МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования
«Самарский национальный исследовательский университет имени академика С.П. Королева»
(Самарский университет)

Институт информатики и кибернетики
Кафедра геоинформатики и информационной безопасности

ОТЧЕТ ПО ПРАКТИКЕ

Вид практики: учебная
(учебная, производственная)

Тип практики: Экспериментально-исследовательская практика

Сроки прохождения практики: с 20.06.2025 г. по 03.07.2025 г.
по направлению подготовки 10.05.03 Информационная безопасность автоматизированных систем
(уровень академического специалитета)
направленность (профиль) «Безопасность открытых информационных систем»

Обучающийся группы № 6313-100503D Чан Куанг Минь

Руководитель практики от университета профессор, д.т.н. Сергеев В.В.

Дата сдачи 03.07.2025 г.
Дата защиты 03.07.2025 г.

Оценка _____

Самара 2025

СОДЕРЖАНИЕ

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования
«Самарский национальный исследовательский университет имени академика С.П. Королева»
(Самарский университет)

Институт информатики и кибернетики
Кафедра геоинформатики и информационной безопасности

Индивидуальное задание на практику

Студенту группы № 6313-100503D Чан Куанг Минь
Направление на практику оформлено приказом по университету №307-ПР
от 11.06.2025 г. на
кафедру геоинформатики и информационной безопасности Самарского университета
(наименование профильной организации или структурного подразделения университета)

Планируемые результаты освоения образовательной программы (компетенции)	**Выполнение определенных видов работ, связанных с будущей профессиональной деятельностью (сбор и анализ данных и материалов, проведение исследований)**	**Результаты практики**
ОПК-1	Способен оценивать роль информации, информационных технологий и информационной безопасности в современном обществе, их значение для обеспечения объективных потребностей личности, общества и государства	Проведен анализ роли симметричного шифрования и информационной безопасности при передаче данных, а также влияние искажения данных на результат дешифровки.
ОПК-2	Способен применять программные средства системного и прикладного назначений, в том числе отечественного производства, для решения задач профессиональной деятельности	Реализованы программные средства на Python с использованием библиотеки cryptography для шифрования данных в различных режимах Camellia и моделирования ошибок передачи.
ОПК-3	Способен использовать математические методы, необходимые для решения задач профессиональной деятельности	Построены графики зависимости числа поврежденных блоков открытого текста от числа поврежденных блоков шифртекста для различных режимов.
ОПК-4	Способен анализировать физическую сущность явлений и процессов, лежащих в основе функционирования микроэлектронной техники, применять основные физические законы и модели для решения задач профессиональной деятельности	В рамках анализа механизмов распространения ошибок в блочных режимах шифрования рассмотрены аналогии с физическими процессами в цифровых системах передачи данных.
ОПК-9	Способен решать задачи профессиональной деятельности с учетом текущего состояния и тенденций развития информационных технологий, средств технической защиты информации, сетей и систем передачи информации	Изучены современные средства защиты информации на примере алгоритма Camellia и его применения в реальных системах защиты данных.
ОПК-14	Способен осуществлять разработку, внедрение и эксплуатацию автоматизированных систем с учетом требований по защите информации, проводить подготовку исходных данных для технико-экономического обоснования проектных решений	Описан и реализован эксперимент по анализу устойчивости автоматизированной системы шифрования к ошибкам передачи, предложены рекомендации по выбору режима.

ОПК-15	Способен осуществлять администрирование и контроль функционирования средств и систем защиты информации автоматизированных систем, инструментальный мониторинг защищенности автоматизированных систем	Проведено моделирование и анализ защищенности шифрованных данных при искажениях, что способствует формированию навыков оценки устойчивости систем к атакам и сбоям.
ОПК-16	Способен анализировать основные этапы и закономерности исторического развития России, ее место и роль в контексте всеобщей истории, в том числе для формирования гражданской позиции и развития патриотизма	Проанализирована важность защиты информации в контексте цифровизации общества и её влияние на информационную безопасность в государстве.

Срок предоставления на кафедру отчета о практике: 03.07.2025 г.

Руководитель практики от университета, д.т.н., профессор _____ В.В.Сергеев

(подпись)

Руководитель практики от профильной организации _____ А.И. Максимов

(подпись)

Задание принял к исполнению обучающийся группы № 6313-100503D _____ Чан Куанг Минь

(подпись)

О Т Ч Е Т

о выполнении индивидуального задания

по экспериментально-исследовательской практике

ВВЕДЕНИЕ

В условиях стремительного развития информационных технологий вопрос безопасности данных становится особенно актуальным. Современные системы связи требуют обеспечения конфиденциальности и целостности информации как при её хранении, так и при передаче. Одним из наиболее эффективных и широко применяемых методов защиты данных является симметричное блочное шифрование. Среди таких алгоритмов выделяется Camellia, сочетающий в себе высокую производительность и высокий уровень криптостойкости.

Целью настоящей учебной практики является изучение алгоритма шифрования Camellia и экспериментальное исследование его работы в различных режимах: CBC, CFB, ECB, и OFB.

В рамках индивидуального задания мне было предложено выполнить следующие задачи:

- Подготовить текстовый корпус для шифрования;
- Реализовать алгоритм Camellia с использованием различных режимов шифрования;
- Внести искажения в шифротекст с целью имитации ошибок передачи;
- Провести расшифровку повреждённых данных и оценить последствия;
- Определить количественную зависимость между числом повреждённых блоков шифротекста и исходного текста;
- Построить графики и обобщить полученные результаты;

1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 Теоретическая справка по симметричному блочному шифрованию

Блочное симметричное шифрование является одним из важнейших методов защиты информации. Оно работает с фиксированными по размеру блоками данных (обычно 64 или 128 бит), и использует один и тот же ключ как для шифрования, так и для дешифрования. Одной из особенностей блочного шифрования является возможность использовать различные режимы работы (например, ECB, CBC, CFB, OFB, CTR), которые позволяют адаптировать алгоритм к различным задачам и повысить безопасность.

