011111111011010100110101100110111011010010101000100000000000000	0.9913402
7	11000011111011010100000101100110111011010010101000100000000000000	0.9985134
8	10000010011011010100000101100110000110110010101000100000000000000	0.99288584
9	10000010010000000000000101100000000000000000000000000000000000000	0.9936028
10	10000010010000001110000101100000110000011111100000000000111100000	0.9907102
11	11011100011011011101110111111111110111111110111111101011111111110	0.9981128
12	11010000011011011101110111111111011011011011011101111101111111110	0.9942135
13	11000000011011010100000101100110000110110010101000100000000011111	0.9939589
14	11011000001011010101110111111111011011011001010111110101101011110	0.9947205
15	11011011111101111110111111111111111111111111111101111111111111111	0.9981761

Продолжение таблицы В.1

Номер	Вектор E = {e1, e2, … , eB}	Значение ρ
16	1101100000101101010011010110011011011011001010110010101101011111	0.9923125
17	1101111111101101111111011100000011011011111011111111111111111111	0.9949875
18	1101000111101101110111011111111011011111111110101110101101011001	0.9949812
19	1101000001101101010011010110011011011011001010100010000000011111	0.9914219
20	0000001111101101111111011101111111011011111011111111111111111111	0.9948714
21	1111101111111101111111011111101111110111111011111111111111011110	0.9989853
22	1001111100101101110111011111111111011011111111111111101111011110	0.9947814
23	1011110111111111110111111111111111111111111111011101111111111011110	0.9989918
24	1011111111111111110111101111111111111111111111011101111111111011110	0.9997614
25	0001110111111111111111111111111111111111111111111111011111111111011110	0.9998912

Таблица В.2 - Определение вероятностей успешной L^3-атаки

Номер	Наличие / Отсутствие L3 (%)	
1	Наличие L3	1,20
	Отсутствие L3	98,80
2	Наличие L3	1,30
	Отсутствие L3	98,70
3	Наличие L3	1,30
	Отсутствие L3	98,70
4	Наличие L3	1,40
	Отсутствие L3	98,60
5	Наличие L3	1,45
	Отсутствие L3	98,55
6	Наличие L3	1,70
	Отсутствие L3	98,30
7	Наличие L3	1,35
	Отсутствие L3	98,65
8	Наличие L3	1,40
	Отсутствие L3	98,60
9	Наличие L3	1,30
	Отсутствие L3	98,70
10	Наличие L3	1,85
	Отсутствие L3	98,15
11	Наличие L3	1,30
	Отсутствие L3	98,70
12	Наличие L3	1,30
	Отсутствие L3	98,70
13	Наличие L3	1,35
	Отсутствие L3	98,65
14	Наличие L3	1,40
	Отсутствие L3	98,60
15	Наличие L3	1,20
	Отсутствие L3	98,80
16	Наличие L3	1,45
	Отсутствие L3	98,55
17	Наличие L3	1,30
	Отсутствие L3	98,70
18	Наличие L3	1,30
	Отсутствие L3	98,70

Продолжение таблицы В.2

Номер	Наличие / Отсутствие L3 (%)	
19	Наличие L3	1,70
	Отсутствие L3	98,30
20	Наличие L3	1,30
	Отсутствие L3	98,70
21	Наличие L3	1,20
	Отсутствие L3	98,80
22	Наличие L3	1,30
	Отсутствие L3	98,70
23	Наличие L3	1,10
	Отсутствие L3	98,90
24	Наличие L3	1,00
	Отсутствие L3	99,00
25	Наличие L3	1,00
	Отсутствие L3	99,00

Приложение Г. Листинг программного модуля приложения по дешифрованию с помощью жадного алгоритма

Выдержки листинга кода основных процедур приложения по дешифрованию заданных закрытых текстов *S* на конкретно сформированном базисе типа Фибоначчи с помощью жадного алгоритма в рамках разработанной вариации симметричной рюкзачной криптосистемы с общей памятью с комментариями.

