МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ

На правах рукописи

Алексеев Максим Олегович

МЕТОДЫ НЕЛИНЕЙНОГО КОДИРОВАНИЯ ДЛЯ ПОВЫШЕНИЯ ДОСТОВЕРНОСТИ ОБРАБОТКИ ИНФОРМАЦИИ

Специальность 05.13.01 – Системный анализ, управление и обработка информации (технические системы)

Диссертация на соискание ученой степени кандидата технических наук

Научный руководитель
доктор технических наук,
профессор Е.Т. Мирончиков

Санкт-Петербург – 2015

Содержание

Введение **5**

1 Модель канала со случайной структурой **11**
1.1 Модель алгебраических манипуляций 11
1.2 Некоторые практические приложения модели канала с алгебраическими манипуляциями 16
1.2.1 Воздействие ионизирующего космического излучения 17
1.2.2 Линейные схемы разделения секрета 20
1.2.3 Привнесение помех в вычислительные устройства 22
1.3 Выводы 28

2 Обзор основных методов повышения помехоустойчивости **29**
2.1 Методы защиты 29
2.1.1 Дублирование оборудования 31
2.1.2 Линейное помехоустойчивое кодирование 33
2.1.3 Хеширование 35
2.1.4 Нелинейное помехоустойчивое кодирование 36
2.2 Выводы 46

3 Границы на параметры нелинейных кодов **48**
3.1 Граница длины систематического R-равномерно надёжного кода 48
3.2 Нижняя граница обнаруживающей способности AMD кода на базе кодов Рида–Маллера 50
3.3 Выводы 53

4 Новые нелинейные кодовые методы повышения помехоустойчивости **54**
4.1 Обобщение надёжных кодов 54
4.1.1 Конструкция обобщённых систематических надёжных кодов 54
4.1.2 Исправление ошибок малой кратности 59
4.1.3 Исправление повторяющихся ошибок 69
4.1.4 Гибридный кодек, обнаруживающий алгебраические манипуляции 71
4.1.5 Сравнение с основными существующими конструкциями 73
4.1.6 Заключение по кодовой конструкции 75

4.2 Надёжный код на основе экспоненциальной почти совершенной нелинейной функции . . . 76
4.2.1 Экспоненциальная нелинейная функция . . . 76
4.2.2 Конструкция кода . . . 77
4.2.3 Применимость кодовой конструкции . . . 77
4.2.4 Заключение по кодовой конструкции . . . 78
4.3 Модификации AMD кода на основе операции умножения информационного и случайного компонентов . . . 79
4.3.1 Код на основе операции умножения информационного и случайного компонентов . . . 79
4.3.2 Модификация на основе расширения случайной величины . . . 81
4.3.3 Модификация на основе разбиения информационного сообщения . . . 82
4.3.4 Заключение по модификациям . . . 83
4.4 Код на основе операции скалярного умножения компонентов информационного сообщения и значения случайной величины . . . 85
4.4.1 Конструкция . . . 85
4.4.2 Сравнение с основными существующими конструкциями . . . 87
4.4.3 Заключение по кодовой конструкции . . . 88
4.5 Выводы . . . 88

5 Научно–технические предложения по применению нелинейных кодовых методов . 91
5.1 Области применения исследуемых нелинейных кодов . . . 91
5.2 Предложения по применению разработанных методов . . . 91
5.2.1 Повышение достоверности данных в космических аппаратах . . . 92
5.2.2 Защита архитектуры шифра AES от вычислительных ошибок . . . 93
5.3 Выводы . . . 98

Заключение . . . 100

Список литературы . . . 102

Список использованных сокращений . . . 115

Приложение А Сторонняя информация и атаки на её основе . . . 116
А.1 Введение . . . 116
А.2 Описание устройства смарт-карт как жертв атак по сторонним каналам . . . 119
А.3 Классификация сторонних каналов . . . 120
А.3.1 Контроль над вычислительным процессом . . . 120
А.3.2 Метод доступа к модулю . . . 120
А.3.3 Метод анализа . . . 122
А.4 Основные типы атак по сторонним каналам . . . 122

A.4.1 Атака зондированием 122
A.4.2 Атака по времени 124
A.4.3 Атака по энергопотреблению 127
A.4.4 Атака по электромагнитному излучению 133
A.4.5 Атака по привнесённым помехам 135
A.4.6 Атака по видимому излучению 145
A.4.7 Акустическая атака 146
A.4.8 Атака по кэш 146
A.4.9 Атака по частоте 148
A.4.10 Атака по сканированию 149
A.5 Вывод 149

Введение

Актуальность темы. Исследование и разработка методов помехоустойчивого кодирования, обеспечивающих достоверность и целостность информации в автоматизированных системах обработки информации и управления, в системах и сетях передачи данных на протяжении многих лет остается актуальной задачей. Это объясняется, с одной стороны, постоянно возрастающими объемами информации, обрабатываемой и хранимой в таких системах, а с другой — значительно возросшими требованиями к достоверности информации, передаваемой по каналам с шумами.

Принципиальная возможность обеспечения сколь угодно высокой достоверности передачи (хранения) информации для широкого класса каналов связи с отличной от нуля пропускной способностью была доказана К. Шенноном в его фундаментальной работе «Теория информации. Математическая теория связи», опубликованной в 1948 году. Поиск конструктивного доказательства теоремы К. Шеннона стимулировал многочисленные исследования, которые сформировали современную теорию помехоустойчивого кодирования. Основной задачей этой теории является построение кодов с характеристиками, обещанными теоремой К. Шеннона, которые имели бы конструктивные методы кодирования и декодирования.

Эффективность использования помехоустойчивого кода для обнаружения/исправления ошибок в канале передачи (хранения) информации зависит от того, насколько корректирующие свойства кода согласованы с законом распределения шума в канале. В алгебраической теории помехоустойчивого кодирования, занимающейся методами блокового кодирования, задачу согласования шума канала с корректирующей способностью кода принято описывать в терминах метрики пространства ошибок, согласованной с кодом.

Если удаётся найти метрику, согласованную с шумами канала, то задача построения наилучшего блокового кода сводится к поиску некоторого подмножества пространства, обладающего заданными метрическими свойствами. Это подмножество и является помехоустойчивым кодом, способным обнаруживать/корректировать наиболее вероятные ошибки канала.

Известно, что подавляющее большинство результатов теории помехоустойчивого кодирования относится к кодам, обнаруживающим/исправляющим ошибки в каналах, согласованных с

метрикой Хэмминга, то есть каналах, описываемых моделью q–ичного симметричного канала без памяти или, другими словами, дискретным отображением канала с белым гауссовским шумом. Однако, как показывают многочисленные экспериментальные исследования, реальные каналы передачи и хранения информации таковыми каналами не являются. Л.М. Финк и В.И. Коржик назвали эти каналы каналами со случайной структурой. Попытки найти метрику, согласованную с шумами для сколь–нибудь широкого класса таких каналов, не привели к успеху.

Существуют два подхода к решению вопроса обеспечения помехоустойчивости и целостности информации при передаче (хранении) в таких каналах. Первый связан с поиском специальных преобразований на входе и выходе канала, позволяющих обеспечить согласование преобразованного шума канала с корректирующими свойствами кода, исправляющего ошибки в метрике Хэмминга. Простейшим примером такого преобразования может служить декорреляция. Более сложные преобразования были предложены в работах Е. Элайеса, Д. Форни, Е.Т. Мирончикова, Г.Ш. Полтырева, Н.А. Шехуновой, Л.М. Финка и В.И. Коржика.

Другой подход предполагает построение специальных методов преобразования данных — кодов, позволяющих обнаруживать (исправлять) любые комбинации ошибок с некоторой отличной от нуля вероятностью. Первым шагом к построению таких методов, по–видимому, следует считать нелинейный код Васильева. Коды, предложенные в разное время в работах Фелпса, Моллера, Карповского, а также учеников Васильева Соловьевой, Августиновича и др., значительно развили этот подход.

Следует заметить, что авторы этих работ в качестве математической модели канала часто используют канал с алгебраическими манипуляциями. Как правило, модель алгебраической манипуляции понимается как изменение содержимого некоторого абстрактного запоминающего устройства на заданную величину при условии отсутствия зависимости между внутренним состоянием устройства и привносимой ошибкой. В модели рассматривается аддитивная ошибка, значение которой может принимать любое ненулевое значение из некоторого конечного алфавита. Таким образом, каналом с алгебраическими манипуляциями является канал, для которого характерно возникновение произвольных ошибок, не зависящих от передаваемых/хранимых данных.

Диссертация посвящена разработке и исследованию кодовых методов, позволяющих организовать достоверную передачу, хранение и обработку данных в канале с алгебраическими манипуляциями, обеспечивая их целостность. Под целостностью данных часто понимают то условие, что данные полны и не были изменены при их обработке (передаче, хранении, представлении). Получаемые коды гарантируют заданный уровень обнаружения любых искажений вне зависимости от их природы и источника. Таким образом, основным содержанием диссертации являются теоретические и прикладные исследования методов обработки информации, позволяющие повысить надёжность и качество функционирования систем передачи, хранения и обработки информации. К этим методам относятся методы построения классов нелинейных кодов, обнаруживающих ошибки, которые не были исследованы прежде в теории помехоустойчивого кодирования.

Цель работы состоит в разработке и исследовании новых кодовых методов, повышающих достоверность обработки информации, подвергающейся алгебраическим манипуляциям, и алгоритмов их декодирования, обладающих меньшей вычислительной сложностью по сравнению с аналогами. Использование этих кодов в системах передачи, хранения и обработки информации позволит повысить надежность обработки и помехозащищённость информации в каналах со случайной структурой, которые описываются моделью канала с алгебраическими манипуляциями. Таким образом, решается актуальная проблема разработки современных методов и алгоритмов решения задач обработки информации.

В соответствии с целью работы были поставлены следующие **задачи диссертационного исследования**:

1. Разработка и исследование кодового метода повышения помехоустойчивости на основе класса обобщённых систематических надёжных кодов, обнаруживающих алгебраические манипуляции.

2. Разработка алгоритма обнаружения и исправления ошибок малой кратности с помощью обобщённых систематических надёжных кодов.

3. Разработка и исследование кодового метода повышения помехоустойчивости, основанного на операции скалярного умножения компонентов информационного сообщения и значения случайной величины.

4. Модификация известного кодового метода повышения помехоустойчивости, основанного на операции умножения информационного и случайного компонентов, с целью уменьшения информационной избыточности.

5. Теоретико–информационный анализ нелинейных кодов, позволяющий вывести оценки (построить границы) экстремальных значений их параметров.

Методы исследования. Для решения поставленных задач использовались методы теории информации, теории помехоустойчивого кодирования, теории вероятностей и элементы теории нелинейных функций над конечными полями.

Степень достоверности результатов определяется корректным использованием методов исследования и математического аппарата, а также положительными итогами практического использования результатов диссертационной работы в ЗАО «Научные приборы» и в Государственном университете аэрокосмического приборостроения. Результаты находятся в соответствии с результатами, полученными другими авторами.

Научной новизной обладают следующие результаты работы:

1. Кодовый метод повышения помехоустойчивости, основанный на классе обобщённых систематических надёжных кодов. При определённых параметрах данные коды позволяют обнаруживать сильные манипуляции с большей вероятностью, чем существующие коды.

Кроме того, преимуществом данных кодов является существование простых алгоритмов исправления ошибок и возможности использования схемы гибридного кодека.

2. Алгоритм обнаружения и исправления ошибок малой кратности для обобщённых систематических надёжных кодов. Данный алгоритм обладает меньшими информационной избыточностью и вычислительной сложностью, чем существующие альтернативы.

3. Кодовый метод повышения помехоустойчивости, основанный на операции скалярного умножения компонентов информационного сообщения и значения случайной величины. Получаемый код отличается от других кодов, основанных на операции умножения в конечном поле, возможностью варьировать размер проверочной части кодового слова. Отличие от существующих кодов, обнаруживающих алгебраические манипуляции, заключается в большей вероятности обнаружения помех и меньшей вычислительной сложности.

4. Нижняя граница вероятности необнаружения алгебраической манипуляции, полученная для кодов, основанных на обобщённых кодах Рида—Маллера. Данная граница является уточнением существующей границы для кодов, обнаруживающих алгебраические манипуляции, применительно к конкретному методу построения кода.

5. Нижняя граница длины систематического равномерно надёжного кода, являющаяся улучшением нижней границы для надёжных кодов в систематическом представлении.

Практическая значимость заключается в том, что в ней разработаны математические методы обеспечения целостности и повышения помехозащищённости данных в каналах со случайной структурой, которые могут быть описаны моделью алгебраических манипуляций. Данные методы позволяют не только указать параметры кодов и построить алгоритмы их декодирования, но и дают возможность количественно оценить качество их использования на практике при обработке, передаче и хранении информации в каналах с алгебраическими манипуляциями.

Результаты диссертации используются для повышения надёжности при хранении информации в твердотельной памяти бортовых накопителей и при обработке её в вычислительных устройствах, устойчивых к привнесённым помехам. Кроме того, коды, получаемые с помощью предлагаемых методов построения, могут быть применены в смежных областях, включающих надёжные схемы разделения секрета, нечёткие экстракторы и другие.

Внедрение и реализация результатов работы. Основные результаты работы были использованы при выполнении научно–исследовательской работы по разработке и исследованию надёжных методов хранения информации в аэрокосмических системах и комплексах, выполняемой по заданию № 2.2716.2014/К Минобрнауки России. Кодовая конструкция, основанная на операции скалярного умножения компонентов информационного сообщения и значения случайной величины, также была использована в ЗАО «Научные приборы» при проектировании защищённых электронных идентификационных документов в рамках выполнения научно–исследовательской работы. Кроме того, теоретические результаты работы используются в учебном процессе кафед-

ры аэрокосмических компьютерных и программных систем Санкт-Петербургского государственного университета аэрокосмического приборостроения.