Современные алгоритмы блочного симметричного шифрования основаны на сложных криптографических структурах, таких как сети Фейстеля (Feistel network) и сети подстановок и перестановок (SPN — Substitution-Permutation Network). Они обеспечивают высокую стойкость к криптоанализу и обладают эффектом лавины, при котором изменение одного бита сильно влияет на результат.

К наиболее известным алгоритмам блочного шифрования относятся:

- AES (Advanced Encryption Standard) — современный стандарт, широко применяемый во всём мире;
- DES (Data Encryption Standard) — устаревший, но исторически важный алгоритм;
- 3DES (Triple DES) — улучшенная версия DES;
- Camellia — японский алгоритм, сертифицированный международными стандартами;
- Blowfish и Twofish — свободные алгоритмы, популярные в открытом программном обеспечении;
- IDEA, Serpent, CAST5, и другие.

1.2 Теоретическая справка по алгоритму Camellia

1.2.1 Общая информация

Алгоритм Camellia — это современный блочный симметричный криптографический алгоритм, разработанный компаниями Mitsubishi Electric и NTT **(**Nippon Telegraph and Telephone**)** в 2000 году. Он был стандартизирован международными организациями ISO/IEC, а также включён в рекомендации IETF

наряду с алгоритмом AES.

Camellia предназначен для обеспечения высокого уровня безопасности и подходит как для программной, так и для аппаратной реализации. Благодаря открытой лицензии, он может использоваться свободно в коммерческих и некоммерческих решениях.

Camellia активно применяется в банковских системах, облачных хранилищах данных, встроенных устройствах (включая IoT) и считается одной из самых надёжных альтернатив AES в современных условиях.

1.2.2 Криптографическая структура и принцип работы

Алгоритм Camellia построен на основе сбалансированной сети Фейстеля с включением современных криптографических компонентов, обеспечивающих высокую степень безопасности и устойчивость к различным видам криптоанализа. Размер обрабатываемого блока составляет 128 бит, а поддерживаемая длина ключа — 128, 192 или 256 бит. В зависимости от длины ключа используется соответствующее количество раундов: 18, 24 или 32.

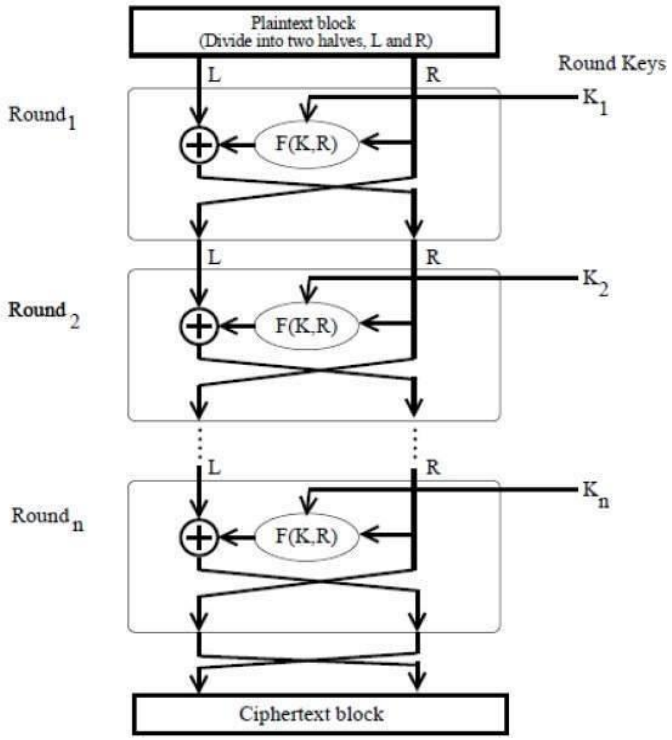

Рисунок 1 – Структура шифра Фейстеля

На начальном этапе блок данных разделяется на две равные части. Перед входом в раундовую часть осуществляется предварительное наложение ключа (key whitening), при котором данные смешиваются с производными ключами. Далее следует серия раундов, в ходе которых половины данных последовательно

обрабатываются с помощью раундовой функции, включающей в себя S-блоки — нелинейные подстановки, и линейные преобразования — P-функции. Эти компоненты обеспечивают запутанность (confusion) и диффузию (diffusion), что критически важно для криптостойкости алгоритма.

Особенностью архитектуры Camellia является наличие дополнительных функций FL и FL^{-1}, которые применяются на определённых этапах и усиливают внутреннюю структуру шифра. После прохождения всех раундов выполняется финальное наложение ключа, после чего обе половины объединяются в единый зашифрованный блок.

Процесс расшифрования в Camellia во многом аналогичен процессу шифрования, благодаря использованию симметричной структуры сети Фейстеля. Основным отличием является порядок применения раундовых ключей: при расшифровке они используются в обратной последовательности, по сравнению с шифрованием. Так как каждая операция внутри раунда (включая функции S-блоков, перестановок, FL/FL^{-1} и XOR) является обратимой, восстановление исходного текста возможно без потерь. Сначала к зашифрованному блоку применяется финальное наложение ключа (тот же шаг, что и при шифровании, но в обратную сторону). Затем выполняется последовательность раундов с теми же функциями, но с ключами в обратном порядке. В завершение используется начальное наложение ключа (key whitening), что даёт в итоге исходный открытый текст.