1. Инициализация ключа и возрастающего базиса.

```
package com;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.DataInputStream;
import java.io.DataOutput;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.util.ArrayList;
import java.util.Collections;
import org.omg.CORBA.FREE_MEM;
import java.math.*;

public class main
{
private ArrayList<Long> seq_d;
private ArrayList<Long> seq_f;
```

```
private  int[] key;
```

2. Функция, возвращающая ключевую последовательность.

```
public int[] getKey()
{       return key;
}
```

3. Функция возвращающая возрастающий базис.

```
public ArrayList<Long> getSeq_f() {
        return seq_f;
}       /**
         * @param args
         */
public static void main (String[] args)
{               main m=new main();
                m.create_collection();
                boolean f = true;
                f = m.expand(304269);   // функция дешифрования сообщения
                System.out.println("Возможность дешифровки:");
                System.out.println(f);
                float density;
                density=m.density(m.getSeq_f());    //подсчет плотности укладки
                System.out.print("Плотность = ");
                System.out.println(density);
       m.write_key(m.getKey());
         // TODO Auto-generated method stub
         }
```

4. Формирование в соответствии с проектированным алгоритмом возрастающего базиса.

```
void create_collection()
{       int zero_count=0;
        try{
        seq_d=new ArrayList<Long>();
                FileInputStream fstream_seq = new FileInputStream("C:\\fib_posl.txt");   // считывание общей памяти
                FileInputStream fstream_key = new FileInputStream("C:\\key.txt");

                DataInputStream in_seq = new DataInputStream(fstream_seq);
                DataInputStream in_key = new DataInputStream(fstream_key);
                BufferedReader br_seq = new BufferedReader(new InputStreamReader(in_seq));
                BufferedReader br_key = new BufferedReader(new InputStreamReader(in_key));

                String strLine_key;

                while ((strLine_key = br_key.readLine()) != null)
     {
                        if (!(strLine_key.equals("0"))){
                                seq_d.add(Long.parseLong(br_seq.readLine()));
                        }
                        else if ((strLine_key.equals("0"))){zero_count++;
                                br_seq.readLine();
                        }
                }
                in_seq.close();
```

```
                in_key.close();
                Collections.sort(seq_d);
                  }
        catch (Exception e)
{
                System.err.println("Error: " + e.getMessage());
                }
        if(seq_d.size()!=0)
{       // алгоритм формирования возрастающего базиса
        seq_f=new ArrayList<Long>();
        seq_f.add((long) 1);
        seq_f.add(seq_d.get(0));
        for(int i=2;i<seq_d.size()+1;i++){
                seq_f.add(i, seq_d.get(i-1));
                for (int j=0;j<seq_f.size()-1;j++){
                        seq_f.set(i, seq_f.get(i)+seq_f.get(j));
                }
        }
}

        else
{
                seq_f=new ArrayList<Long>();
                int a=1;
                int b=1;
                int sum_fib;
                seq_f.add(0, (long) 1);
                seq_f.add(1, (long) 1);
                for(int i = 2; i < zero_count; i++){
               sum_fib = a + b;
```

```
            a = b;
            b = sum_fib;
            seq_f.add(i, (long) sum_fib);

            //System.out.print(sum_fib + "$#");
        }
    }

    System.out.println("Полученная последовательность: ");
    for (int i=0;i<=seq_f.size()-1;i++)
    {
            System.out.print(seq_f.get(i));
        System.out.print(" ");
    }
        System.out.println();
             }
```

5. Функция дешифровки сообщения с помощью «жадного алгоритма».

```
boolean expand(long number)
{
        key=new int [seq_f.size()];
        boolean flag=false;

        for(int i=seq_f.size()-1;i>=0;i--){

            if(number>seq_f.get(i)){
                    number=number-seq_f.get(i);
                    key[i]=1;
            }
```

```
                else if (number<seq_f.get(i))
{
                        key[i]=0;
                }
                else if(number==seq_f.get(i))
{
                        flag=true;

                                key[i]=1;
                                number=0;
                                System.out.println("D = "+number);

                        }
        }
           if (!flag)
{
                System.out.println("D = "+number);
           }
        return flag;
}
```