Апробация работы. Основные результаты работы докладывались и обсуждались на следующих семинарах, конференциях и симпозиумах:

Научных сессиях ГУАП (Санкт–Петербург, Россия, 2011-2013); семинаре лаборатории «Reliable Computing Laboratory» Бостонского университета (Бостон, США, 2011); Всероссийской научной конференции по проблемам информатики «СПИСОК» (Санкт–Петербург, Россия, 2012); 23–ей научно–технической конференции «Методы и технические средства обеспечения безопасности информации» (Санкт–Петербург, Россия, 2014); семинаре «Информатика и компьютерные технологии» СПИИРАН (Санкт–Петербург, Россия, 2014), симпозиуме «Workshop on Coding and Cryptography» (Париж, Франция, 2015).

Публикации. Результаты, представленные в диссертационной работе, опубликованы в 12 печатных работах [1–12]. Среди них 5 работ [1–5] опубликованы в изданиях, включённых в перечень ВАК.

Основные положения, выносимые на защиту:

1. кодовый метод повышения помехоустойчивости на основе класса обобщённых систематических надёжных кодов, обнаруживающих алгебраические манипуляции;

2. алгоритм обнаружения и исправления ошибок малой кратности с помощью обобщённых систематических надёжных кодов;

3. кодовый метод повышения помехоустойчивости, основанный на операции скалярного умножения компонентов информационного сообщения и значения случайной величины, а также его модификации;

4. границы экстремальных значений параметров нелинейных кодов, обнаруживающих алгебраические манипуляции.

Объем и структура работы. Диссертационная работа состоит из введения, пяти глав, заключения, списка использованных источников (142 наименования), а также списка использованных сокращений. Диссертация содержит 115 страниц, включая 6 таблиц и 17 рисунков. Приложение содержит 35 страниц, включая 14 рисунков и 1 таблицу.

В первой главе работы рассматривается модель канала с алгебраическими манипуляциями. Приводятся примеры каналов со случайной структурой, которые могут быть описаны моделью канала с алгебраическими манипуляциями. Более подробно расписываются три практических приложения модели: воздействие космического излучения, линейные схемы разделения секрета с мошенниками и привнесение помех в вычислительные устройства.

Во второй главе рассматриваются основные методы защиты от помех, включающие дублирование, использование помехоустойчивых кодов и хеширование. Приводятся схемы параллельного обнаружения ошибок с использованием данных методов. Анализируется эффективность данных методов при искажениях данных, описываемых алгебраическими манипуляциями.

Третья глава посвящена теоретико–информационному анализу нелинейных кодов, который позволяет построить границы их параметров. Первая граница определяет теоретическую минимальную длину систематического R–равномерно надёжного кода. Вторая граница является нижней границей на обнаруживающую способность кода, обнаруживающего алгебраические манипуляции, основанного на обобщённых кодах Рида—Маллера.

В четвёртой главе приведены новые кодовые методы повышения помехоустойчивости на основе нелинейных кодов, разработанные автором в процессе выполнения диссертационной работы. Кроме того, представлены две модификации существующего кода на основе операции умножения, позволяющие уменьшить информационную избыточность. Для обобщённых систематических надёжных кодов разработан метод исправления ошибок малой кратности, а также описывается принцип гибридного кодека на их основе.

Пятая глава содержит научно–технические предложения по применению нелинейных кодов. Первое применение — повышение надёжности хранения данных в твердотельных накопителях космических аппаратов. Второе — проектированию защищённой аппаратной архитектуры шифра AES, устойчивой к привнесённым помехам. Помимо этого, перечисляются основные направления использования исследуемых классов нелинейных кодов.

В заключении кратко перечислены основные результаты, полученные в ходе выполнения диссертационной работы.

1 Модель канала со случайной структурой

1.1 Модель алгебраических манипуляций

Одной из важнейших характеристик автоматизированных систем обработки информации и управления является их помехозащищённость. Это обусловлено постоянно возрастающими объёмами обрабатываемой и передаваемой информации (и, соответственно, пропорционально растущими скоростями обработки и передачи), а также ростом требований к её достоверности. В классическом труде А.А. Харкевича «Борьба с помехами» помехоустойчивость определяется как способность системы противостоять вредному влиянию помех [13]. Помехой называется стороннее возмущение, действующее в технической системе и препятствующее правильному приёму (хранению, обработке) сигналов [13]. Источники помех могут быть как внутри системы обработки информации, так и вне её. Борьба с регулярными помехами, величины которых постоянны и известны, не представляет особых трудностей и осуществляется, например, компенсацией или использованием соответствующих фильтров [13]. В данной работе будут рассматриваться помехи другого рода — случайные, не поддающиеся предсказанию. Фактически, помехой будем считать реализации некоторого дискретного случайного процесса. Очевидно, что борьба с такого рода помехами является значительно более трудной задачей.

Причины возникновения случайных помех бывают самые различные (тепловой шум, взаимная интерференция, атмосферные помехи, космические и так далее) и глубоко заложены в природе вещей. К примеру, флуктуационные помехи (тепловой шум, дробовый эффект) есть результат дискретного строения вещества и статистической природы ряда физических величин [13]. Действительно, многие физические величины представляют собой результат усреднения по большому количеству частиц, поведение и вклад которых случайны. Следовательно, флуктуации этих физических величин принципиально неустранимы, остаётся лишь пытаться оценить вклад флуктуаций в исследуемый процесс и стремиться его компенсировать.

Стоит отметить, что надёжность функционирования системы и качество обработки информации определяются, в том числе, помехоустойчивостью всех компонентов системы. Например, рассматривая систему связи, необходимо говорить о помехоустойчивости выбранной схемы модуляции, используемого помехоустойчивого кода, метода синхронизации. Говоря о надёжности функционирования некоторого вычислительного устройства, необходимо рассматривать помехозащищённость таких его компонентов, как арифметико–логическое устройство, регистры, ячейки памяти, интерфейсы ввода/вывода и так далее.

Для повышения помехоустойчивости системы используются различные методы [14–16]:

– увеличение мощности сигналов;

– применение физических методов защиты каналов обработки (передачи, хранения) информации;

– выбор оптимальных методов обработки (передачи, хранения) информации;

– использование помехоустойчивых приёмников.

Одним из наиболее эффективных методов повышения достоверности информации является использование кодовых методов, основанных на привнесении в обрабатываемые данные специальным образом выбранной (вычисленной) информационной избыточности [14]. Суть метода заключается в такой обработке информации, при которой передаваемые сообщения преобразовываются в некоторые разрешённые последовательности — кодовые слова — из заранее подготовленного множества, называемого кодом. Каждое кодовое слово обладает информационной избыточностью — проверочными символами, которые не несут информации, но используются для обнаружения и исправления ошибок.

Данная диссертационная работа посвящена исследованию помехозащищённости нелинейных кодовых методов, а также разработке способов её повышения за счёт использования новых кодов и алгоритмов декодирования. Повышение помехозащищённости технических систем приводит к повышению их надёжности и качества функционирования.

Опубликованные в фундаментальной работе Клода Шеннона «Теория информации. Математическая теория связи» теоремы о кодировании для каналов с шумом, казалось бы, решили вопрос с достоверностью передаваемых и хранимых данных [17]. Выбрав канал, чья пропускная способность превышает требуемую скорость передачи данных, использование достаточно длинного случайно выбранного кода обеспечивает сколь угодно близкую к нулю вероятность ошибочного декодирования сообщений.

Однако, последовавшие за статьёй исследования в этой области показали, что на практике осуществление данного подхода к повышению верности декодирования является трудновыполнимой задачей [15]. Во–первых, использование случайного кодирования с декодированием, основанным на переборе всех возможных разрешённых комбинаций, для кодов большой длины является технически невыполнимым, в то время как использование кодов малой длины оказывается недостаточно эффективным. Во–вторых, пришедшие на замену случайному кодированию алгебраические коды строились и исследовались применительно к идеализированному симметричному каналу с независимыми ошибками [14]. Применение их на практике зачастую оканчивалось неудачей, что было вызвано тем, что реальные дискретные каналы передачи и хранения данных далеки от этой идеализированной модели. Вслед за первыми неудачами по использованию помехоустойчивого кодирования на практике последовали работы, посвященные экспериментальным исследованиям существующих дискретных каналов и описанию их с помощью

математических моделей [15]. Эти модели позволяли определять вероятностные характеристики потоков ошибок, по которым можно было оценивать эффективность применения различных помехоустойчивых кодов.

Эффективное использование кодовых методов для обнаружения/исправления ошибок в канале передачи (хранения) информации возможно лишь при согласованности свойств кода с законом распределения шума в канале [14]. Для описания задачи согласования шума канала с обнаруживающей/корректирующей способностью кода в алгебраической теории помехоустойчивого кодирования, занимающейся построением и изучением методов блокового кодирования, принято использовать термины метрики пространства ошибок, согласованной с кодом.

В случае, если удаётся найти метрику, согласованную с распределением шума в канале, то задача построения блокового кода сводится к выбору некоторого подмножества пространства, обладающего заданными дистанционными свойствами в данной метрике. Выбранное подмножество и является помехоустойчивым кодом, способным обнаруживать/исправлять наиболее вероятные ошибки канала.

Как было отмечено выше, большая часть результатов теории помехоустойчивого кодирования относится к кодам, исправляющим ошибки в каналах, согласованных с метрикой Хэмминга, то есть каналах, описываемых моделью q–ичного симметричного канала без памяти или, другими словами, дискретным отображением канала с белым гауссовским шумом [14, 16]. Однако, многочисленные экспериментальные исследования показывают, что большинство реальных каналов передачи и хранения информации таковыми каналами не являются.

Характеристики большой части реальных каналов зависят от изменяющихся во времени случайных процессов (например, замирания). Изменение, происходящие с амплитудой и фазой сигнала при прохождении его через непрерывный канал, которое затрудняет приём этого сигнала, называется мультипликативной помехой. На практике, мультипликативная помеха возникает всегда, когда параметры канала передачи (обработки, хранения) претерпевают изменения во времени [13]. В сущности, так обстоит дело во всех реальных технических системах, но иногда величина и вклад мультипликативной помехи настолько малы, что ими можно пренебречь. В других случаях, когда случайные изменения структуры канала передачи (обработки, хранения) в значительной мере перестраивают весь механизм передачи (обработки, хранения), передача может стать абсолютно невозможной. Л.М. Финк и В.И. Коржик назвали эти каналы каналами со случайной структурой [15]. В частности, в этот класс каналов ими были отнесены все радиоканалы связи протяжностью более нескольких километров [15]. Помимо них, в качестве каналов со случайной структурой можно привести следующие примеры «тяжёлых» каналов:

– каналы тропосферной связи;

– воздействие ионизирующего излучения на технические системы;

– каналы, относящиеся к модели произвольно изменяющихся каналов (arbitrary varying channel) [18];

– воздействие космического излучения на летательные аппараты;

– дефекты запоминающих устройств, вызванные факторами, зависящими от времени («старение» памяти);

– действия мошенников в схемах разделения секрета [19];

– привнесение помех в работу устройства за счёт нестандартных физических воздействий.

Попытки найти метрику, согласованную с шумами для сколь–нибудь широкого класса таких каналов, не привели к успеху.

Для обеспечения достоверности информации в каналах со случайной структурой используются два подхода:

– специальные преобразования канала;

– построение методов обнаружения произвольных искажений данных с заданной вероятностью.

Первый подход подразумевает преобразование канала передачи (хранения, обработки) с помощью специальных методов, целью которого является обеспечение возможности согласования преобразованного шума канала с корректирующими свойствами кода в метрике Хэмминга. Простейшими примерами таких преобразований являются следующие [13, 15, 16]:

– случайное внутриблочное перемешивание;

– внеблочное перемешивание;

– поразрядное суммирование;

– перемножение блоков;

– декодирование с предсказанием.

Второй подход связан с построением специальных кодовых методов повышения помехоустойчивости, позволяющих обнаруживать любые конфигурации ошибок с некоторой ненулевой вероятностью. Значительный вклад в этой области достигнут благодаря нелинейным методам кодирования: кодам Васильева, Соловьёвой, Фелпса, Карповского и других [20–26].

В частности, в работах Карповского при разработке и исследовании нелинейных кодовых методов используется модель дискретного канала с алгебраическими манипуляциями. Приведём описание модели этого канала [27]. Пусть имеется абстрактное запоминающее устройство, обозначаемое $\Sigma(G)$. В данное устройство поступает на хранение некоторый элемент g из конечной аддитивной абелевой группы G. В результате некоторых факторов, данные, хранящиеся в устройстве, подвергаются искажению на некоторую ненулевую величину $\delta \in G$. При этом отсутствует зависимость между внутренним состоянием устройства $\Sigma(G)$ и величиной привносимого искажения δ, однако возможна зависимость между входными данными g и искажением

δ. Если входные данные хранились в устройстве в исходном виде, то при чтении информации из $\Sigma(G)$ пользователь получит значение $g + \delta \in G$ и не сможет определить факт искажения данных (рисунок 1.1). Очевидно, что такая ситуация является нежелательной. Следовательно, необходимо некоторым образом преобразовывать данные, поступающие в $\Sigma(G)$, так, чтобы можно было обеспечить их целостность. Фактически, ставится задача помехоустойчивого кодирования внутреннего состояния некоторой технической системы. Таким образом, алгебраическая манипуляция, как правило, понимается как изменение содержимого некоторого абстрактного запоминающего устройства на заданную величину (ошибку) при условии отсутствия зависимости между внутренним состоянием устройства и величиной привносимого искажения. В качестве абстрактного устройства может выступать любое реальное устройство, осуществляющее обработку, хранение или передачу данных. Привносимая ошибка принимается аддитивной[1)] и может принимать любое ненулевое значение из заданной группы.