1.3 Теоретическая справка по режимам работы блочных шифров

Блочные шифры применяются для защиты информации путём преобразования исходных данных фиксированными по размеру блоками. Однако в реальных условиях данные редко соответствуют строго определённой длине блока. Для шифрования сообщений произвольной длины используются различные режимы работы блочных шифров, каждый из которых обладает своими особенностями в плане безопасности, устойчивости к ошибкам и производительности. В данной работе рассматриваются четыре классических режима: ECB, CBC, CFB и OFB.

1.3.1 Режим ECB (Electronic Codebook)

В режиме ECB процесс шифрования осуществляется путём разбиения всего открытого текста на блоки фиксированной длины (как правило, 128 бит), каждый из которых обрабатывается отдельно с использованием одного и того же алгоритма и ключа. Независимая обработка блоков не только упрощает реализацию схемы, но и позволяет выполнять параллельное шифрование, что значительно повышает производительность системы, особенно в условиях ограниченных вычислительных ресурсов.

Процесс расшифрования организован аналогичным образом: каждый блок шифротекста обрабатывается отдельно тем же самым алгоритмом, но с применением обратного преобразования. Поскольку блоки не связаны между собой, для расшифровки не требуется дополнительная информация — достаточно иметь доступ к ключу и зашифрованным данным.

Однако такое разделение блоков и отсутствие взаимосвязи между ними создают серьёзную уязвимость. Если два блока открытого текста совпадают, то и соответствующие блоки шифротекста будут идентичны. Это может позволить злоумышленнику выявить повторяющиеся структуры и сделать выводы о содержимом исходных данных. Именно по этой причине, несмотря на простоту и техническую эффективность, режим ECB не обеспечивает должного уровня безопасности и не рекомендуется для защиты чувствительной информации

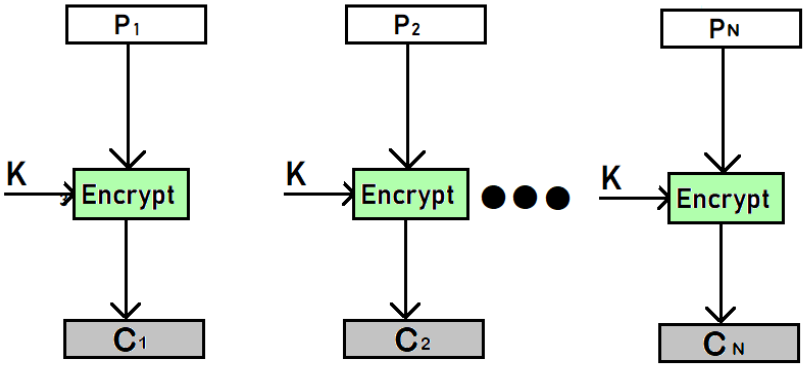

Рисунок 2 – Схема режима шифрования ECB

1.3.2 Режим CBC (Cipher Block Chaining)

Режим сцепления блоков шифротекста (CBC) был разработан для устранения главного недостатка режима ECB — его детерминированности, при которой одинаковые блоки открытого текста всегда приводят к одинаковым результатам шифрования. В CBC данный недостаток устраняется за счёт цепного

связывания блоков: каждый новый блок открытого текста перед шифрованием объединяется побитово по XOR с предыдущим зашифрованным блоком.

Процесс шифрования осуществляется таким образом, что каждый блок открытого текста сначала объединяется по XOR с предыдущим блоком шифротекста (или с IV в случае первого блока), после чего результат подаётся на вход блочного шифра, где он преобразуется с использованием секретного ключа. Полученный выход и есть очередной блок шифротекста, который далее участвует в обработке следующего блока.

Расшифровка в режиме CBC происходит в обратном порядке, но по тем же принципам. Каждый блок шифротекста сначала проходит расшифрование при помощи того же секретного ключа, а затем результат объединяется по XOR с предыдущим блоком шифротекста (или с IV для первого блока). Это позволяет восстановить исходный текст без потерь, при условии корректности всех входных данных.

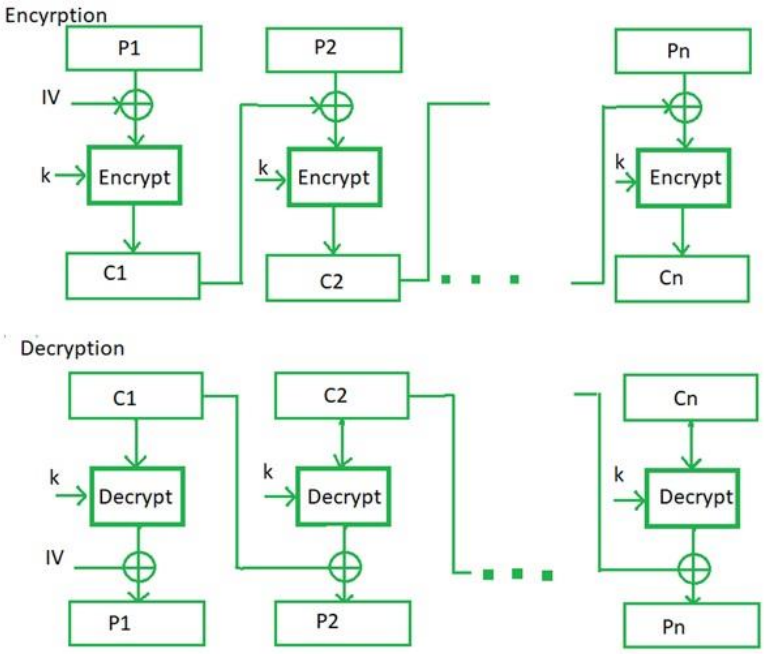

Рисунок 3 – Шифрование и расшифрование в режиме сцепления блоков (CBC)

Таким образом, CBC надёжно маскирует повторяющиеся структуры в данных, повышает криптостойкость и делает анализ шифротекста значительно более сложным. Однако этот режим не поддерживает параллельную обработку, так как каждый блок зависит от предыдущего, что ограничивает производительность. Кроме того, повреждение одного блока шифротекста приводит к искажению не только текущего, но и следующего блока при расшифровке, что также следует учитывать при применении CBC на практике.