6. Запись ключа дешифровки в файл.

```
void write_key(int [] key)
{
         try{

                 File f=new File("C:\\razl.txt");

                  if(!f.exists())
                  {
```

```
f.createNewFile();
}
                FileOutputStream fitream = new FileOutputStream(f);
                DataOutputStream out_seq = new DataOutputStream(fitream);
                BufferedWriter bwr = new BufferedWriter(new
OutputStreamWriter(out_seq));
                String strLine_key;

                for(int i=0;i<key.length;i++){
                        strLine_key=Integer.toString(key[i]);
                        bwr.write(strLine_key);
                        bwr.newLine();
                }
                bwr.close();
                  }
        catch (Exception e)
{
                System.err.println("Error: " + e.getMessage());
                }
}
```

7. Подсчет плотности укладки рюкзака.

```
float density(ArrayList<Long> l)
{
        return (float) (l.size()/(Math.log(l.get(l.size()-1))/Math.log(2)));
}
}
```

Приложение Д. Листинг программного модуля приложения по шифрованию и дешифрованию файлов с помощью разработанных алгоритмов

Выдержки листинга кода основных процедур программного обеспечения по практической реализации разработанной вариации *CBC*-блочной симметричной рюкзачной криптосистемы в рамках криптографических двухсторонних протоколов с общей памятью с комментариями.

1. Инициализация среды.

```
unit Knapsack;
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ShlObj, DateUtils, ExtCtrls, IOUtils, ComCtrls;

type

  TMainForm = class(TForm)
    EncryptButton: TButton;
    OpenFileDialog: TOpenDialog;
    SaveFileDialog: TSaveDialog;
    OpenKeyDialog: TOpenDialog;
    SaveKeyDialog: TSaveDialog;
    DecryptButton: TButton;
    ProgressBar: TProgressBar;
    OpenShareDialog: TOpenDialog;
    DensityEdit: TEdit;
    RadioGroup: TRadioGroup;
```

```
    InfoLabel: TLabel;
    AboutButton: TButton;
    procedure EncryptButtonClick(Sender: TObject);
    procedure DecryptButtonClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure RadioGroupClick(Sender: TObject);
    procedure AboutButtonClick(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

const
  Capacity = 100;
  MAX_SIZE = 63;

var
  MainForm: TMainForm;
  f: File of Byte;
  g: File of Byte;
  key_file, shared_file : TextFile;
  Enc : array of Byte;
  Dec : array of Byte;
  Wre : array of Byte;
  Wrd : array of Byte;
  Key : array [0..MAX_SIZE] of Integer;
  Mem : array [0..MAX_SIZE] of Integer;
  Pos : array [0..MAX_SIZE] of Integer;
```

```
  Key_new : array of Integer;
  Key_new_d : array of Integer;
  Delta : array [0..MAX_SIZE] of Integer;
  Delta_new : array of Integer;
  i, j, Num, value, Len, k, count, temp, l, min : Integer;
  Result, name_file, ext_file, path_file, name_file_d, ext_file_d, path_file_d,
name_file_with_ext, name_file_new, ext_file_new : String;
  DesktopPidl : PItemIDList;
  DesktopPath : array [0..MAX_PATH] of Char;
  Density, size_memory, size_memory_r : Double;

implementation
{$R *.dfm}
```

2. Создание главной формы, очистка полей и отключение кнопок.

```
procedure TMainForm.FormCreate(Sender: TObject);
begin
RadioGroup.Enabled := false;
DensityEdit.Clear;
InfoLabel.Caption := '';
DensityEdit.Hint := '';
end;
```