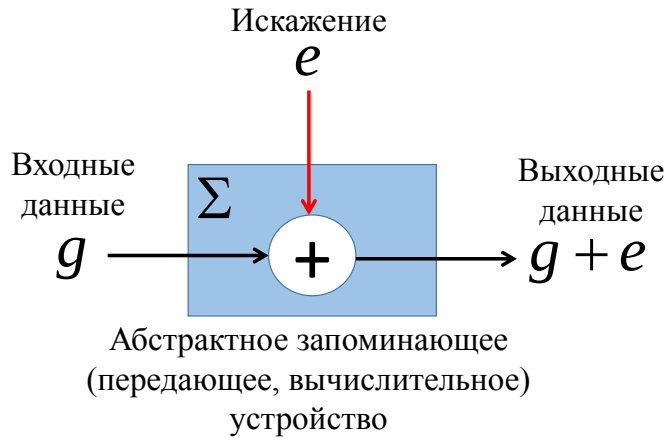


Рисунок 1.1 – Схема алгебраической манипуляции

В общем случае, моделью источника ошибок можно рассматривать случайный дискретный ансамбль, элементами которого являются все образцы ошибок, с заданным на нём законом распределения. Канал характеризуется одинаковыми входным и выходным алфавитами.

В. И. Коржик и Л. М. Финк доказали, что поразрядное суммирование с равномерно распределённой случайной величиной перед и после канала связи приводит к тому, что преобразованный канал становится каналом с аддитивной ошибкой, каким бы ни был исходный канал [15]. Данное преобразование канала может быть легко реализовано в технической системе, гарантируя защиту от полного спектра возможных искажений данных.

Алгебраическую манипуляцию будем считать успешной, если при обработке (передаче, хранении) данных произошедшая ошибка не была обнаружена. Следовательно, качество надёжности обработки и передачи информации определяется вероятностью необнаружения манипуляции или вероятностью необнаружения искажения данных, что в данном случае одно и тоже, которые

[1)]Выше было указано, что причиной случайной структуры канала обычно выступает мультипликативная помеха. В данном случае речь шла о непрерывном канале, после его дискретизации (переход к дискретному каналу) результат воздействия мультипликативной помехи может быть сведен к эквивалентной аддитивной помехе.

выступают в качестве количественной меры надёжности обработки информации в рассматриваемом канале.

Алгебраические манипуляции делятся на два вида: слабые и сильные. Слабая модель используется для описания ситуаций, когда отсутствует зависимость между привносимой ошибкой и данными, поступившими на вход рассматриваемого абстрактного устройства. Напомним, что внутри устройства входные данные могут подвергаться некоторому преобразованию (например, помехоустойчивому кодированию или шифрованию), поэтому в общем случае, хранимые данные (состояние устройства) отличаются от поступивших на вход устройства. Другими словами, слабая манипуляция описывает ситуацию, когда значение привносимой ошибки случайно, не зависит от входных данных (и внутреннего состояния) и зачастую распределено равномерно. Вероятность необнаружения слабых манипуляций будет обозначаться P^{w}_{undet}.

Модель сильных алгебраических манипуляций используется в тех случаях, когда между входными данными (но не внутренним состоянием) и привносимой ошибкой существует зависимость. Примером канала с сильными манипуляциями является асимметричный канал, в котором величина возникающей ошибки напрямую зависит от поступающих данных. При сильных манипуляциях может достигаться большая вероятность необнаруживаемого искажения информации; обозначим её P^{s}_{undet}. В общем случае, при сильной манипуляции достигается более высокая вероятность искажения данных, нежели при слабой. Эффективность разрабатываемых и исследуемых в работе методов нелинейного кодирования будет оцениваться границей сверху данной вероятности. Это позволит гарантировать достоверность обработки информации во всех каналах с сильными манипуляциями без учета зависимости между ошибками и данными. Если рассматриваемый метод защиты обеспечивает одинаковый уровень обнаружения слабых и сильных манипуляций, то для краткости вероятность их необнаружения будет обозначаться P_{undet}.

1.2 Некоторые практические приложения модели канала с алгебраическими манипуляциями

Относительно общая модель канала с алгебраическими манипуляциями может быть применена к широкому спектру реальных каналов со случайной структурой. Рассмотрим более подробно три практических приложения исследуемой модели канала, которые представляются наиболее актуальными в смысле применения разрабатываемых методов на практике. Их актуальность, в частности, обосновывается количеством публикаций, представленных в научно–технической литературе [19, 24, 25, 27–35].

К этим приложениям относятся:

– воздействие ионизирующего космического излучения (солнечная радиация, галактическое космическое излучение, радиационные пояса Земли и так далее) на аэрокосмическое оборудование;

– схемы разделения секрета, среди участников которых присутствуют мошенники;

– проектирование защищённых вычислительных устройств, устойчивых к привнесённым помехам.

1.2.1 Воздействие ионизирующего космического излучения

Ионизирующее излучение оказывает существенное влияние на свойства полупроводников [31]. Данное явление исследуется довольно давно, его результатам посвящено большое количество работ по физике, материаловедению и приборостроению (например, [34–37]).

Различают два типа ионизирующего излучения:

– коротковолновое электромагнитное излучение (рентгеновское излучение и гамма–излучение);

– потоки частиц (электроны, позитроны, нейтроны, альфа–частицы, тяжелые ионы, возникающие при делении ядер).

В качестве основных источников ионизирующего излучения можно привести следующие [31, 34]:

– галактическое космическое излучение;

– солнечная радиация;

– внешний радиационный пояс Земли (расстояние до Земли около 17000 км);

– внутренний радиационный пояс Земли (расстояние до Земли около 4000 км).

Галактическое космическое излучение обладает наиболее высокой энергией по сравнению с другими видами космической корпускулярной радиации. Энергия изучения заключена в чрезвычайно широком интервале — от 108 до 1021 эВ, но плотность потока его частиц сравнительно невелика [35].

Плазма солнечного ветра, состоящая в основном из протонов и электронов, движется в окрестностях Земли со скоростью 320 – 400 км/с [31]. Кинетическая энергия протонов при такой скорости составляет 600 – 800 эВ, а электронов — лишь 0,3 – 0,4 эВ, поскольку масса электрона почти в 2000 раз меньше массы протона. Основным воздействующим фактором солнечного ветра является поток протонов. Во время вспышек на Солнце скорость солнечного ветра может увеличиться до 1000 км/с, при этом соответственно возрастает энергия протонов и плотность их потока [34]. Воздействие протонов солнечного ветра на материалы сводится к следующим основным эффектам: распылению и созданию радиационных дефектов структуры в приповерхностном слое за счет внедрения протонов и «смещения» атомов вещества [34].

Во время вспышек Солнце испускает еще солнечные космические лучи в основном это потоки протонов с энергиями от 1 до 10^4 МэВ [34]. Энергия протонов солнечных космических во много раз превышает энергию протонов солнечного ветра.

Вокруг Земли есть 2 пояса заряженных частиц — так называемые радиационные пояса Ван Аллена: на высоте около 4000 км из протонов, и на высоте около 17 000 км из электронов [34,35]. Частицы там движутся по замкнутым орбитам, захваченные магнитным полем Земли. Также есть бразильская магнитная аномалия — где внутренний радиационный пояс ближе подходит к Земле — до высоты 200 км [34].

По мере роста энергии заряженных частиц, воздействующих на космические аппараты и бортовые вычислительные комплексы, в материалах и элементах аппаратуры усиливаются вызываемые ими радиационные эффекты: образование дефектов структуры, ионизация и возбуждение атомов вещества, ядерные превращения.

Все радиационные изменения, происходящие в материалах и элементах аппаратуры можно разделить на обратимые и необратимые. Обратимые происходят в материалах в процессе облучения и практически полностью исчезают после его прекращения. Необратимые изменения непрерывно накапливаются в процессе облучения и сохраняются полностью или частично после прекращения облучения. Изменения первого вида связаны в основном с ионизацией и возбуждением атомов вещества, изменения второго вида — с образованием радиационных дефектов [34–37].

Радиационная стойкость полупроводниковых приборов оценивается главным образом по необратимым эффектам, которые могут в конечном итоге привести к отказу или полному выходу прибора из строя. Отказом считается уход основного контролируемого параметра прибора из заданного интервала допустимых значений. Для характеристики радиационной стойкости часто используется понятие «предельно допустимый поток» — отнесенное к единице площади максимальное число попаданий частиц, которое материал или прибор выдерживает без изменения параметров. Например, при облучении кремниевых транзисторов протонами с энергией 20 МэВ коэффициент усиления этих транзисторов начинает уменьшаться после того, как суммарный поток протонов превысит величину 10^{11} протон/см^2 [34]. Заметим, что такой поток протонов высокой энергии могут получить за год приборы, находящиеся вне гермоотсека космического аппарата, при полете во внутреннем радиационном поясе. Более универсальным критерием для оценки радиационной стойкости является величина поглощенной дозы излучения, при которой начинаются заметные изменения параметров материалов или приборов. Поглощенная доза ионизирующего излучения измеряется энергией излучения, которая поглощается единицей массы вещества. Единица поглощенной дозы носит название Грей (Гр) и имеет размерность джоулей на килограмм. Иногда используется другая единица для измерения поглощенной дозы излучения рад = 0,01 Гр. Производимые интегральные схемы делятся на несколько классов по признаку радиационной устойчивости к накопленной дозе излучения (таблица 1.1) [31, 35].

Тяжелые заряженные частицы (протоны, альфа–частицы и ионы больших энергий) имеют такую высокую энергию (диапазон энергий электронов от 0,05 до 5 МэВ, для протонов – от 0,1 до 50 МэВ), что «пробивают» микросхему насквозь (вместе с корпусом спутника), и оставляют за собой «шлейф» заряда [34,35]. В лучшем случае это может привести к программной ошибке (0 станет 1 или наоборот — single–event upset), в худшем — привести к тиристорному защелкиванию

Таблица 1.1 – Классификация микросхем по признаку радиационной устойчивости [31]

Категория микросхем	Минимальное значение предельной дозы, крад	Максимальное значение предельной дозы, крад
Коммерческие (commercial)	2,0	12,0
Промышленные (industrial)	20,0	60,0
Военного применения (military)	100,0	2000,0

(single–event latchup). У защелкнутого чипа питание закорачивается на землю через паразитный тиристр, что может привести к выходу микросхемы из строя. Если питание успеть отключить и подключить до сгорания, то все будет работать как обычно.

Отмечается, что тяжёлые заряженные частицы космического пространства, воздействуя на интегральную микросхему, могут вызвать искажения отдельных битов данных или программы. Интенсивность сбоев зависит от типа используемой памяти, параметров орбиты и активности Солнца. Для статической памяти 537–ой серии объёмом 32 кбайт, используемой в системном контроллере рассматриваемой бортовой вычислительной системы, ожидаемая средняя интенсивность сбоев для высоких орбит может доходить в сутки до 0,0001 сбоя, а пиковая — до 1 сбоя [31]. Сбои динамической памяти объёмом 2–4 Мбайт могут достигать интенсивности 0,01 сбоя в сутки, а флэш–памяти — 0,0000001 сбоя в сутки [31]. Постоянные запоминающие устройства контроллеров, не использующие зарядовое хранение информации, подобным сбоям не подвержено.

Сверх большие интегральные схемы динамической памяти служат основными источниками сбоев контроллера на борту космических аппаратов из–за высокой вероятности инверсий логического состояния ячеек памяти при попадании в них высокоэнергетических частиц. В то же время вероятность сбоев процессора, контроллеров, постоянных запоминающих устройств и других интегральных схем из–за ионизационных эффектов значительно ниже (более чем на два порядка), а вероятность инверсий логического состояния ячеек флэш–памяти практически равна нулю [32]. Таким образом, можно сделать следующий вывод: основным источником ошибок в памяти космических аппаратов являются ошибки, которые возникают в статической или динамической оперативной памяти. Ошибки во флэш–памяти имеют значительно более низкую интенсивность, что говорит о достаточно высокой устойчивости флэш–памяти к радиационному излучению по сравнению с другими видами памяти [32].

Кроме того, для некоторых типов памяти характерно ухудшение их помехоустойчивых качеств при длительном использовании (без учёта космического излучения) [31]. Например, этот

эффект особенно явно проявляется у флеш–памяти. Флеш–память имеет довольно ограниченный ресурс использования, что вызвано значительным увеличением мощности шума в канале хранения данных под воздействием следующих факторов [38]:

– Ограниченный ресурс записи. По мере увеличения количества циклов записи/стирания накопленный остаточный заряд в плавающем затворе каждой ячейки памяти. Причем, в каждой ячейке он разный, и разница увеличивается с увеличением циклов записи/стирания. Поскольку запись/стирание осуществляется большими блоками, то скомпенсировать это отличие в накопленном заряде в каждой отдельной ячейке не представляется возможным. Данный фактор является основной причиной «старения» флеш–памяти.

– Ограниченное время хранения заряда на плавающем затворе. Со временем за счет микротоков утечки заряда затвора происходит изменение величины заряда. Типичное время хранения заряда ограничено обычно 10 годами.

– Интерференция и перетекание заряда между соседними ячейками.

– Искажения, вызванные чтением соседних ячеек памяти.

– Появление отказавших ячеек/блоков памяти. Неравномерное использование элементов памяти может приводить к их неравномерному «старению», в результате чего часть ячеек/блоков выходит из строя раньше остальных.

– Другие факторы.

Из представленных данных по воздействию космического излучения на летательные аппараты следует, что для повышения надёжности функционирования оборудования необходимо предусмотреть возможность обеспечения целостности данных при условии произвольных конфигураций ошибок. Более того, если рассматривать память бортовых вычислительных комплексов, то при длительном нахождении на орбите возможно ухудшение её помехоустойчивости и надёжности, вызванное интенсивным/длительным использованием. Таким образом, для описания воздействия космического излучения различных источников (солнечная радиация, галактическое космическое излучение, радиационные пояса Земли и так далее) и факторов «старения» на оборудование космических аппаратов также может быть использована модель алгебраических манипуляций.