1.3.3 Режим CFB (Cipher Feedback Mode)

Режим обратной связи по шифротексту (Cipher Feedback Mode, CFB) представляет собой способ использования блочного шифра в потоковом режиме, что делает его особенно удобным для обработки данных, поступающих последовательно — по байтам или блокам. Принцип работы CFB основан на использовании зашифрованного значения предыдущего блока шифротекста в качестве входа для формирования следующего шифровального потока.

Процесс шифрования начинается с того, что вектор инициализации (IV) шифруется с использованием основного ключа, а затем результат объединяется по операции XOR с первым блоком открытого текста. Полученный шифротекст становится входом для следующего раунда, где он вновь шифруется и используется для XOR со следующим блоком данных. Такая схема повторяется до окончания всего сообщения.

Расшифровка в режиме CFB по своей логике идентична шифрованию, поскольку тот же процесс применяется, но на вход подаются уже зашифрованные блоки. В каждом раунде шифруется предыдущий блок шифротекста, а затем результат XOR-ится с текущим шифротекстом, восстанавливая исходный открытый текст. Благодаря этому для расшифровки не требуется отдельного алгоритма, что упрощает реализацию.

CFB активно используется в системах, требующих шифрования в реальном времени, таких как передача данных по сетевым каналам, поскольку не требует предварительной буферизации всего сообщения. Он позволяет выполнять как шифрование, так и расшифровку по мере поступления данных, что критично для потоковых приложений. Тем не менее, режим CFB обладает высокой чувствительностью к ошибкам. Повреждение одного блока шифротекста приводит к искажению не только текущего блока расшифровки, но и следующего, что снижает надёжность при работе в условиях нестабильных каналов связи. Для обеспечения целостности данных могут потребоваться дополнительные механизмы контроля или аутентификации.

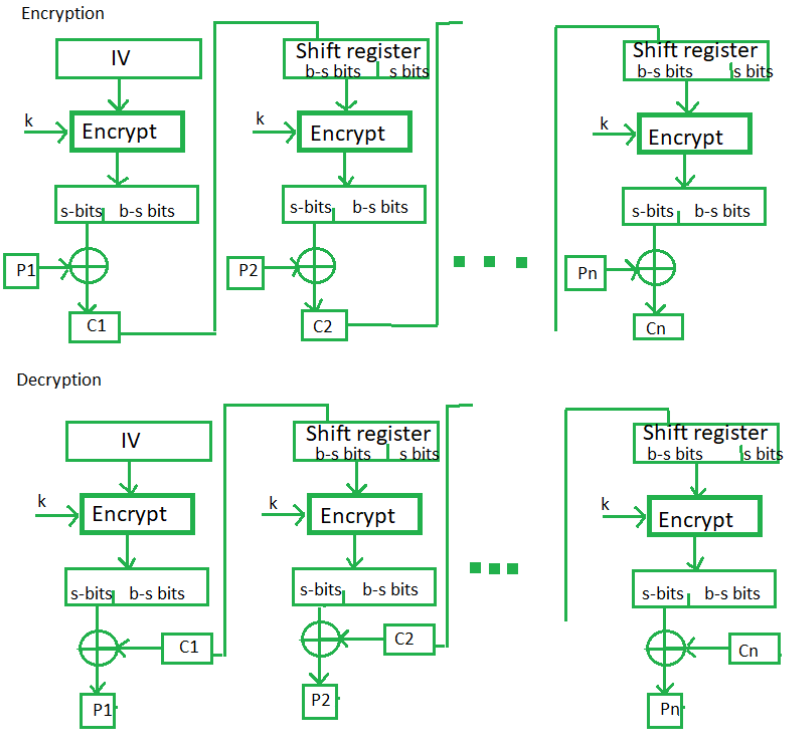

Рисунок 4 – Схема шифрования и расшифрования в режиме CFB

1.3.4 Режим OFB (Output Feedback Mode)

Режим обратной связи по выходу (OFB, Output Feedback Mode) представляет собой потоковый режим работы блочного шифра, схожий по структуре с режимом CFB, но имеющий принципиальные отличия в способе формирования шифрующего потока.

В отличие от режима CFB, в котором на вход шифра подаётся предыдущий шифротекст, в OFB шифруется результат предыдущей операции шифрования (начиная с вектора инициализации IV), независимо от текста сообщения. Полученные выходные блоки используются как ключевой поток, который поочерёдно объединяется по операции XOR с блоками открытого текста для получения шифротекста.

Такая схема позволяет полностью устранить влияние ошибок передачи: если в процессе передачи искажается один бит шифротекста, это приводит лишь к искажению одного соответствующего бита в расшифрованном сообщении, без распространения ошибки на следующие блоки, как это происходит в CBC или CFB. Благодаря этому OFB может быть полезен в условиях нестабильных или шумных каналов связи.