3. Процедура шифрования выбранного файла.

```
procedure TMainForm.EncryptButtonClick(Sender: TObject);
begin

RadioGroup.Enabled := false;
```

```
DensityEdit.Clear;
InfoLabel.Caption := '';
DensityEdit.Hint := '';
```

4. Определение пути до рабочего стола пользователя.

```
SHGetSpecialFolderLocation(0, CSIDL_DESKTOP, DesktopPidl);
SHGetPathFromIDList(DesktopPidl, DesktopPath);
Result := DesktopPath;

ProgressBar.Position := 0;
```

5. Открытие диалога с выбором файла для шифрования.

```
OpenFileDialog.Title := 'Выберите файл для шифрования';
OpenFileDialog.InitialDir := Result;
OpenFileDialog.Filter := '';

if not ((OpenFileDialog.Execute) and (FileExists(OpenFileDialog.FileName))) then
begin
ShowMessage('Открытие файла прервано,' + #10#13 + 'либо выбранного файла не существует!');
ProgressBar.Position := 0;
Exit;
end;

AssignFile(f, OpenFileDialog.FileName);
Reset(f);
```

6. Определение пути к файлу, его имени и расширения.

```
ext_file := ExtractFileExt(OpenFileDialog.FileName);
path_file := ExtractFilePath(OpenFileDialog.FileName);

name_file:=
Copy(ExtractFileName(OpenFileDialog.FileName),1,(LastDelimiter(Chr(46),
      ExtractFileName(OpenFileDialog.FileName))-1));
ProgressBar.Position := 15;
```

7. Считывание файла в динамический массив.

```
i := 0;
  while not Eof(f) do begin
    if i = Length(Enc) then SetLength(Enc, i + Capacity);
    Read (f, Enc[i]);
    Inc(i);
    if i = Length(Enc) / 2 then
    ProgressBar.Position := 50;
  end;
```

8. Установление длин других динамических массивов.

```
  SetLength(Enc, i);
  SetLength(Wre, i);
  SetLength(Key_new, i);
  SetLength(Delta_new, i);
  CloseFile(f);
  OpenFileDialog.FileName := '';
```

9. Открытие диалога для считывания общей памяти.

```
OpenShareDialog.Title := 'Выберите файл общей памяти';
OpenShareDialog.InitialDir := Result;
OpenShareDialog.Filter := 'Knapsack shared memory files|*.shm';

if not ((OpenShareDialog.Execute) and (FileExists(OpenShareDialog.FileName)))
then
begin
ShowMessage('Открытие общей памяти прервано,' + #10#13 + 'либо выбранного файла не существует!');
ProgressBar.Position := 0;
Exit;
end;

AssignFile(shared_file, OpenShareDialog.FileName);
Reset(shared_file);
```

10. Считывание общей памяти в массив.

```
k := 0;
   while not Eof(shared_file) do begin
   Readln (shared_file, Mem[k]);
   Inc(k);
   end;

CloseFile(shared_file);
OpenShareDialog.FileName := '';

 for i := 0 to High(Mem) do
```

```
begin
  min := i;
  for j := i + l to High(Mem) do
    if Mem[j] < Mem[min] then
      min := j;
      temp := Mem[i];
      Mem[i] := Mem[min];
      Mem[min] := temp;
end;
```

11. Формирование необходимого базиса типа Фибоначчи - формирование рюкзака.

```
for i := 1 to High(Pos) do begin
  Pos[0] := 1;
  Pos[i] := Pos[i-1] + Mem[i-1];
end;

for i := 0 to High(Key) do begin
  Key[i] := Pos[i];
end;

for i := Low(Delta) to High(Delta) do begin
  Delta[i] := Pos[i] - Mem[i];
end;

i := 0;
while (i < Length(Key_new)) do begin
  for j := Low(Key) to High(Key) do begin
  if (i < Length(Key_new)) then begin
```

```
Key_new[i] := Key[j];
Inc(i);

end
else begin
break;
end;
end;
end;
i := 0;
while (i < Length(Delta_new)) do begin
  for j := Low(Delta) to High(Delta) do begin
  if (i < Length(Delta_new)) then begin

Delta_new[i] := Delta[j];
Inc(i);

end
else begin
break;
end;
end;
end;
```