1.2.2 Линейные схемы разделения секрета

В линейной схеме порогового разделения секрета (например, схема Шамира, Блума, Блэкли и другие) *секрет* s распределён между n участниками так, что каждый участник получает некоторую алгебраическую *долю* секрета [33, 39, 40]. Любое *подходящее* подмножество участников (например, не менее $k \leq n$ участников) может объединить свои доли и восстановить секрет s по средствам линейного преобразования, в то время как любое *неподходящее* множество (менее

k участников) не может получить какой-нибудь информации о секрете. К сожалению, корректность восстановленного секрета обеспечивается только тогда, когда все объединённые доли не были изменены. В частности, если хотя бы один из участников *подходящего* множества является мошенником, это может привести к восстановлению изменённого секрета $\hat{s}$. Более того, разность между корректным секретом s и восстановленным $\hat{s}$ контролируется участниками–мошенниками, что объясняется линейностью схемы. К счастью, возможности мошенников ограничены следующими фактами:

– благодаря конфиденциальности схемы разделения секрета, величина искажения, вносимого злоумышленниками, определяется только их априорным знанием о секрете;

– в силу линейности схемы, для соответствующей модификации своих долей мошенники должны знать разность между s и $\hat{s}$.

Рассмотрим алгоритм осуществления модификаций доли секрета мошенниками на примере из [41]. Используется схема Шамира, количество участников равно пяти, секрет s равен «11», а для восстановления секрета требуется три участника.

Выберем простое число $p = 13$. Далее, согласно алгоритму, необходимо построить полином второй степени, например:

$$F(x) = (7x^2 + 8x + 11) \bmod 13.$$

В данном свободный член, равный «11», и есть разделяемый секрет s, а коэффициенты при остальных членах — некоторые случайные числа, которые используются только при генерации долей. Далее вычисляем доли участников (точки, через которые проходит выбранный полином):

$$\begin{aligned}
k_1 &= F(1) = (7 \cdot 1^2 + 8 \cdot 1 + 11) \bmod 13 = 0,\\
k_2 &= F(2) = (7 \cdot 2^2 + 8 \cdot 2 + 11) \bmod 13 = 3,\\
k_3 &= F(3) = (7 \cdot 3^2 + 8 \cdot 3 + 11) \bmod 13 = 7,\\
k_4 &= F(4) = (7 \cdot 4^2 + 8 \cdot 4 + 11) \bmod 13 = 12,\\
k_5 &= F(5) = (7 \cdot 5^2 + 8 \cdot 5 + 11) \bmod 13 = 5.
\end{aligned}$$

После этого доли с их номерами, числом $p = 13$ и степенью многочлена делятся между участниками.

Любые три участника, объединившись, могут восстановить секрет s, например, с помощью интерполяционного полинома Лагранжа. Продемонстрируем этот процесс на примере объединения участников с долями номер два, три и пять ($k_2 = 3$, $k_3 = 7$ и $k_5 = 5$). Полином Лагранжа имеет следующий вид:

$$F(x) = \sum_i l_i(x) y_i \bmod p,$$

где базисные полиномы $l_i(x)$ определяются по формуле:

$$l_i(x) = \prod_{i \neq j} \frac{x - x_j}{x_i - x_j} \bmod p.$$

Для рассматриваемого примера получаем следующие базисные полиномы:

$$l_1(x) = \frac{x-x_2}{x_1-x_2}\cdot\frac{x-x_3}{x_1-x_3} = \frac{x-3}{2-3}\cdot\frac{x-5}{2-5} = 9x^2+6x+5 \bmod 13,$$
$$l_2(x) = \frac{x-x_1}{x_2-x_1}\cdot\frac{x-x_3}{x_2-x_3} = \frac{x-2}{3-2}\cdot\frac{x-5}{3-5} = 6x^2+10x+8 \bmod 13,$$
$$l_3(x) = \frac{x-x_1}{x_3-x_1}\cdot\frac{x-x_2}{x_3-x_2} = \frac{x-2}{5-2}\cdot\frac{x-3}{5-3} = 11x^2+10x+1 \bmod 13.$$

Исходный полином вычисляем следующим образом:

$$F(x) = 3\cdot l_1(x) + 7\cdot l_2(x) + 5\cdot l_3(x) \bmod p.$$

Откуда получаем следующее выражение для свободного члена, являющегося искомым секретом:

$$s = 3\cdot 5 + 7\cdot 8 + 5\cdot 1 = 11 \bmod 13 = 11.$$

Допустим, среди участников восстановления секрета был мошенник. Без потери общности предположим, что это участник с долей номер 2 ($k_2 = 3$). При восстановлении секрета мошенник модифицировал свою долю k_2 и вернул значение $\hat{k}_2 = 4$. В этом случае значение восстановленного секрета равно:

$$\hat{s} = 4\cdot 5 + 7\cdot 8 + 5\cdot 1 = 11 \bmod 13 = 3.$$

За счёт линейности используемой схемы мошенник мог заранее высчитать величину, на которую восстановленный секрет будет отличаться от исходного. Получаем:

$$\delta_s = \hat{s} - s = (\hat{k}_2 - k_2)\cdot 5 + (k_3 - k_3)\cdot 8 + (k_5 - k_5)\cdot 1 = 5\cdot 1 \bmod 13 = 5.$$

Действительно, $\hat{s} = s + \delta_s \bmod p = 11 + 5 \bmod 13 = 3$. Однако, значение искажаемого секрета (внутреннее состояние) не было известно мошеннику, поэтому он мог привнести заданное искажение δ_s, но не обладал информацией о результирующем значении искажённого секрета до его восстановления.

Фактически, представленную линейную схему разделения секрета с мошенниками можно рассматривать как хранение секрета s в абстрактном запоминающем устройстве Σ. Таким образом, к задаче обнаружения факта восстановления изменённого секрета также можно применить модель дискретного канала с алгебраическими манипуляциями [19].

1.2.3 Привнесение помех в вычислительные устройства

Ещё одной областью использования модели канала со случайной структурой, описываемой моделью алгебраических манипуляций, является проектирование защищённых вычислительных устройств, устойчивых к внешним помехам. Известно, что реализации криптографических алгоритмов уязвимы к атакам по сторонним каналам, включая одну из наиболее эффективных атак — атаку по привнесённым помехам [42]. Она заключается в привнесении ошибок в работу криптографического модуля с последующим анализом результатов его функционирования. Было

показано, что наличие искажений и вычислительных ошибок в работе устройства может приводить к выявлению информации о параметрах системы (например, ключах шифрования). Ниже приведём подробное описание данной атаки, включающее перечисление основных методов наведения помех и методов анализа ошибочных криптографических вычислений.

В общем, атака по привнесённым помехам состоит из двух этапов. Первый заключается в наведении помех в определённый момент работы устройства. Зачастую, некорректную работу устройства вызывают с помощью различного рода физических воздействий (например, [43, 44]). Наиболее распространённые методы воздействия [45]:

– манипулирование тактовой частотой шифровального устройства. При заданных отклонениях тактирующего сигнала от нормы можно добиться изменения последовательности выполняемых команд вплоть до пропуска определённой инструкции. Подобные методы воздействия особенно применимы к смарт-картам, тактовый сигнал для которых предоставляется внешним считывающим устройством;

– изменение внутренней архитектуры шифратора (например, нарушение электрических контактов);

– манипулирование напряжением питания криптографического устройства. Отклонения в питании могут приводить к ошибкам работы устройства на определённых этапах, не приводя при этом к выведению устройства из строя. Например, понижение уровня питания при операциях чтения из памяти могут приводить к некорректным результатам чтения (которые зачастую могут быть предсказаны заранее).

– воздействие лазерным лучом или сфокусированным световым пучком. С его помощью можно изменять состояние ячеек памяти и влиять на условные переходы в исполняемом коде;

– воздействие переменным магнитным полем, приводящим к возникновению вихревых токов в цепях устройства. Это может приводить, например, к изменению значений, хранимых в памяти;

– значительное изменение температуры некоторой части шифратора;

– помещение устройства в сильное электромагнитное поле, приводящее к возникновению индукционного тока в проводящих элементах шифратора.

Далее осуществляется анализ результатов работы устройства в условиях помех с целью вычисления секретного параметра. Он выполняется с использованием аппарата простого или дифференциального анализ ошибок [46, 47]. Данные методы были успешно применены к большинству современных шифров.

Модель привнесённых помех

Атаки по привнесённым помехам могут быть классифицированы на основе способности злоумышленника контролировать такие параметры внедряемых помех, как время возникновения, местоположение, тип помех и конфигурацию возникающих ошибок [48, 49]. При условии наличия широкого арсенала методов и техник внедрения помех, доступных криптоаналитику, становится крайне трудно моделировать и предсказывать конфигурации ошибок, являющиеся результатом влияния помех на выходные данные атакуемого устройства. Например, в [50] показано, что кратность ошибок может регулироваться за счёт уменьшения напряжения электропитания до определённого уровня.

Однако, с переходом технологий к субмикронным размерам криптоаналитику становится всё труднее провоцировать заданные конфигурации ошибок в выходных данных атакуемого устройства [51]. Более того, все известные автору механизмы внедрения помех обеспечивают ограниченные пространственное и временное разрешения. Для примера, область возникновения помех при воздействии лазером, являющимся одним из наиболее эффективных инструментов криптоаналитика, определяется технологическими особенностями устройства, а также фокусной площадью лазерного пучка [51]. Время между двумя последовательными облучениями зависит от скорости перезарядки устройства и от задержки между запуском облучения и его осуществлением [49].

Несмотря на вышеперечисленные сложности для криптоаналитика, в данной диссертационной работе будет рассматриваться модель помехи, не включающая ограничения тех или иных методов её наведения. Другими словами, тип помех, кратность возникающих ошибок и другие характеристики атаки не будут привязаны к определённым методам внедрения помех, а будут по возможности обобщены на самый широкий случай. Это достигается за счёт использования модели дискретного канала с алгебраическими манипуляциями для описания привнесённых искажений.

Описывая особенности атак с привнесение помех, можно выделить два основных пункта. Во–первых, практика показывает, что криптоаналитик способен воздействовать на устройство таким образом, чтобы провоцировать помехи в его работе. К сожалению, гарантировать защиту устройства от всех потенциальных воздействий, которые могут привести к возникновению помех, не представляется возможным. Во–вторых, использование защищённого криптографического устройства обеспечивает то, что злоумышленник не может получить полного доступа к устройству, его внутренней архитектуре и состоянию. Действительно, снабжая критически важные блоки обработки данных соответствующими методами защиты от несанкционированного доступа (раздел А.4.1 приложения А), мы с большой долей вероятности лишаем злоумышленника возможности напрямую изучать внутреннее состояние этих блоков и их содержимое. Кроме того, при проектировании современных защищённых устройств уже учитывается угроза со стороны атак по сторонним каналам. В связи с этим, в чипах, помимо средств защиты от непосредственного доступа к устройству, реализуются контрмеры, направленные против наиболее при-

меняемых атак (раздел А.4 приложения А). Так, в большинстве работ [52–55], рассматривающих методы защиты вычислительных устройств от атак с привнесением помех, подразумевают, что устройство защищено от зондирования, воздействия на тактовый сигнал [48], управляющие цепи (например, конечные автоматы и регистры с текущим состоянием устройства) [56] и блок обнаружения ошибок с помощью соответствующих контрмер. Таким образом, при выборе модели помех используется предположение, что злоумышленник не имеет прямого доступа к внутреннему состоянию устройства, но способен провоцировать в нём помехи. В поддержку данного предположения можно привести получившие в последнее время наибольшее распространение полуагрессивные атаки по сторонним каналам. Они не подразумевают создания дополнительных электрических контактов или декапсуляции чипа (раздел А.3 приложения А). Описанные выше положения относительно привнесения помех в работу вычислительных устройств нашли отражение в модели канала с алгебраическими манипуляциями.

Методы обработки результатов внедрения помех

В качестве метода обработки результатов работы устройства при условии помех возможны различные варианты в зависимости от особенностей атакуемого алгоритма. Согласно классификации по методу, используемому для анализа результатов, атаки по привнесённым помехам делятся на два типа:

– с простым анализом помех;

– с дифференциальным анализом помех.

Суть дифференциального анализа помех сводится к вычислению пары шифротекстов, один из которых является корректным, а второй — ошибочным (вычисленным при условии наличия помех). Сравнение этих шифротекстов может приводить к выявлению информации о значении секретного ключа. Все остальные существующие способы анализа относятся к простому анализу помех.

Большое количество атак с простым анализом помех приведено в [46]. Наиболее наглядным примером такой атаки является следующий сценарий, приведённый в статье. Пусть криптоаналитик может модифицировать содержимое памяти, хранящей часть исполняемого кода (интересные практические эксперименты с модификацией значений, хранимых в памяти, описаны в [57, 58]). Изменение кода осуществляется таким образом, чтобы устройство, исполняя его, прочитало и вывело, например, на встроенный дисплей, значение секретного ключа. Другой сценарий атаки заключается в изменении значения определённых переменных и констант. Например, уменьшив до единицы значение константы, определяющей количество раундов шифрования симметричного шифра, криптоаналитик может полностью компрометировать секретные параметры симметричного шифра.

Дифференциальный анализ помех

Далее приведём пример атаки на RSA с дифференциальным анализом помех. Первая работа, поставившая вопрос об угрозе вычислительных ошибок для криптографических алгоритмов, вышла в сентябре 1996 года [59]. В ней Бонэ (Boneh) и другие из лаборатории Bellcore предложили атаку на криптосистемы с открытым ключом, такие как RSA. В [47] Бихам (Biham) и Шамир (Shamir) расширили эту атаку на случай симметричных криптосистем, таких как DES, и назвали такую атаку дифференциальным анализом помех (differential fault analysis, DFA). Кроме того, Бихам и Шамир разработали техники для взлома засекреченных шифров с помощью DFA, когда внутренняя структура шифра и выполняемые им операции неизвестны. Также, в [47] указывалось, что дифференциальный анализ ошибок может быть использован для взлома шифров IDEA, RC5, FEAL и поточных шифров. В дальнейшем, работы по DFA сконцентрировались на AES как на наиболее популярном на данный момент шифре. Большинство атак против AES использует свойства шифрующей функции [60–64], остальные нацелены на расписание ключей [65–67]. Необходимо отметить, что модели атак на AES очень различаются в зависимости от размера искажаемых данных (один бит, один байт и так далее), от этого зависит и необходимое для выявления ключа количество ошибочных шифротекстов.