Однако режим OFB требует строгой синхронизации между отправителем и получателем: даже небольшое рассогласование во входных данных приведёт к

полной потере корректности расшифровки. Кроме того, как и другие потоковые режимы, он уязвим к повторному использованию одного и того же IV с одинаковым ключом — это может привести к утечке информации.

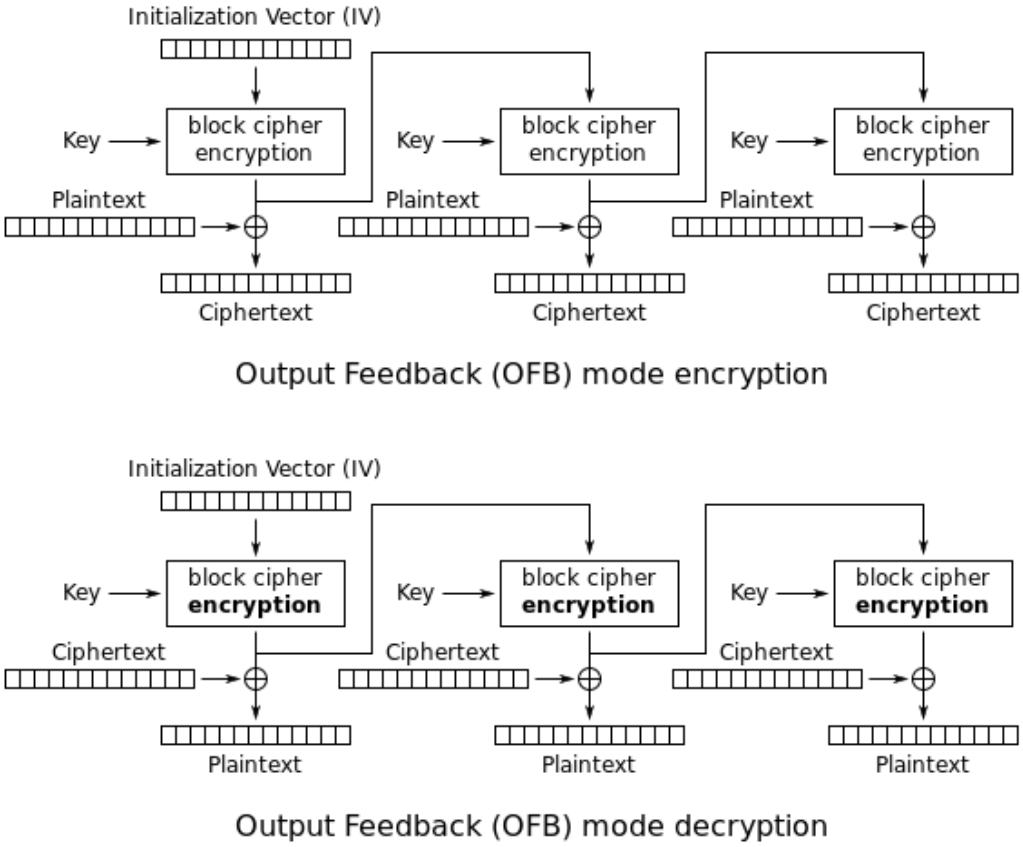

Рисунок 5 – Схема шифрования и расшифрования в режиме обратной связи по выходу (OFB)

2 ОПИСАНИЕ ПОСТАВЛЕННОГО ЭКСПЕРИМЕНТА

2.1 Цель эксперимента

Оценить влияние ошибок в шифротексте на целостность расшифрованного текста при использовании различных режимов работы алгоритма Camellia: ECB, CBC, CFB и OFB.

2.2 Этапы проведения эксперимента

1. Выбор алгоритма шифрования: В качестве шифра был выбран симметричный блочный алгоритм Camellia, поддерживающий работу в нескольких режимах.

2. Подготовка исходных данных: В качестве открытого текста был использован текстовый файл plaintext.txt. Данные были разбиты на блоки длиной 16 байт (размер блока Camellia).

3. Шифрование: Открытый текст был зашифрован с использованием алгоритма Camellia в четырёх режимах: ECB, CBC, CFB и OFB. Для каждого режима использовались одинаковые сессионные ключи и векторы инициализации, сгенерированные случайным образом.

4. Имитация ошибок передачи: Для моделирования искажений, возникающих при передаче по каналам связи, в зашифрованный текст были внесены случайные ошибки: от 1 до 10 блоков были повреждены путём инверсии одного байта в каждом выбранном блоке.

5. Расшифровка и анализ: Повреждённые шифротексты были расшифрованы, и проведён подсчёт количества блоков расшифрованного текста, отличающихся от оригинала. Это значение обозначается как M при заданном количестве искажённых блоков N.

6. Визуализация результатов: Для каждого режима были построены графики зависимости M=f(N), иллюстрирующие характер распространения ошибок. Графики сохранены в папке results/.