12. Шифрование файла на заданном ключе.

```
for i := Low(Wre) to High(Wre) do begin
  Wre[i] := Enc[i] + Key_new[i] - Delta_new[i];
end;
```

```
ProgressBar.Position := 80;
```

13. Запись зашифрованного файла и ключа.

```
AssignFile(key_file, path_file + 'key.kpt');
Rewrite(key_file);

j := 0;
while not (j = Length(Key)) do
begin
Writeln(key_file, Key[j]);
Inc(j);
end;

CloseFile(key_file);

AssignFile(g, path_file + name_file + ext_file + '_encrypted' + '.cpt');
Rewrite(g);

j := 0;
while not (j = Length(Wre)) do
begin
Write(g, Wre[j]);
Inc(j);
end;

ProgressBar.Position := 100;

CloseFile(g);
```

```
ShowMessage('Файл зашифрован.' + #10#13 + 'Ключ шифрования и полученный файл помещены рядом с исходным!');

RadioGroup.Enabled := true;
DensityEdit.Clear;
InfoLabel.Caption := '';
DensityEdit.Hint := '';

end;
```

14. Процедура дешифрования выбранного файла.

```
procedure TMainForm.DecryptButtonClick(Sender: TObject);
begin

RadioGroup.Enabled := false;
DensityEdit.Clear;
InfoLabel.Caption := '';
DensityEdit.Hint := '';
```

15. Определение пути до рабочего стола пользователя.

```
SHGetSpecialFolderLocation(0, CSIDL_DESKTOP, DesktopPidl);
SHGetPathFromIDList(DesktopPidl, DesktopPath);
Result := DesktopPath;
ProgressBar.Position := 0;
```

16. Открытие диалога выбора файла для дешифрования.

```
OpenFileDialog.Title := 'Выберите файл для дешифрования';
OpenFileDialog.InitialDir := Result;
OpenFileDialog.Filter := 'Knapsack encrypted files|*.cpt';

if not ((OpenFileDialog.Execute) and (FileExists(OpenFileDialog.FileName))) then
begin
ShowMessage('Открытие файла прервано,' + #10#13 + 'либо выбранного файла не существует!');
ProgressBar.Position := 0;
Exit;
end;
AssignFile(f, OpenFileDialog.FileName);
Reset(f);
```

17. Определение пути к файлу, его имени и расширения.

```
ext_file_d := ExtractFileExt(OpenFileDialog.FileName);
path_file_d := ExtractFilePath(OpenFileDialog.FileName);
name_file_d:=Copy(ExtractFileName(OpenFileDialog.FileName),1,(LastDelimiter(Chr(46), ExtractFileName(OpenFileDialog.FileName))-1));

name_file_with_ext := Copy(name_file_d, 0 , Length(name_file_d) - 10);

ext_file_new := ExtractFileExt(name_file_with_ext);
name_file_new :=
Copy(ExtractFileName(name_file_with_ext),1,(LastDelimiter(Chr(46),
    ExtractFileName(name_file_with_ext))-1));
```

```
ProgressBar.Position := 15;
```

18. Считывание дешифруемого файла в динамический массив.

```
i := 0;
  while not Eof(f) do begin
    if i = Length(Dec) then SetLength(Dec, i + Capacity);
    Read (f, Dec[i]);
    Inc(i);
    if i = Length(Dec) / 2 then
    ProgressBar.Position := 50;
  end;
```

19. Установление длин других динамических массивов.

```
  SetLength(Dec, i);
  SetLength(Wrd, i);
  SetLength(Key_new_d, i);
  CloseFile(f);
  OpenFileDialog.FileName := '';
```