Для демонстрации принципа DFA рассмотрим пример атаки на алгоритм RSA, реализованный на основе китайской теоремы об остатках (chinese remainder theorem, CRT) из работы [59]. В работе показано, что для успешного взлома CRT–RSA достаточно внедрения единственной ошибки в процессе вычисления электронной цифровой подписи (ЭЦП). Приведём описание этой атаки.

Рассмотрим систему, использующую алгоритм RSA для генерации ЭЦП. Пусть $n = p \cdot d$ есть произведение двух больших простых чисел. Для подписи сообщения x система вычисляет $x^d \bmod n$, где d есть секретная экспонента (часть закрытого ключа). Под x подразумевается целое число в диапазоне от 1 до n. Надёжность алгоритма основывается на сложности факторизации числа n. Фактически, если известны числа p и q, то злоумышленник легко взломает шифр, то есть сможет подписывать документы без знания секретной экспоненты.

Зачастую для упрощения реализации алгоритма RSA используют китайскую теорему об остатках. Сначала вычисляют $E_1 = x^d \bmod p$ и $E_2 = x^d \bmod q$. Далее, используя CRT, вычисляют подпись $E = x^d \bmod n$. Согласно теореме, для заданных p и q всегда найдутся целые числа a, b такие что:

$$\begin{cases} a \equiv 1 \pmod{p} \\ a \equiv 0 \pmod{q} \end{cases} \quad \text{и} \quad \begin{cases} b \equiv 0 \pmod{p} \\ b \equiv 1 \pmod{q}. \end{cases}$$

Из этого следует, что

$$E = aE_1 + bE_2 \pmod{n}.$$

Таким образом, подпись представляет собой линейную комбинацию E_1 и E_2. Подобная реализация алгоритма RSA позволяет значительно уменьшить аппаратную или временную избыточность за счёт уменьшения размеров множителей.

Атака, предлагаемая в [59], заключается в вычислении двух подписей для одного сообщения: одна из них является корректной, в то время как другая — ошибочной. Пусть M есть сообщение, тогда $E = M^d \bmod n$ есть корректная подпись этого сообщения. Пусть $\hat{E} = M^d \bmod n$ есть ошибочная подпись. Напомним, что подписи E и $\hat{E}$ вычислялись следующим образом:

$$E = aE_1 + bE_2 \pmod{n} \quad \text{и} \quad \hat{E} = a\hat{E}_1 + b\hat{E}_2 \pmod{n}.$$

Допустим, в процессе вычисления одного их чисел $\hat{E}_1, \hat{E}_2$ произошла ошибка. Без потери общности предположим, что ошибка в $\hat{E}_1$ (то есть $\hat{E}_1 \neq E_1 \bmod p$), при этом $\hat{E}_2$ корректна (то есть $\hat{E}_2 = E_2$). Тогда

$$\text{НОД}(E - \hat{E}, n) = \text{НОД}(a(E_1 - \hat{E}_1), n) = q,$$

из чего следует простота разложения n на сомножители.

Получаем, что злоумышленник, имея две подписи одного сообщения (корректную E и ошибочную $\hat{E}$), способен вычислить p и q. Таким образом, внедряя единственную ошибку в процесс вычисления ЭЦП, криптоаналитик получает возможность взломать криптографическую систему. Стоит отметить, что данная атака работает при очень общей модели помехи (тип помех и кратность возникающих ошибок не играют роли).

В [68] А. Ленстра (A. Lenstra) показал, что для взлома достаточно единственной ошибочной подписи известного сообщения M. Пусть $E = M^d \bmod n$, а $\hat{E}$ есть ошибочная подпись с описанным выше искажением, то есть $E \equiv \hat{E} \bmod q$, но $E \not\equiv \hat{E} \bmod p$. Тогда

$$\text{НОД}(M - \hat{E}^e, N) = q,$$

где e есть открытая экспонента, используемая для подтверждения подписи. Таким образом, зная отрытый текст M и единственную ошибочную подпись $\hat{E}$, злоумышленник способен разложить n на множители. Данная атака использует простой метод анализа помех (раздел 1.2.3) и является эффективным примером его применения.

Кроме того, в работе показано, что аналогичной угрозе подвержена вероятностная схема подписи Рабина. Некоторые другие криптографические системы (например, протокол Фиата—Шамира и схема Шнорра) также уязвимы к внедрению ошибок, однако, успешная атака требует внедрения большего количества ошибок. Особое внимание в работе уделяется защищённости от помех центров сертификации.

Результаты, представленные в [59], показывают, что существуют классы теоретико-числовых задач в криптографии, вычислительные ошибки в которых могут приводить к выявлению секретных параметров злоумышленником.

Заключение

Таким образом, остро стоит задача обеспечения целостности информации, обрабатываемой вычислительными устройствами, с целью защиты от привнесённых помех и вычислительных ошибок. Выше было показано, что привносимые помехи могут быть описаны моделью алгебраических манипуляций. Следовательно, разработка и исследование методов, обеспечивающих повышение помехоустойчивости устройств, может быть эффективно использовано при проектировании вычислительных устройств, устойчивых к привнесённым помехам.

1.3 Выводы

При разработке и исследовании методов нелинейного кодирования была выбрана модель канала с алгебраическими манипуляциями, описывающая широкий класс каналов со случайной структурой помех. Из представленных примеров практических приложений исследуемой модели канала следует, что актуальной задачей является разработка и совершенствование методов повышения помехоустойчивости обработки (передачи, хранения) информации техническими системами при условии алгебраических манипуляций.

В следующей главе будут приведены основные методы защиты автоматизированных систем обработки информации от помех. Будет показано, что классические методы повышения помехоустойчивости (с использованием линейных кодов) не всегда эффективны в условиях искажений, описываемых моделью алгебраических манипуляций. Использование модели канала с алгебраическими манипуляциями способствует разработке методов кодирования, основанных на нелинейных кодах, и даёт возможность провести анализ их эффективности. Разработке и анализу нелинейных методов кодирования посвящены главы 3 и 4.

2 Обзор основных методов повышения помехоустойчивости

В данной главе рассматриваются основные методы защиты технических систем от помех, угрожающих целостности данных.

2.1 Методы защиты

Классическим подходом для проектирования систем, устойчивых к помехам, является использование техники параллельного обнаружения ошибок (concurrent error detection, CED) [69–74]. Принцип, лежащий в основе схем параллельного обнаружения ошибок, следующий: предположим, что рассматриваемая система реализует функцию g и выдаёт значение $g(i)$ на входное значение i. Схема параллельного обнаружения ошибки обычно содержит модуль, который независимо вычисляет — «предсказывает» — некоторую специальную характеристику выходного значения $g(i)$. Далее, блок проверки сравнивает предсказанную характеристику с характеристикой, вычисленной на основе реального выхода устройства. В случае их несовпадения генерируется сигнал об ошибке. Некоторые примеры характеристик $g(i)$: само значение $g(i)$, чётность $g(i)$, количество нулевых или единичных битов в $g(i)$ и так далее. Общая схема параллельного обнаружения ошибок приведена на рисунке 2.1. Закрашенные блоки на рисунке обозначают аппаратную избыточность, необходимую для реализации данного метода защиты. Любая схема CED характеризуется набором или классом ошибок, при наличии которых нарушается целостность данных, то есть происходит их необнаруживаемое искажение [75]. Под обеспечением целостности данных понимается такое поведение системы, при котором система либо выдаёт корректные выходные данные, либо сигнализирует о выдаче ошибочных данных. Основную угрозу для целостности данных представляют необнаруживаемые ошибки. Для заданного метода защиты будем считать необнаруживаемой такую ошибку, появление которой не может быть обнаружено с помощью этого метода. Другими словами, ошибка e является необнаруживаемой в том случае, если для любой разрешённой обрабатываемой (передаваемой, хранящейся) комбинации $v \in V$ выполняется равенство $e + v = u \in V$, где V есть множество разрешённых обрабатываемых (передаваемых, хранящихся) комбинаций. Множество необнаруживаемых ошибок, внедрение которых будет успешным при любых обрабатываемых данных, будем называть постоянной конфигурацией необнаруживаемых ошибок.

Важно отметить, что блок проверки (или блок обнаружения ошибки) должен быть надёжно защищён от несанкционированного доступа. Например, блок проверки может быть экраниро-

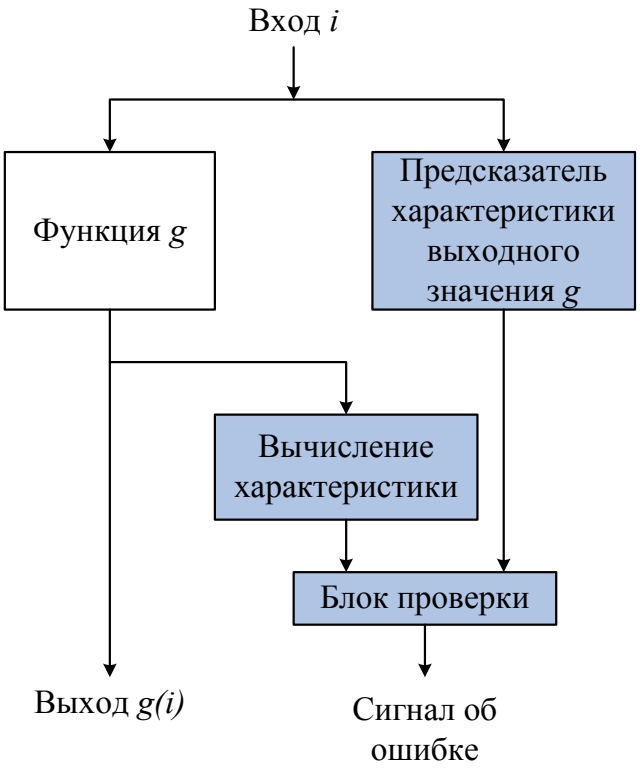


Рисунок 2.1 – Общая схема параллельного обнаружения ошибок

ван, защищён с помощью датчиков, дублирован. В противном случае, защита данных не может быть осуществлена. Таким образом, необходимость защиты всего устройства от физических воздействий заменяется защитой одного небольшого блока. Это позволяет значительно уменьшить размер чипа и затраты на его создание.

В большинстве случаев, как, например, на рисунке 2.1, схемы CED используют аппаратную избыточность (использование блоков предсказания и проверки). Схемы с временной избыточностью (например, [76–78]) также могут быть использованы для параллельного обнаружения ошибок. Временная избыточность негативно сказывается на производительности системы, однако стоимость аппаратных затрат в таком случае обычно значительно ниже, чем при аппаратной избыточности.

Примером использования схем CED с временной избыточностью является метод защиты симметричных шифров от помех, описанный в работе [78]. Этот метод, основанный на симметрии и обратимости шифров с закрытым ключом, осуществляет шифрование (дешифрование), после чего выполняется обратная операция — дешифрование (шифрование). Если в процессе работы устройства ошибки не было, то обратная операция должна привести к исходным данным. Данный метод может быть реализован на различных уровнях (уровень алгоритма, раунда, операции и т.д.). Например, при реализации на уровне операций каждая выполняемая шифратором операция проверяется обратным действием (циклический сдвиг операнда вправо проверяется циклическим сдвигом результата исходного сдвига, но уже влево). Плюсом реализации такого метода на уровне операции является возможность определить местоположение возникновения помехи. Как уже было сказано, производительность системы с таким методом защиты

значительно снижается, так как обратную операцию можно выполнить только после завершения исходной.

Ввиду меньшей задержки обработки данных (и, как следствие, более высокой производительности системы) наибольшее распространение получили схемы параллельного обнаружения ошибок с аппаратной избыточностью. При этом, основными характеристиками выходного значения являются само значение (дублирование оборудования) и проверочные символы линейного кода (использование линейных помехоустойчивых кодов (ЛПК)). Примером использования дублирования оборудования является метод защиты вычислительного модуля из [79], где каждый аппаратный модуль дублируется резервным блоком. Примеры использование ЛПК для защиты можно найти в [80, 81]. Обычно используются такие коды, как код с проверкой на чётность, код Хэмминга, квадратично–вычетные коды [82].

В последнее время появились работы по использованию нелинейных помехоустойчивых кодов (НПК) в схемах CED для защиты технических систем [24,25,83,84]. Некоторые классы НПК обладают высокой обнаруживающей способностью, кроме того, их нелинейность может быть использована для уменьшения вероятности искажения данных.

В данной диссертационной работе будут рассматриваться схемы параллельного обнаружения ошибок, основанные на аппаратной избыточности. Это обусловлено тем, что они не так критически сказываются на производительности системы, а также являются более универсальными и могут быть использованы для различных устройств без значительных модификаций. Далее будут рассмотрены основные методы защиты технических систем от помех:

1. Дублирование оборудования;
2. Использование линейных помехоустойчивых кодов;
3. Хеширование;
4. Использование нелинейных помехоустойчивых кодов.

Для них будут представлены общие архитектуры схем параллельного обнаружения ошибок. Отдельное внимание будет уделено классам ошибок, которые нарушают целостность данных в системе.