2.3 Структура программного проекта

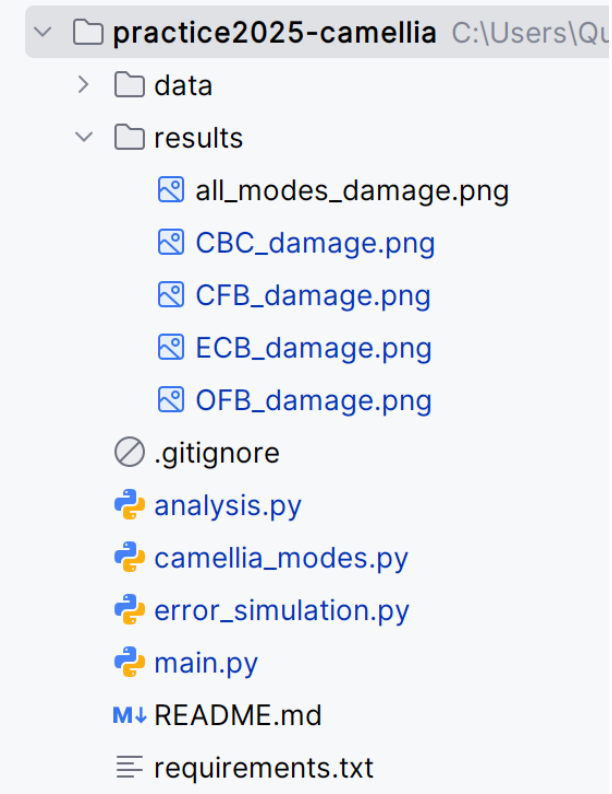

Рисунок 6 – Структура проекта Camellia

Для реализации и автоматизации эксперимента был разработан программный проект, включающий в себя следующие основные компоненты:

- Каталог data/: используется для хранения открытого текста (plaintext), предназначенного для шифрования.
- Каталог results/: содержит результаты эксперимента.
- Файл analysis.py: визуализирует результаты, строя и сохраняя графики ошибок для разных режимов шифрования.
- Файл camellia_modes.py: содержит реализацию блочного шифра Camellia в различных режимах работы (ECB, CBC, CFB, OFB) с использованием библиотеки cryptography.
- Файл error_simulation.py: отвечает за внесение искусственных ошибок в шифротекст для моделирования нестабильных условий передачи данных.
- Файл main.py: основной управляющий скрипт. Последовательно вызывает шифрование, имитацию ошибок, расшифровку и анализ, обеспечивая полный цикл эксперимента.

2.4 Результаты эксперимента

В ходе эксперимента были построены графики зависимости количества повреждённых блоков открытого текста (M) от количества искажённых блоков шифртекста (N) при использовании различных режимов блочного шифра Camellia: ECB, CBC, CFB, OFB. Результаты приведены на рисунке ниже.

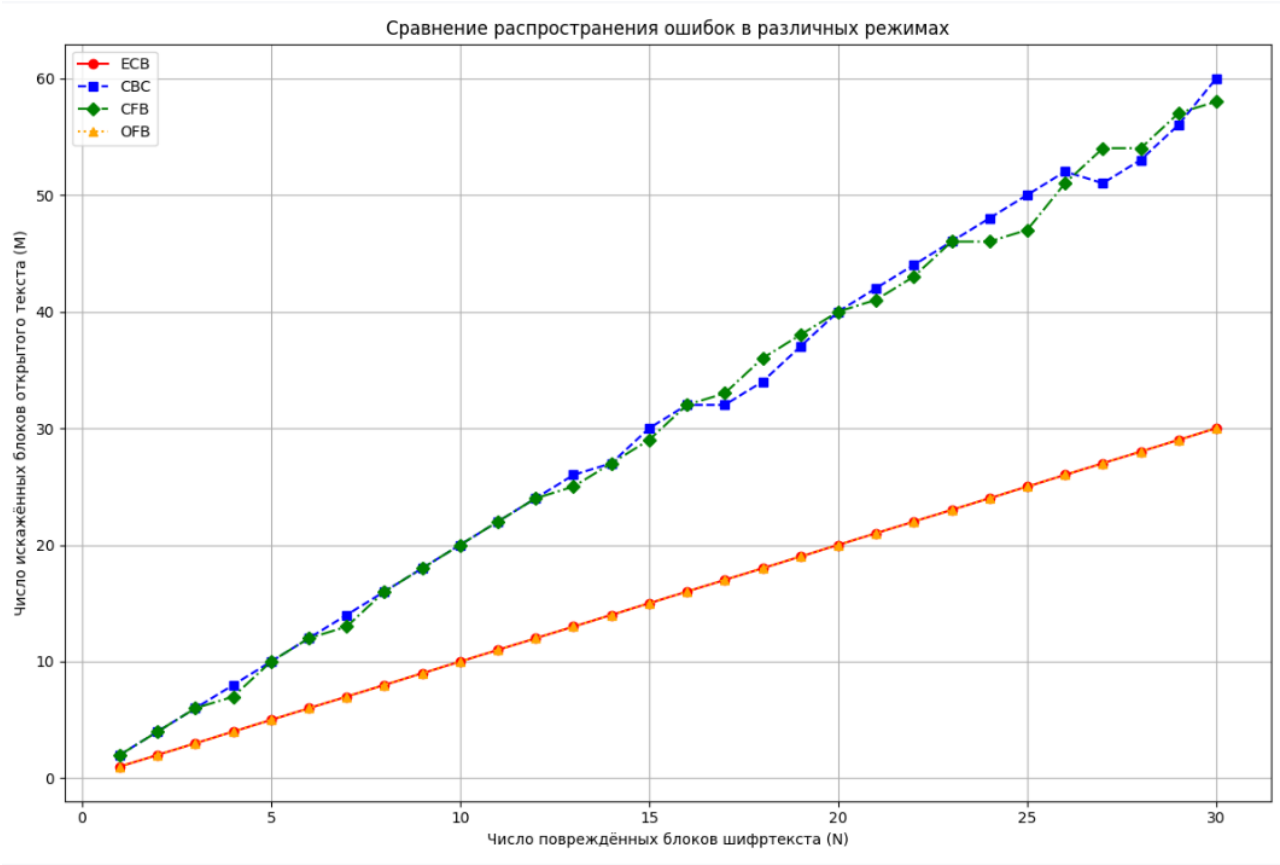

Рисунок 7 – Зависимость количества искажённых блоков открытого текста от количества повреждённых блоков шифртекста для различных режимов Camellia

На графике видно, что режим ECB демонстрирует строго линейную зависимость между количеством повреждённых блоков шифртекста и количеством искажённых блоков открытого текста. Это объясняется тем, что каждый блок шифруется независимо, без взаимного влияния. Повреждение одного блока не влияет на другие, и в результате график представляет собой прямую линию, отражающую предсказуемое и локализованное распространение ошибок.