20. Открытие диалога для файла общей памяти.

```
OpenShareDialog.Title := 'Выберите файл общей памяти';
OpenShareDialog.InitialDir := Result;
OpenShareDialog.Filter := 'Knapsack shared memory files|*.shm';

if not ((OpenShareDialog.Execute) and (FileExists(OpenShareDialog.FileName)))
then
begin
```

```
ShowMessage('Открытие общей памяти прервано,' + #10#13 + 'либо выбранного файла не существует!');
ProgressBar.Position := 0;
Exit;
end;

AssignFile(shared_file, OpenShareDialog.FileName);
Reset(shared_file);
```

21. Считывание общей памяти в массив.

```
k := 0;
   while not Eof(shared_file) do begin
   Readln (shared_file, Mem[k]);
   Inc(k);
   end;
CloseFile(shared_file);
OpenShareDialog.FileName := '';

 for i := 0 to High(Mem) do
  begin
    min := i;
    for j := i + l to High(Mem) do
      if Mem[j] < Mem[min] then
        min := j;
        temp := Mem[i];
        Mem[i] := Mem[min];
        Mem[min] := temp;
  end;
```

```
 for i := 1 to High(Pos) do begin
   Pos[0] := 1;
   Pos[i] := Pos[i-1] + Mem[i-1];
 end;
 for i := 0 to High(Key) do begin
   Key[i] := Pos[i];
 end;

 for i := Low(Delta) to High(Delta) do begin
   Delta[i] := Pos[i] - Mem[i];
 end;
```

22. Открытие диалога и считывание ключевого файла.

```
OpenKeyDialog.Title := 'Выберите файл ключа для дешифрования';
OpenKeyDialog.InitialDir := Result;
OpenKeyDialog.Filter := 'Knapsack key files|*.kpt';

if not ((OpenKeyDialog.Execute) and (FileExists(OpenKeyDialog.FileName)))
then
begin
ShowMessage('Открытие ключа прервано,' + #10#13 + 'либо выбранного
файла не существует!');
ProgressBar.Position := 0;
Exit;
end;

AssignFile(key_file, OpenKeyDialog.FileName);
Reset(key_file);
```

```
value := 0;
   while not Eof(key_file) do begin
   Readln (key_file, Key[value]);
   Inc(value);
   end;

CloseFile(key_file);
OpenKeyDialog.FileName := '';

 i := 0;
 while (i < Length(Key_new_d)) do begin
   for j := Low(Key) to High(Key) do begin
   if (i < Length(Key_new_d)) then begin

 Key_new_d[i] := Key[j];
 Inc(i);

 end
 else begin
 break;
 end;
 end;
 end;

  i := 0;
 while (i < Length(Delta_new)) do begin
   for j := Low(Delta) to High(Delta) do begin
   if (i < Length(Delta_new)) then begin

 Delta_new[i] := Delta[j];
```

```
  Inc(i);
  end
  else begin
  break;
  end;
  end;
  end;
```

23. Дешифрование заданного файла.

```
  for i := Low(Wrd) to High(Wrd) do begin
    Wrd[i] := Dec[i] - Key_new_d[i] + Delta_new[i];
  end;
```

```
ProgressBar.Position := 80;
```

24. Запись дешифрованного файла.

```
AssignFile(g, path_file_d + name_file_new + '_decrypted' + ext_file_new);
Rewrite(g);

j := 0;
while not (j = Length(Wrd)) do
begin
Write(g, Wrd[j]);
Inc(j);
end;

ProgressBar.Position := 100;
CloseFile(g);
```

```
ShowMessage('Файл дешифрован.' + #10#13 + 'Полученный файл помещен рядом с исходным!');

RadioGroup.Enabled := true;
DensityEdit.Clear;
InfoLabel.Caption := '';
DensityEdit.Hint := '';
end;
```