2.1.1 Дублирование оборудования

Дублирование оборудования является наиболее очевидным методом борьбы с искажениями. При защите аппаратного блока создаётся его копия, которая используется наравне с оригинальным устройством [79, 85, 86]. Выходные данные обоих блоков сравниваются, и при наличии различий между ними принимается решение об искажениях данных. Формально, характеристикой выходного значения $g(i)$ выступает само $g(i)$. Схема обнаружения ошибок с помощью дублирования оборудования приведена на рисунке 2.2. Как и в большинстве работ на эту тему,

предполагается, что блок обнаружения ошибки защищён от внешнего воздействия и вмешательства в его работу. Стоит отметить, что количество дублирующих блоков может быть больше

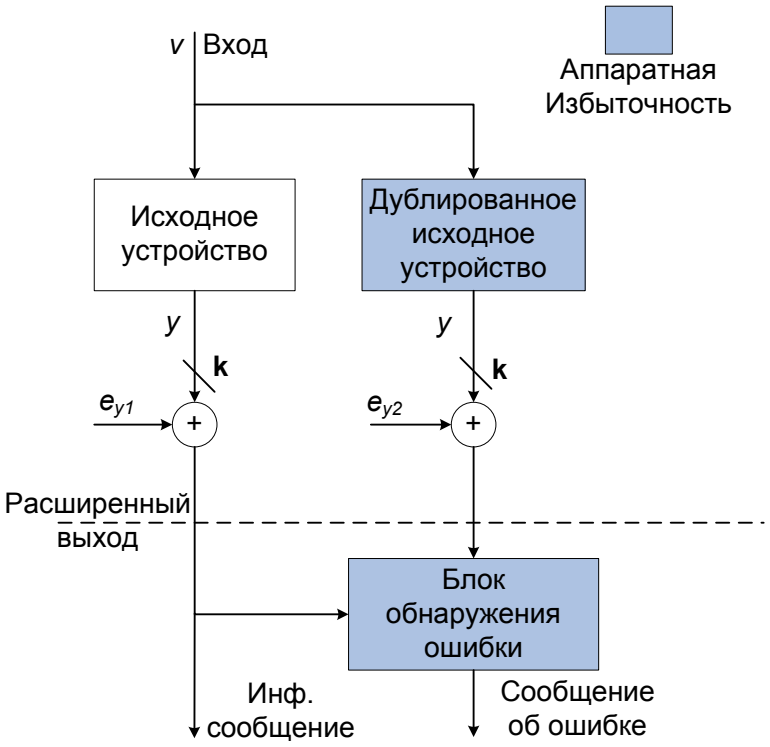


Рисунок 2.2 – Схема обнаружения ошибок с использованием дублирования оборудования

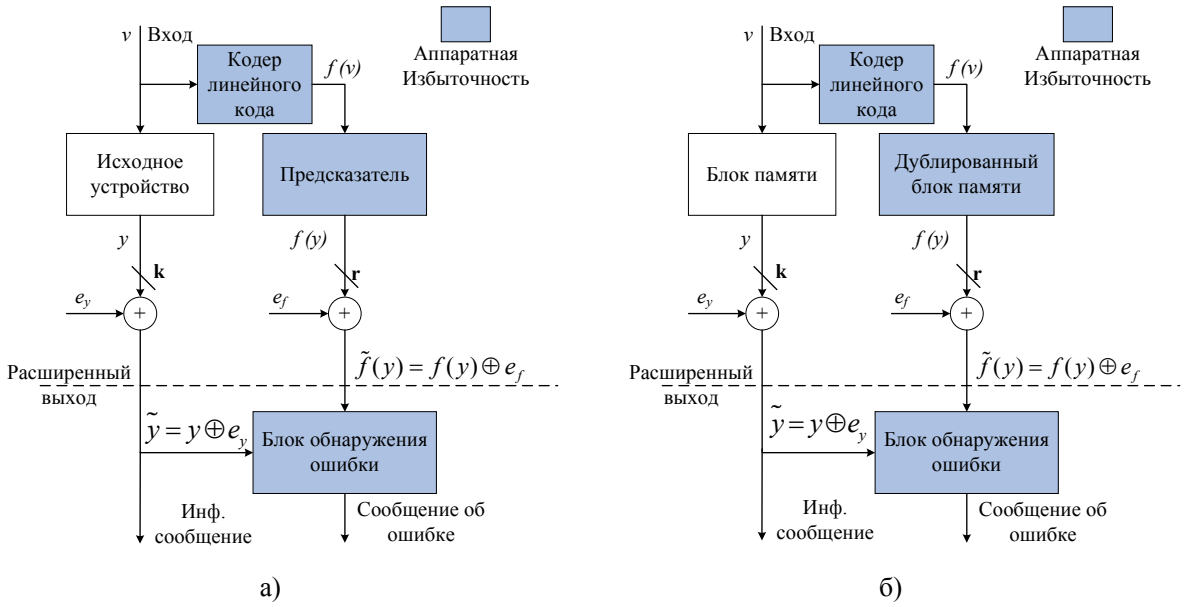
одного — в этом случае помимо обнаружения помех может приниматься решение относительно значения выходных данных с использованием мажоритарного принципа. При защите блока памяти возможно использование как аппаратной избыточности (дублирование блока памяти), так и информационной (дублирование данных, записываемых в память).

Легко заметить, что возникновение одинаковых ошибок e_{y1} и e_{y2} в исходном и дублирующем блоках приводит к тому, что нарушается целостность данных — устройство выдаёт ошибочные данные, не подавая сигнал об ошибке. Количество таких комбинаций ошибок для q-ичных данных размерности k равно $q^k - 1$ из q^{2k} возможных (исключается случай, когда $e_{y1} = e_{y2} = 0$). Следовательно, возникновение одинаковых помех в блоках гарантированно приводит к необнаруживаемой ошибке в работе устройства. Возникновение такой ошибки в определённых случаях представляется достаточно вероятным событием, что сводит к минимуму надёжность и помехоустойчивость технической системы при алгебраических манипуляциях. Стоит отметить, что данная схема была разработана и используется для защиты от помех естественного происхождения. характерных для классических каналов с шумом, против которых она является достаточно эффективной.

Таким образом, дублирование оборудования не обеспечивает защищённость технических систем от рассматриваемой модели помехи — даже при слабой модели алгебраической манипуляции возможно необнаруживаемое возникновение искажения данных.

2.1.2 Линейное помехоустойчивое кодирование

Другим классическим методом защиты от помех является использование линейных помехоустойчивых кодов [5, 14, 80–82]. Наибольшее распространение получили систематические коды, что объясняется простотой их использования в схемах CED. В качестве характеристики, которую вычисляет предсказатель, используется значение проверочных символов линейного кода. Принцип использования ЛПК в схемах параллельного обнаружения ошибок приведён на рисунке 2.3. В качестве дополнительного оборудования, необходимого для обнаружения помех, используются кодер, предсказатель и блок обнаружения ошибок.



а) б)

Рисунок 2.3 – Пример схемы параллельного обнаружения ошибок с использованием линейных помехоустойчивых кодов: а) для защиты линейных блоков вычислительных устройств б) для защиты блоков памяти

Несмотря на то, что кодер и предсказатель изображены как отдельные блоки, формально, предсказателем является их совокупность. Но для более наглядной демонстрации использования помехоустойчивых кодов в схемах CED было решено на рисунке выделить кодер в отдельный блок. При таком делении предсказатель сводится к блоку, осуществляющему преобразование аналогично исходному устройству, но, возможно, над данными другой размерности (если $k \neq r$). Кодер вычисляет функцию (характеристику) от этого значения. Линейность операций кодирования и преобразования, выполняемого исходным блоком, позволяет располагать блоки в порядке, обратном последовательности операций (эта ситуация продемонстрирована на рисунке 2.3). Кроме того, за счёт линейности преобразований возможна реализация единого блока, осуществляющего итоговое преобразование.

Исходный модуль осуществляет линейное преобразование h входных данных, при котором значение v отображается в выход $h(v) = y$. Допустим, в устройстве возникают помехи, приводящие к возникновению ошибки e_y. Тогда выход исходного устройства можно записать как

$\tilde{y} = y + e_y$. Данные v, поступающие в защищаемый блок, также поступают в кодер линейного кода, вычисляющий проверочные символы кода с помощью кодирующей функции f. Далее расположен предсказатель, вычисляющий значение функции h от проверочных символов $f(v)$. Линейность осуществляемых преобразований приводит к тому, что выход предсказателя можно записать как $h(f(v)) = f(h(v)) = f(y)$. Воздействие помех оказывает влияние и на функционирование предсказателя, выход которого обозначим через $\tilde{f}(y) = f(y) + e_f$, где e_f есть величина возникающей ошибки. Таким образом, в блок обнаружения ошибок поступают искажённый выход исходного устройства $\tilde{y}$ и искажённый выход предсказателя $\tilde{f}(y)$. Блок обнаружения ошибки проверяет, является ли поступивший вектор $\left(\tilde{y}|\tilde{f}(y)\right)$ словом используемого линейного кода. Эта проверка может быть реализована различными способами, например, умножением на проверочную матрицу кода. Альтернативным способом является вычисление проверочных символов сообщения $\tilde{y}$ с последующим сравнением с выходом предсказателя. В случае, если имеет место равенство

$$f(\tilde{y}) = \tilde{f}(y),$$

$$f(y + e_y) = f(y) + e_f,$$

то слово принадлежит коду, и ошибка $e = (e_y, e_f)$ не будет обнаружена. В противном случае система информируется об обнаружении ошибки.

При защите блока памяти осуществляются аналогичные действия: входные данные сначала подвергаются процедуре кодирования, после чего записываются в память вместе с проверочными символами, формируя кодовое слово. При чтении данных их целостность проверяется за счёт проверки принадлежности прочитанного значения линейному коду.

При использовании схем параллельного обнаружения ошибок техническая система зачастую разбивается на элементы, каждый из которых защищается с помощью помехоустойчивого кода. Такое деление позволяет упростить процедуру проектирования предсказателя, а также определять элементы устройства, работающие с помехами.

Подобное использование линейных кодов в схемах параллельного обнаружения ошибки позволяет использовать помехоустойчивые характеристики кодов для обеспечения целостности данных. Линейные коды концентрируют свои обнаруживающие и исправляющие способности на конкретных конфигурациях ошибок. Например, коды с проверкой на чётность обнаруживают ошибки нечётной кратности, при этом не обнаруживая ошибки чётной кратности [14]. Использование линейного помехоустойчивого кода с минимальным расстоянием d_0 гарантирует обнаружение $s \leq d_0 - 1$ или исправление $t \leq \lfloor (d_0 - 1)/2 \rfloor$ ошибок. В режиме совместного обнаружения и исправления ошибок код обеспечивает обнаружение s и исправление $t < s$ ошибок, если минимальное расстояние $d_0 \geq t + s + 1$. Например, расширенный код Хэмминга с $d_0 = 4$ гарантированно исправляет однократные ($t = 1$) и обнаруживает двукратные ($s = 2$) ошибки.

Пусть код $C \in GF(q)^n$ используется в режиме обнаружения ошибок, где n есть длина кода. Допустим, что в системе происходит искажение обрабатываемых данных $c \in C$ на величину ошибки e, то есть $\tilde{c} = c + e$. Если $\tilde{c} \notin C$, то гарантируется обнаружение ошибки. В против-

ном случае, когда $\tilde{c} \in C$, ошибка не будет обнаружена кодом. Из линейности кодов следует, что в этом случае $e \in C$. Следовательно, если возникающие искажения соответствуют кодовым словам, то они гарантированно не могут быть обнаружены данным методом повышения помехоустойчивости на основе ЛПК. Если обозначить через k размерность кода, то количество необнаруживаемых ошибок равно количеству кодовых слов минус один, то есть $q^k - 1$ (исключается нулевое кодовое слово).

Теперь рассмотрим ситуацию, когда код C работает в режиме исправления ошибок. Если код исправляет до t ошибок, то возникновение любой ошибки e, вес Хэмминга которой не превышает t, не приведёт к нарушению целостности данных, так как ошибка будет исправлена. В случае, когда ошибка $e \in C$, то она является необнаруживаемой. Если же $e \notin C$ и вес ошибки превышает t, то тут возможны два варианта: либо ошибка будет обнаружена, либо будет произведено декодирование в другое кодовое слово. Последний вариант приводит к тому, что наличие искажений обнаруживается, но выходное значение является некорректным. Устойчивость системы к таким ситуациям зависит от поведения, предусмотренного при её проектировании. Количество ошибок, приводящих к декодированию в другое слово, равно $(q^k - 1) \cdot \sum_{i=1}^{t} C_n^i (q-1)^i$. Таким образом, количество необнаруживаемых ошибок в режиме исправления остаётся прежним, но добавляется возможность коррекции искажений невысокой кратности (до t включительно). Кроме того, зачастую, ситуация ошибочного декодирования в значительной степени нарушает целостность данных, негативно сказываясь на помехоустойчивости. Режим исправления ошибок часто используется в схемах защиты памяти, что позволяет корректно обрабатывать искажения, вызванные естественными причинами (классическая модель канала с шумом), но не произвольными ошибками (модель канала с алгебраическими манипуляциями).

Использование кода в режиме совместного обнаружения и исправления ошибок также приводит к наличию как $q^k - 1$ необнаруживаемых ошибок, так и ошибок, приводящих к декодированию в другое кодовое слово.

Таким образом, q-ичный линейный код размерности k в силу своей линейности имеет $q^k - 1$ необнаруживаемых ошибок, соответствующих кодовым словам. Следовательно, даже при слабой модели манипуляции возможно возникновение конфигураций ошибок, которые не будут обнаружены кодом. Получаем, что использование линейных кодов, целесообразное при классической модели помех в канале с шумом, не является эффективной мерой для защиты вычислительных устройств от искажений, описываемых моделью алгебраических манипуляций. Стоит отметить, что дублирование оборудования можно рассматривать как использование линейного кода–повторения (с соответствующим количеством необнаруживаемых ошибок $q^k - 1$).

2.1.3 Хеширование

В инфокоммуникационных системах для обеспечения целостности данных зачастую используются хеш–функции, отображающие входные данные произвольной длины в выходное значение фиксированного размера [40]. Используются как криптографические хеш–функции, стойкие

к коллизиям первого и второго рода, так и более простые по структуре и проще реализуемые в аппаратуре хеш–функции на основе контрольной суммы.

Хеширование предназначено для обнаружения искажений обрабатываемых данных. Принцип использования хеш–функций аналогичен применению помехоустойчивого кодирования и заключается в вычислении информационной избыточности, называемой хешем или сводкой сообщения. Повторное вычисление хеша после передачи данных по каналу связи (обработки, хранения) и сравнение его с исходным значением хеша позволяет проверять целостность переданных (обработанных, хранящихся) данных.