В противоположность этому, режим CBC показывает выраженный эффект распространения ошибок. Повреждение одного блока шифртекста приводит к

искажению текущего и следующего блока открытого текста. В результате график демонстрирует более крутой рост значения M по сравнению с N.

Режим CFB ведёт себя схожим образом, однако степень распространения ошибки несколько меньше, чем у CBC. Ошибка может влиять на несколько следующих байт, но, как правило, не затрагивает целиком следующий блок.

Наконец, режим OFB показывает наибольшую устойчивость к ошибкам. Так как он основан на независимом потоке ключей, ошибка в шифртексте не влияет на генерацию следующих блоков. Повреждения остаются локальными и не распространяются, а количество искажённых блоков открытого текста растёт практически линейно, близко к линии y = x.

2.5 Вывод

Проведённый эксперимент подтвердил теоретические представления о поведении различных режимов блочного шифра Camellia при наличии ошибок в шифртексте. Режимы ECB и OFB проявили высокую устойчивость: в первом случае ошибка ограничивается только соответствующим блоком, а во втором — отдельным байтом, без влияния на последующие участки текста. Режим CBC продемонстрировал наиболее сильное распространение ошибок, когда искажение одного блока шифртекста влечёт за собой повреждение нескольких блоков открытого текста. Режим CFB занял промежуточное положение: его поведение также указывает на возможность распространения ошибок, но в меньшей степени по сравнению с CBC. Таким образом, выбор режима шифрования должен учитывать не только криптостойкость, но и устойчивость к ошибкам при передаче данных по ненадёжным каналам связи.

ЗАКЛЮЧЕНИЕ

В ходе прохождения экспериментально-исследовательской практики была проведена комплексная работа, направленная на изучение влияния искажений в шифротексте на корректность расшифровки при использовании различных режимов симметричного блочного шифра Camellia. В рамках практики были реализованы и протестированы четыре режима работы: ECB, CBC, CFB и OFB. Полученные результаты и построенные графики подтвердили теоретические предположения о поведении каждого режима при наличии ошибок и позволили провести сравнительный анализ их устойчивости.

В ходе выполнения заданий были приобретены навыки работы с криптографическими библиотеками на языке Python, разработана система моделирования ошибок, а также реализованы функции автоматического шифрования, внесения искажений, дешифрования и визуализации результатов в виде графиков.

За время прохождения практики освоены необходимые компетенции, в частности умения применять математические и программные методы для решения задач в области информационной безопасности, анализировать стойкость систем к ошибкам и разрабатывать экспериментальные решения с использованием современных криптографических подходов.

Полученные результаты могут быть использованы при проектировании защищённых систем передачи данных, с учётом требований к надёжности и устойчивости к возможным искажениям при передаче информации.

ПРИЛОЖЕНИЕ А

Код в файле analysis.py

```
import matplotlib.pyplot as plt
import os

COLOR_MAP = {
    "ECB": "red",
    "CBC": "blue",
    "CFB": "green",
    "OFB": "orange"
}

LINESTYLES = {
    "ECB": "-",
    "CBC": "--",
    "CFB": "-.",
    "OFB": ":"
}

MARKERS = {
    "ECB": "o",
    "CBC": "s",
    "CFB": "D",
    "OFB": "^"
}

def plot_damage(mode_name, damaged_blocks, N_values):
    os.makedirs("results", exist_ok=True)
    plt.figure(figsize=(10, 6))
    color = COLOR_MAP.get(mode_name, "black")
    linestyle = LINESTYLES.get(mode_name, "-")
    marker = MARKERS.get(mode_name, "o")
    plt.plot(N_values, damaged_blocks, marker=marker, linestyle=linestyle,
color=color)
    plt.xlabel("Число повреждённых блоков шифртекста (N)")
    plt.ylabel("Число искажённых блоков открытого текста (M)")
    plt.title(f"Распространение ошибок в режиме {mode_name}")
    plt.grid(True)
    plt.savefig(f"results/{mode_name}_damage.png")
    plt.clf()

def plot_all_modes(damage_dict):
    os.makedirs("results", exist_ok=True)
    plt.figure(figsize=(12, 8))
    for mode_name, damaged_blocks in damage_dict.items():
        x_values = list(range(1, len(damaged_blocks) + 1))
        color = COLOR_MAP.get(mode_name, "black")
        linestyle = LINESTYLES.get(mode_name, "-")
        marker = MARKERS.get(mode_name, "o")
        plt.plot(x_values, damaged_blocks, marker=marker, linestyle=linestyle,
                 label=mode_name, color=color)

    plt.xlabel("Число повреждённых блоков шифртекста (N)")
    plt.ylabel("Число искажённых блоков открытого текста (M)")
    plt.title("Сравнение распространения ошибок в различных режимах")
    plt.legend()
    plt.grid(True)
    plt.tight_layout()
    plt.savefig("results/all_modes_damage.png")
    plt.clf()
```