25. Процедура вычисления необходимых оценок рюкзачной криптосистемы.

```
procedure TMainForm.RadioGroupClick(Sender: TObject);
begin
```

26. Вычисление плотности укладки полученной рюкзачной криптосистемы.

```
if (RadioGroup.ItemIndex = 0) then begin
Density := Length(Key) / (ln (Key[High(Key)]) / ln(2)) ;
DensityEdit.Text := Copy(FloatToStr(Density), 0, 6) + ' o.e.';
DensityEdit.Hint := 'Плотность укладки полученной симметричной рюкзачной криптографической системы';
InfoLabel.Caption := 'Плотность укладки полученного рюкзака:';
end;
```

27. Оценка общей памяти.

```
if (RadioGroup.ItemIndex = 1) then begin
size_memory := 0;
for i := Low(Mem) to High(Mem) do begin
```

```
size_memory := size_memory + Mem[i];
end;

size_memory_r := exp((0.31*Mem[High(Mem)])*ln(2));
DensityEdit.Text := FloatToStr(size_memory) + ' o.e.';
DensityEdit.Hint := 'Объем общей памяти между отправителем и получателем';
if (size_memory <= size_memory_r) then begin
ShowMessage('Объем общей памяти гарантирует плотность укладки более еденицы!');
InfoLabel.Caption := 'Гарантирована плотность укладки > 1';
end
else begin
ShowMessage('Объем общей памяти не гарантирует плотность укладки более еденицы!');
InfoLabel.Caption := 'Не гарантирована плотность укладки > 1';
end;
end;
```

28. Оценка необходимого объема общей памяти.

```
if (RadioGroup.ItemIndex = 2) then begin
size_memory_r := exp((0.31*Mem[High(Mem)])*ln(2));
DensityEdit.Text := FloatToStr(size_memory_r) + ' o.e.';
DensityEdit.Hint := 'Максимально допустимый объем общей памяти, когда плотность укладки > 1';
InfoLabel.Caption := 'Максимальный объем ОП, когда плотность > 1';
end;
end;
end.
```

Приложение Е. Копии актов о внедрении результатов диссертационного исследования

УТВЕРЖДАЮ
Генеральный директор ООО «Русский мастер»
Яшин С.А.

«18» февраля 2016 г.

Владимирская область, Судогодский район

АКТ
внедрения результатов работы
Метлинова Александра Дмитриевича
«Система безопасного хранения и передачи данных на основе симметричных рюкзачных криптографических систем с общей памятью»

Следующие результаты работы Метлинова А.Д. апробированы и используются в ООО «Русский мастер»:

1. Алгоритмы шифрования, дешифрования, хеширования конфиденциальной информации, а также алгоритмы генерации и передачи ключевой информации
2. Реализованная на основе предложенной криптосистемы с общей памятью система безопасного хранилища и передачи по открытым каналам связи конфиденциальной информации Trans Kept&Sec 1.3.
3. Авторские методики тестирования полученной защищенной среды передачи конфиденциальной информации: скоростная методика оценки работы защищенной среды; оценки объема общей памяти, минимально необходимого для ухода от *LLL*-атаки и т.д.

Применение этих результатов позволило безопасно хранить данные предприятия, которые составляют его коммерческую тайну, а также безопасно их передавать между подразделениями предприятия, не опасаясь хищения; снизить материальные затраты на активные средства по защите этих данных, облегчить и ускорить работу подразделений предприятия.

Генеральный директор ООО «Русский мастер» Яшин С.А.

Рисунок Д.1 – Акт внедрения в ООО «Русский мастер»

УТВЕРЖДАЮ
Директор ООО «ДИВАНиЯ»
Яшин С.А.
«20» марта 2017 г.