В качестве примера криптографически стойких хеш–функций можно привести Secure Hash Algorithm 1 (SHA–1) с размером хеша 160 бит, ГОСТ Р 34.11–2012 с размером хеша 256 или 512 бит, Message Digest 6 (MD–6) с переменным размером хеша от 1 до 512 бит, а также семейство хеш-функций Secure Hash Algorithm Version 2 (SHA–2), предназначенных для генерации хеша произвольной длины [40, 87]. Среди хеш–функций на основе контрольной суммы выделяются функции на основе циклического избыточного кода (cyclic redundancy code, CRC) с достаточно большим диапазоном размеров хешей [87].

Легко заметить, что относительно простые хеш–функции на основе контрольных сумм, зачастую являющиеся частным случаем использования ЛПК (например, хеш на основе CRC), не обеспечивают защиту от искажений, описываемых моделью алгебраических манипуляций. Стойкие хеш–функции позволяют обеспечить обеспечение целостности при алгебраических манипуляциях (в случае сильных манипуляций — при использовании криптографической соли (строки случайных данных)). Однако, фиксированный размер хешей и их относительно высокая вычислительная сложность накладывают значительные ограничения на их применимость в технических системах. Требования, предъявляемые к стойким хеш–функциям, значительно превышают требования к методам защиты от алгебраических манипуляций, поэтому имеет смысл использовать более простые методы, о которых будет сказано далее.

Необходимо указать, что имитовставки, использующиеся для обеспечения целостности и защиты от фальсификации передаваемой информации, не рассматриваются в данной диссертационной работе, так как требуют использования секретных ключей [40].

2.1.4 Нелинейное помехоустойчивое кодирование

Естественным решением проблемы линейных кодов и хеширования стало использование нелинейных кодов. Среди нелинейных кодов существуют классы кодов, которые не имеют постоянной конфигурации необнаруживаемых ошибок — каждая ошибка обнаруживается с ненулевой вероятностью. Напомним, что для заданного метода защиты будем считать необнаруживаемой такую ошибку, появление которой не может быть обнаружено с помощью этого метода. Другими словами, множество конфигураций ошибок, возникновение которых не может быть обнаружено при любых обрабатываемых данных, является пустым. Таким образом, вне зависимости от того, какая конфигурация ошибок возникнет, данные классы кодов гарантируют её обнаружение

с заданной вероятностью. Наибольшее распространение получили систематические нелинейные коды, так как они обеспечивают минимальную задержку декодирования, а также легко применимы в схемах параллельного обнаружения ошибки. Кроме того, далее будут рассматриваться коды над конечными полями характеристики два, однако большинство результатов может быть обобщено для любых полей.

Нелинейные функции

Вычисление проверочных символов нелинейных кодов осуществляется с помощью нелинейных функций. Пусть f есть функция, отображающая элементы поля $GF(2^k)$ в поле $GF(2^r)$: $a \to b$. Нелинейность этой функции может быть измерена с помощью величины $D_a f(x) = f(x+a) - f(x)$, называемой производной по направлению a [88]. Пусть

$$P_f = \max_{0 \neq a \in GF(2^k)} \max_{b \in GF(2^r)} \frac{||\{x \in GF(2^k) : D_a f(x) = b\}||}{||\{x\}||}. \tag{2.1}$$

Величина P_f является показателем нелинейности функции: чем меньше ее значение, тем выше нелинейность функции. Для линейных функций $P_f = 1$.

Двоичная функция $f : GF(2^k) \to GF(2^r)$ называется совершенной нелинейной (perfect nonlinear, или, сокращённо, PN–функция), если $P_f = 1/2^r$. Примером совершенной нелинейной функции является функция $f(x) = \sum_{i=1}^{s} x_{2i-1} \cdot x_{2i}$, отображающая элемент $x = (x_1, x_2, ..., x_{2s}) \in GF(2^{2sr})$ в поле $GF(2^r)$, где $x_i \in GF(2^r)$. Требование к функции быть совершенной нелинейной является достаточно жёстким, в следствие чего количество таких функций крайне невелико, а для многих важных значений параметров они просто не существуют [88]. В связи с этим большое внимание уделяется почти совершенным нелинейным функциям (almost perfect nonlinear, или, сокращённо, APN–функции). Кроме того, важным фактором исследования APN–функций является то, PN–перестановки для конечных полей характеристики два не существуют.

Почти совершенной нелинейной называется такая функция $f : GF(2^k) \to GF(2^r)$, для которой количество решений уравнения $f(x+a) - f(x) = b$ среди всех $0 \neq a \in GF(2^k)$ и $b \in GF(2^r)$ не превосходит двух. Таким образом, именно APN–функции обеспечивают минимальный показатель нелинейности $P_f = 2/2^k = 2^{1-k}$ преобразований, осуществляемых при перестановках в поле $GF(2^k)$. Примером такой APN–перестановки является $f(x) = x^3$, $x \in GF(2^k)$. Вопрос о существовании APN–перестановок при чётном k до сих пор остаётся открытой проблемой [88].

Подробная информация относительно нелинейных функций представлена в [88–91].

Надёжные коды

Одним из классов нелинейных кодов, используемых для защиты от помех, являются *надёжные коды, обнаруживающие ошибки* [6]. В некоторых работах они называются также *слабо защищёнными кодами, обнаруживающими алгебраические манипуляции* (weakly secure algebraic manipulation detection codes) [27], но в данной работе мы будем использовать терминологию из

работ профессора Марка Карповского (Mark Karpovsky) [24,28,92] и называть их надёжными кодами. Суть этих кодов заключается в таком выборе кодовых слов, при котором минимизируется количество одинаковых разностей между всеми парами слов. Другими словами, минимизируется вероятность того, что прибавление некоторой величины к кодовому слову приведёт к переходу в другое кодовое слово.

Код C называется R–надёжным, если мощность пересечения кода со всеми его сдвигами $\tilde{C} = \{\tilde{c} : \tilde{c} = c + e, c \in C, e \in GF(2^n), e \neq 0\}$ ограничена сверху величиной R [24]:

$$R = \max_{0 \neq e \in GF(2^n)} ||\{c : c \in C, c + e \in C\}|| < ||C||. \tag{2.2}$$

Равномерно надёжным называется код, мощность пересечения которого со всеми сдвигами равна R. Необходимо отметить, что для линейных кодов $R = M$. Для краткости будем называть R–надёжные коды надёжными. Графическое изображение определения надёжного кода представлено на рисунке 2.4.

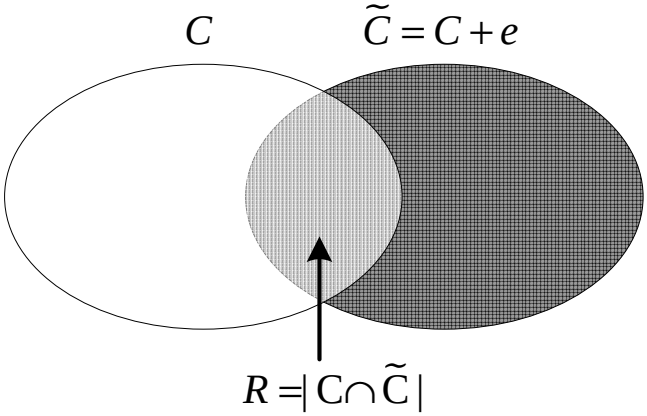


Рисунок 2.4 – Графически представленное определение надёжности

Таким образом, для каждой ненулевой ошибки e найдётся не более $R < ||C||$ кодовых слов, прибавление ошибки к которым даст кодовые слова, то есть ошибка не будет обнаружена. Отсюда следует, что вероятность необнаружения ошибки кодом, мощность которого равна $||C|| = M$, при условии равновероятности передаваемых сообщений ограничена сверху величиной

$$P^w_{undet} \leq R/M.$$

Для равномерно надёжных кодов в последней формуле имеет место равенство. Для R–надёжного кода вероятность наихудшего случая пропуска ошибки не превышает R/M при условии равновероятности слов надёжного кода. Очевидно, что надёжные коды, обладающие минимальным значением R для заданного M, также будут иметь наименьшую вероятность пропуска ошибки и, следовательно, гарантируют заданный уровень помехоустойчивости в случае непредсказуемого распределения ошибок, так как вероятность наихудшего случая ограничена сверху.

R–надёжный код длины n бит с M кодовыми словами, удовлетворяющий равенству $M^2 - M = R(2^n - 1)$, является совершенным [24]. Совершенные надёжные коды эквивалентны классическим комбинаторным структурам, таким как разностные множества и симметричные дизайны [93]. В [94] было показано, что все симметричные конструкции над двоичными полями

и, следовательно, совершенные надёжные двоичные коды существуют только для чётных размерностей и ограниченного набора параметров. Более того, систематические надёжные коды, которые наиболее часто используются на практике для обнаружения ошибок в вычислительных устройствах в силу разделения информационных и проверочных символов, не могут быть совершенными. Соответствующая граница на параметры систематических надёжных кодов будет представлена в разделе 3.1.

Общий принцип построения надёжных кодов приведён в следующей теореме из [24].

Утверждение 2.1. *Пусть f представляет собой нелинейную функцию, которая отображает $GF(2^k)$ в $GF(2^r)$, где $k \geq r$. Набор векторов, получающихся в результате конкатенации $(y|f(y))$, где $y \in GF(2^k)$ и $f(y) \in GF(2^r)$, образует надёжный систематический код с параметрами $R = 2^k \cdot P_f$, $n = k + r$ и $M = 2^k$ (здесь и далее символом «|»обозначена конкатенация).*

Примером кодовой конструкции, использующей PN–функцию является следующая [24]. Пусть $y = (y_1|y_2|\ldots|y_{2s}|y_{2s+1})$, $y_i \in GF(2^r)$, $s \geq 1$. Вектор $y \in GF(2^{(2s+1)r})$ принадлежит коду C, если

$$y_1 \cdot y_2 + y_3 \cdot y_4 + \ldots + y_{2s-1} \cdot y_{2s} = y_{2s+1}.$$

Функция $\sum_{i=1}^{s} y_{2i-1} \cdot y_{2i}$ является совершенной нелинейной функцией из $GF(2^{2sr})$ в $GF(2^r)$. Полученный код является надёжным кодом с параметрами $R = 2^{(2s-1)r}$, $n = (2s+1)r$ и $M = 2^{2sr}$. Данный код при $r = 1$ может быть использован вместо линейного кода с проверкой на чётность. Получаемый систематический надёжный код обладает той же избыточностью, что и линейный. Принципиальное отличие кодов заключается в том, что надёжный код будет пропускать ошибку с вероятностью не более $P^w_{undet} \leq 1/2$ вне зависимости от кратности ошибки, обеспечивая тем самым возможность обнаружения ошибок вне зависимости от их распределения. Линейный же код, как обсуждалось в разделе 2.1.2, не может гарантировать заданный уровень обнаружения при непредсказуемом распределении значений ошибок.

Примером 1–надёжного кода является следующий: $C_1 = \{(y|y^2)\}$, $y \in GF(p^k)$, $p \neq 2$. Коды $C_2 = \{(y|y^3)\}$, $y \in GF(2^k)$ и $C_3 = \{(y|y^{-1})\}$, $y \in GF(2^k)$, k — нечётное число, являются 2–надёжными. В качестве примера систематического 4–надёжного кода можно привести следующий: $C_4 = \{(y|y^{-1})\}$, $y \in GF(2^k)$, k — чётное.

Надёжные коды в силу метода своего построения зачастую обладают небольшим минимальным расстоянием, что не позволяет использовать их для коррекции ошибок. В работе [24] описывается подкласс надёжных кодов с заданным минимальным расстоянием. Надёжный код, у которого $P_{undet}(e) = 0$ для всех ошибок кратности менее d, называется надёжным кодом с минимальным расстоянием d. Через $P_{undet}(e)$ здесь обозначена вероятность пропуска конкретной ошибки e. Такой код, являясь надёжным, не имеет необнаруживаемых ошибок, и при этом обеспечивает 100% обнаружение ошибок малой кратности. Это может быть использовано для проектирования устройств, усиленно защищённых от наиболее частых конфигураций встречаемых ошибок, сохраняя при этом возможность обнаружения любых ошибок. Кроме того, коды с минимальным расстоянием не менее трёх могут быть использованы для коррекции ошибок.

Пример исправления однократной ошибки с помощью надёжных кодов будет приведён в разделе 4.1.2. Более подробно с методами построения кодов с минимальным расстоянием можно в работах [26, 30, 95].

Кроме того, для надёжных кодов был разработан метод исправления повторяющихся ошибок [96]. При повторении одной конфигурации ошибок в течение трёх и более тактов работы вычислительного устройства имеется возможность вычислить значение произошедшей ошибки и правильно декодировать искажённые данные. Для этого необходимо решить систему уравнений в конечных полях. Принцип исправления повторяющейся ошибки с помощью 2–надёжного кода $\{(y|y^3)\}$, $y \in GF(2^k)$, будет продемонстрирован в разделе 4.1.3.

Помимо надёжных кодов, существуют также и частично надёжные коды [24]. Такие коды объединяют линейные и нелинейные преобразования для вычисления проверочной части, что позволяет уменьшить аппаратные затраты на их реализацию. Частично надёжные коды имеют необнаруживаемые ошибки, но их количество меньше, чем у линейных кодов с теми же параметрами.

Принцип использования надёжных кодов в схемах параллельного обнаружения ошибок приведён на рисунке 2.5. Легко заметить, что он аналогичен использованию линейных кодов за тем исключением, что в качестве характеристики выхода оригинального устройства выступают проверочные символы надёжного кода.