ПРИЛОЖЕНИЕ Б

Код в файле camellia_modes.py

```
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes
from cryptography.hazmat.backends import default_backend

BLOCK_SIZE = 16

def pad(data: bytes) -> bytes:
    padding_len = BLOCK_SIZE - len(data) % BLOCK_SIZE
    return data + bytes([padding_len]) * padding_len

def unpad(data: bytes) -> bytes:
    padding_len = data[-1]
    return data[:-padding_len]

def encrypt_camellia(mode_name, key, iv, plaintext):
    algo = algorithms.Camellia(key)

    if mode_name == "ECB":
        mode = modes.ECB()
        padded_data = pad(plaintext)
    elif mode_name == "CBC":
        mode = modes.CBC(iv)
        padded_data = pad(plaintext)
    elif mode_name == "CFB":
        mode = modes.CFB(iv)
        padded_data = plaintext
    elif mode_name == "OFB":
        mode = modes.OFB(iv)
        padded_data = plaintext
    else:
        raise ValueError(f"Unsupported mode: {mode_name}")

    cipher = Cipher(algo, mode, backend=default_backend())
    encryptor = cipher.encryptor()
    return encryptor.update(padded_data) + encryptor.finalize()

def decrypt_camellia(mode_name, key, iv, ciphertext):
    algo = algorithms.Camellia(key)

    if mode_name == "ECB":
        mode = modes.ECB()
    elif mode_name == "CBC":
        mode = modes.CBC(iv)
    elif mode_name == "CFB":
        mode = modes.CFB(iv)
    elif mode_name == "OFB":
        mode = modes.OFB(iv)
    else:
        raise ValueError(f"Unsupported mode: {mode_name}")

    cipher = Cipher(algo, mode, backend=default_backend())
    decryptor = cipher.decryptor()
    decrypted_data = decryptor.update(ciphertext) + decryptor.finalize()

    if mode_name in ["ECB", "CBC"]:
        return unpad(decrypted_data)
    else:
        return decrypted_data
```

ПРИЛОЖЕНИЕ В

Код в файле error_simulation.py

```
import random

def corrupt_ciphertext(ciphertext: bytes, block_size: int, mode_name: str,
                       corrupted_blocks: list[int], byte_indices: list[int]) ->
bytes:
    corrupted = bytearray(ciphertext)

    for block_idx, byte_in_block in zip(corrupted_blocks, byte_indices):
        start = block_idx * block_size

        if mode_name in ["ECB", "CBC"]:
            for i in range(block_size):
                corrupted[start + i] ^= 0xFF
        else:
            corrupted[start + byte_in_block] ^= 0xFF

    return bytes(corrupted)
```

ПРИЛОЖЕНИЕ Г

Код в файле main.py

```
import os
import random
import warnings
from camellia_modes import encrypt_camellia, decrypt_camellia
from error_simulation import corrupt_ciphertext
from analysis import plot_damage, plot_all_modes

warnings.filterwarnings("ignore", category=UserWarning)

BLOCK_SIZE = 16
MODES = ["ECB", "CBC", "CFB", "OFB"]
key = os.urandom(16)
iv = os.urandom(16)

with open("data/plaintext.txt", "rb") as f:
    original = f.read()

damage_results = {}

for mode in MODES:
    damaged = []

    iv_to_use = iv if mode != "ECB" else None
    ciphertext = encrypt_camellia(mode, key, iv_to_use, original)

    with open(f"data/{mode}_ciphertext.bin", "wb") as cipher_file:
        cipher_file.write(ciphertext)

    total_blocks = len(ciphertext) // BLOCK_SIZE
    max_n = min(total_blocks, 30)

    for n in range(1, max_n + 1):
        corrupted_blocks = random.sample(range(total_blocks), n)
        byte_indices = [random.randint(0, BLOCK_SIZE - 1) for _ in
corrupted_blocks]

        corrupted = corrupt_ciphertext(ciphertext, BLOCK_SIZE, mode,
corrupted_blocks, byte_indices)

        try:
            decrypted = decrypt_camellia(mode, key, iv_to_use, corrupted)
        except Exception:
            decrypted = b""

        m = 0
        min_len = min(len(original), len(decrypted))
        for i in range(0, min_len, BLOCK_SIZE):
            block_orig = original[i:i + BLOCK_SIZE]
            block_decrypt = decrypted[i:i + BLOCK_SIZE]
            if block_orig != block_decrypt:
                m += 1

        damaged.append(m)

    damage_results[mode] = damaged
    plot_damage(mode, damaged, list(range(1, len(damaged) + 1)))

plot_all_modes(damage_results)
```

ОТЗЫВ
О ПРОХОЖДЕНИИ ПРАКТИКИ

Вид практики учебная практика
(учебная, производственная, преддипломная)

Тип практики Экспериментально-исследовательская практика
(учебная, производственная, преддипломная)

Сроки прохождения практики: с 20.06.2025 г. по 03.07.2025 г.
по направлению подготовки 10.05.03 Информационная безопасность автоматизированных систем (уровень академического специалитета)
направленность (профиль) «Безопасность открытых информационных систем»
студентом группы № 6313-100503D Чан Куанг Минь

№ п/п	Критерии оценки	Оценка (по 5-балльной шкале)
1.	Общая систематичность и ответственность работы в ходе практики	
2.	Степень личного участия и самостоятельности практиканта в представляемой работе	
3.	Выполнение поставленных целей и задач	
4.	Корректность в сборе, анализе и интерпретации представляемых данных	
5.	Качество оформления отчетной документации	
ИТОГОВАЯ ОЦЕНКА*		

Руководитель практики
от профильной организации _____ А.И. Максимов
(подпись)

* Итоговая оценка выставляется как средняя арифметическая оценок по пяти критериям оценки