Владимирская область, Судогодский район

АКТ
внедрения результатов работы
Метлинова Александра Дмитриевича
«Система безопасного удаленного доступа к ценным информационным ресурсам предприятия на основе модификации проткала TLS с помощью CRC-блочной симметричной рюкзачной криптосистемы с общей памятью»

Следующие результаты диссертационной работы Метлинова А.Д. апробированы и используются в ООО «ДИВАНиЯ»:

1. Протокол TLS с модифицированными алгоритмами аутентификации пользователей (удаленных узлов) и сервера, шифрования, дешифрования, хеширования информационным ресурсов предприятия.
2. Реализованная на основе предложенной модификации протокола TLS система безопасного удаленного доступа к ценным информационным ресурсам предприятия Divaniya_SecAccess 1.2.

Применение этих результатов позволило организовать безопасный удаленный доступ и переду ценных информационных ресурсов предприятия между подразделениями; снизить до минимума материальные затраты на активные средства по защите этих данных, ускорить работу с этими данными в 3 раза. Снижено количество инцидентов по взлому сервером с этими данными на 95%, ускорена работа удаленных пользователей с серверами в 2 раза, время на аутентификацию каждого пользователя сокращено с минуты до десяти секунд.

Директор ООО «ДИВАНиЯ» Яшин С.А.
Главный инженер ИТР Шепель О.В.

Рисунок Д.2 – Акт внедрения в ООО «Дивания»

УТВЕРЖДАЮ
ИП Щерба А.Ю.
«21» февраля 2017 г.

Владимирская область, город Владимир, Большой проезд, 15а

АКТ

внедрения результатов работы

Метлинова Александра Дмитриевича

«Система безопасного хранения данных на основе блочной рюкзачной криптосистемы с pre-shared memory»

Следующие результаты работы Метлинова А.Д. апробированы и используются в ИП:

1. Алгоритмы шифрования, дешифрования, хеширования конфиденциальной информации, составляющей коммерческую тайну ИП.
2. Реализованная на основе предложенной криптосистемы с pre-shared memory система безопасного хранилища конфиденциальной информации Trans Kept&Sec 1.3.1 (build_for_Tsherba).

Применение этих результатов позволило безопасно хранить данные индивидуального предпринимателя, которые составляют его коммерческую тайну и максимально снизить материальные затраты на активные средства по защите этих данных от внешних злоумышленников.

ИП Щерба А.Ю. Щерба А.Ю.

Руководитель разработки Бубело Д.С.

Рисунок Д.3 – Акт внедрения в ИП Щерба А.Ю


РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2014614981

Симметричная рюкзачная криптографическая система (вариация с плотностью укладки более единицы).

Правообладатель: Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых» (RU)

Авторы: Метлинов Александр Дмитриевич (RU), Александров Алексей Викторович (RU)

Заявка № 2014612250
Дата поступления 18 марта 2014 г.
Дата государственной регистрации в Реестре программ для ЭВМ 14 мая 2014 г.

Руководитель Федеральной службы по интеллектуальной собственности К.П. Семенов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2014614937

Тестовый комплекс для симметричной рюкзачной криптографической системы (вариация с плотностью укладки более единицы).

Правообладатель: Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых» (ВлГУ) (RU)

Авторы: Метлинов Александр Дмитриевич (RU), Александров Алексей Викторович (RU)

Заявка № 2014612247
Дата поступления 18 марта 2014 г.
Дата государственной регистрации в Реестре программ для ЭВМ 14 мая 2014 г.

Руководитель Федеральной службы по интеллектуальной собственности К.П. Семенов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2015616165

Генератор общей памяти для симметричной рюкзачной криптографической системы.

Правообладатель: Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых» (RU)

Авторы: Метлинов Александр Дмитриевич (RU), Александров Алексей Викторович (RU)

Заявка № 2015612822
Дата поступления 10 апреля 2015 г.
Дата государственной регистрации в Реестре программ для ЭВМ 02 июня 2015 г.

Врио руководителя Федеральной службы по интеллектуальной собственности Л.Л. Кирий

Рисунок Д.4 – Свидетельства о регистрации прав на разработанное ПО