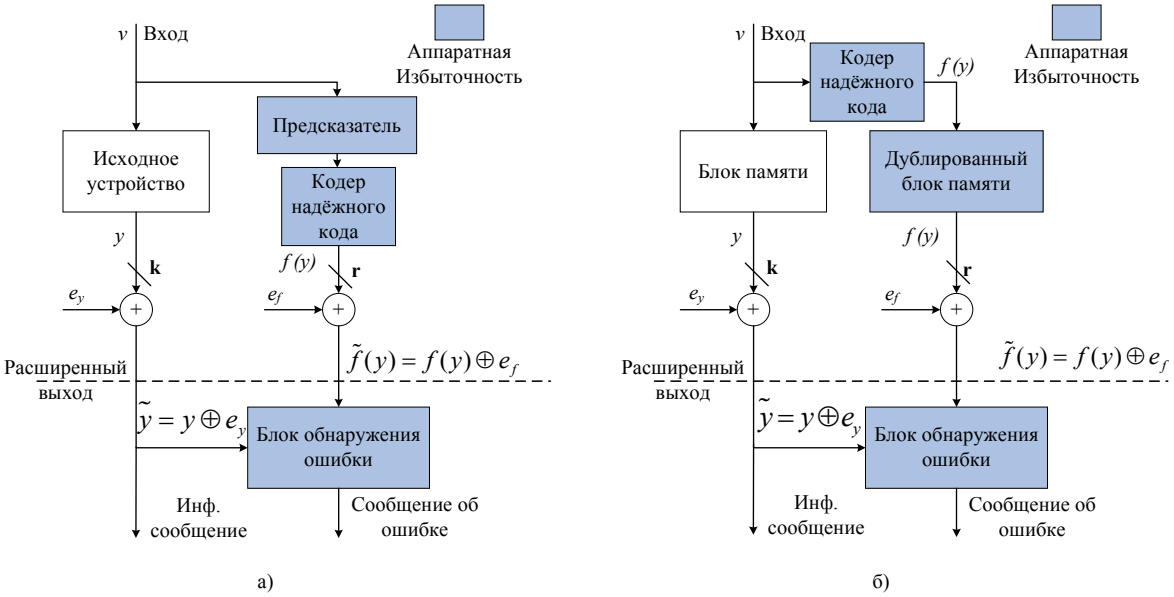


Рисунок 2.5 – Схема использования надёжных кодов для параллельного обнаружения ошибок: а) для защиты линейных блоков вычислительных устройств б) для защиты блоков памяти

Рассматривая надёжные коды относительно слабой модели алгебраических манипуляций, легко заметить, что, в отличие от линейных кодов, они не имеют необнаруживаемых ошибок. Другими словами, не существует такой конфигурации ошибок, которая не будет обнаружена с вероятностью, равной единице. Возникновение любой ошибки останется необнаруженным с вероятностью $P^w_{undet} \leq R/M$. Варьируя значения параметров R и M, можно достигнуть требу-

емого уровня обнаружения слабых манипуляций. Отсюда следует вывод, что надёжные коды, обнаруживающие ошибки, гарантируют заданный уровень защищённости технических систем от искажений, описываемых слабой моделью алгебраических манипуляций.

При этом надёжные коды не обеспечивают защиту от сильной модели манипуляций. Если зависимость между поступившими в систему данными и возникающим искажением приводит к тому, что искажение принимает значение разности между текущим кодовым словом и любым другим, то такое искажение не может быть обнаружено. Следовательно, вероятность пропуска сильной манипуляции составляет $P^s_{undet} = 1$. Таким образом, надёжные коды, обнаруживающие ошибки, неэффективны против искажений, которые описываются сильной моделью алгебраических манипуляций. Из этого следует вывод, что применение кодовых методов на основе надёжных кодов эффективно лишь в тех случаях, когда возникающие в устройстве искажения в точности описываются слабой моделью манипуляций, то есть когда кодируемые сообщения равновероятны и не влияют на появляющиеся ошибки.

Сильно защищённые коды, обнаруживающие алгебраические манипуляции

Другим классом нелинейных кодов, используемым для повышения помехоустойчивости технических систем, являются *сильно защищённые коды, обнаруживающие алгебраические манипуляции* (strongly secure algebraic manipulation detection codes) [19, 25, 27, 83]. Для краткости будем называть их AMD кодами. Данный класс кодов был разработан специально для защиты от искажений, описываемых сильной моделью алгебраических манипуляций.

Очевидно, что для обеспечения защиты от искажений, описываемых сильной моделью манипуляций, необходимо избавиться от детерминированности процедуры кодирования. Привнесение случайности в процесс кодирования может быть осуществлено с помощью некоторой случайной величины, не зависящей от входных данных. Пусть каждому кодируемому сообщению соответствует набор кодовых слов, выбор одного из которых в качестве результата кодирования осуществляется на основании значения некоторой случайной величины. Таким образом, даже при наличии зависимости между входными данными и возникающим искажением устраняется ситуация, когда ошибка может гарантированно принять значение, соответствующее разности между текущим внутренним состоянием системы и любым другим разрешённым состоянием; исчезает зависимость между внутренним состоянием системы и ошибкой. Таким образом, наибольшей вероятностью остаться необнаруженной в такой ситуации обладает такая ошибка, появление которой является необнаруженным при наибольшем количестве значений случайной величины. Рассматривая данную ситуацию относительно искусственной природы помех (например, привнесение помех в вычислительное устройство злоумышленником), получаем, что злоумышленник способен вычислить лишь набор возможных кодовых слов, но не само кодовое слово. В этом случае наиболее эффективной стратегией злоумышленника является привнесение ошибки, успешное внедрение которой осуществится при наибольшем количестве значений случайной величины. Отсюда следует необходимое и достаточное требование к конструкциям AMD кодов:

среди всех возможных комбинаций входных данных и значений возникающих ошибок (перебор всех возможных зависимостей между входными данными и ошибкой), количество значений случайной величины (от которой не зависит ни ошибка, ни входные данные), при которых данная ошибка не будет обнаружена, должно быть меньше общего количества значений этой случайной величины. Необходимо отметить, что используемый принцип отображения сообщения в множество кодовых слов аналогичен принципу, лежащему в основе кодового зашумления [8].

Пусть $y \in GF(2^k)$ есть информационное сообщение, а $x \in GF(2^m)$ есть значение случайной величины. Код $C = \{(y|x|f(x,y)\}$ является AMD кодом, если используемая функция $f(x,y) \in GF(2^r)$ обеспечивает выполнение следующего условия при всех сообщениях y и ошибках $e = (e_y \in GF(2^k)|e_x \in GF(2^m)|e_f \in GF(2^r))$:

$$\max_{y,e\neq 0} \frac{|\{x : S = 0\}|}{|\{x\}|} < 1, \tag{2.3}$$

где синдром S вычисляется как $S = f(x,y) + f(x + e_x, y + e_y) + e_f$.
Другими словами, для AMD кода не найдётся такой комбинации сообщения y и ошибки e, при которой синдром S будет равен нулю (признак отсутствия или пропуска ошибки) вне зависимости от значения случайной величины x. Получаем, что такой код не имеет необнаруживаемых ошибок, каждая ошибка обнаруживается с заданной вероятностью, которая будет приведена ниже. Необходимо подчеркнуть, что подразумевается любая ненулевая ошибка e. Приведённая формулировка определяет AMD код в широком смысле.

AMD кодом в узком смысле называется код, гарантирующий выполнение неравенства (2.3) только при условии, что обязательно была искажена информационная часть кодового слова, то есть имела место ошибка $e = (e_y|e_x|e_f)$ такая, что $e_y \neq 0$ (e_x и e_f могут принимать любые значения). Другими словами, AMD код в узком смысле не имеет необнаруживаемых ошибок только среди подмножества ошибок $e = (0 \neq e_y \in GF(2^k)|e_x \in GF(2^m)|e_f \in GF(2^r))$. Поскольку целью использования данных кодов является обнаружение ошибок именно в информационной части кодового слова, коды в узком смысле также могут быть использованы для защиты технических систем.

Выражение $S(x) = f(x,y) + f(x + e_x, y + e_y) + e_f$, рассматриваемое как уравнение от неизвестной переменной x, называется уравнением маскирования ошибки (error masking equation, УМО).

Легко заметить, что вероятность необнаружения ошибок при использовании AMD кода ограничена сверху следующей величиной:

$$P^s_{undet} \leq P^w_{undet} \leq \max_{y,e\neq 0} \frac{|\{x : S(x) = 0\}|}{|\{x\}|} < 1. \tag{2.4}$$

Принцип использования кодов, обнаруживающих алгебраические манипуляции, в схемах с параллельным обнаружением ошибок продемонстрирован на рисунке 2.6. Их рисунка видно, что по сравнению с линейными и надёжными кодами в схему добавлен генератор псевдослучайных чисел. Генератор псевдослучайных чисел и все провода, по которым передаётся значение случайной величины, должны быть защищены от доступа к их внутреннему состоянию. В

противном случае применение AMD кодов не будет эффективным. При защите блока памяти требуется дополнительное место для хранения значения случайной величины.

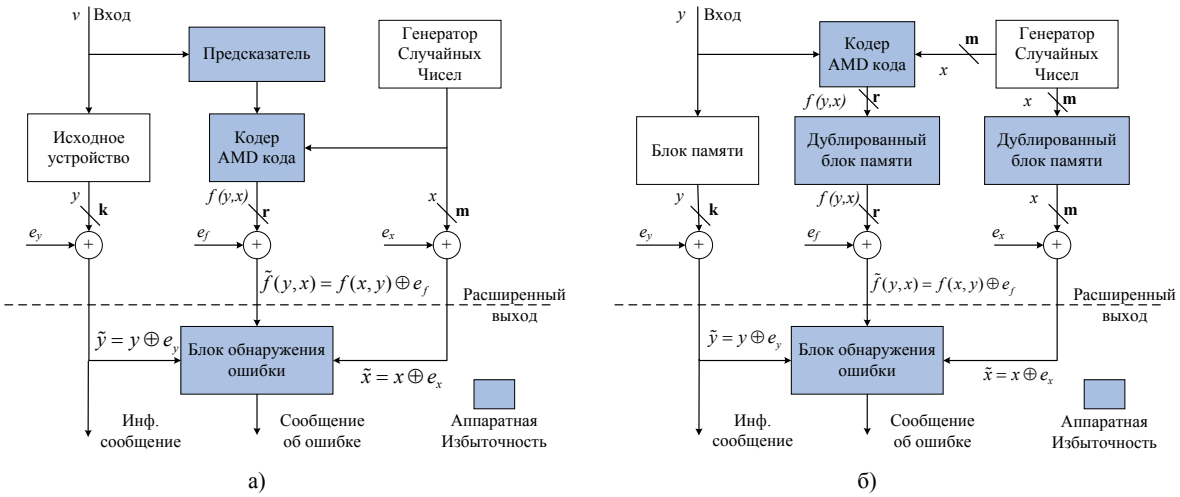


Рисунок 2.6 – Схема использования кодов, обнаруживающих алгебраические манипуляции, для параллельного обнаружения ошибок: а) для защиты линейных блоков вычислительных устройств б) для защиты блоков памяти

Для построения AMD кодов используются различные математические объекты: линейные помехоустойчивые коды, разностные множества, коды аутентификации, полиномы [27].

Рассмотрим пример AMD кода в узком смысле на основе операции умножения в поле [19]. Его кодовые слова выглядят следующим образом:

$$c = (y|x|f(x,y) = x \cdot y),$$

где $y \in GF(2^k)$ есть информационное сообщение;

$x \in GF(2^k)$ есть значение случайной величины;

$f(x,y) \in GF(2^k)$ есть проверочный символ, вычисляемый как произведение x и y в конечном поле $GF(2^k)$.

Использование данного кода позволяет обеспечить вероятность пропуска ошибок, равную $P_{undet} \leq 2^{-k}$. Более подробно про эту конструкцию будет написано в разделе 4.3.1.

Достаточно интересной конструкцией кода в узком смысле является конструкция на основе ЛПК из [27]. Пусть код C представляет собой конкатенацию информационной части $y \in GF(2^k)$, значения случайной величины $x \in \{1, 2, \ldots, 2^k - 1\}$ и проверочной части $f(x,y)$:

$$C = \{(y|x|f(x,y) = [y * G]_x)\},$$

где $G = (1, 2, \ldots, 2^k - 1)$ есть порождающая матрица линейного кода размера $1 \times 2^k - 1$, состоящая из всех ненулевых элементов поля $GF(2^k)$;

$[y * G]_x$ есть x-ый символ вектора $y * G$.

Другими словами, в качестве проверочного символа AMD кода на основании значения случайной величины x выбирается один из проверочных символов линейного кода, задаваемого

матрицей G. Данный AMD код обеспечивает вероятность пропуска ошибок $e = (e_y \neq 0 | e_x | e_f) \in GF(2^{3k})$, равную $P_{undet} = 1/2^k$ [27].

Единственным примером AMD кода в широком смысле является конструкция, основанная на обобщённых кодах Рида–Маллера (РМ), называемая также конструкцией на полиномах [25, 83]. Она хорошо изучена и является эффективным методом обнаружения алгебраических манипуляций; кроме того, для неё разработаны алгоритмы исправления повторяющихся ошибок и ошибок малой кратности [29, 97]. Для большого количества параметров k, m и r данная конструкция является оптимальной в смысле вероятности обнаружения алгебраических манипуляций для AMD кодов в широком смысле (коды лежат на границе (3.1), которая будет приведена в разделе 3.2). Рассмотрим эту конструкцию более подробно.

Известно, что q–ичный код РМ порядка b с t переменными может быть получен подстановкой всех элементов поля $GF(q^t)$ в полиномы от t q–ичных переменных, чья степень не превышает b [14]. Обозначим через $b+2$ степень полинома, используемого в качестве кодирующей функции. Вычислим параметры α и β следующим образом: $b + 2 = \alpha(q-1) + \beta \leq t(q-1)$, $0 \leq \alpha \leq t$, $0 \leq \beta \leq q-2$. Предположим, что при $t = 1$ параметр β является нечётным. Тогда пусть

$$A(x) = \begin{cases} \sum\limits_{i=0}^{t-1} x_i^{b+2}, & \text{если } \alpha = 0,\ \beta - \text{нечётное;} \\ \sum\limits_{i=0}^{t-1} x_0 x_i^{b+1}, t > 1, & \text{если } \alpha = 0,\ \beta - \text{чётное;} \\ \sum\limits_{i=0}^{t-1} x_i^{\beta} \prod\limits_{j=1}^{\alpha} x_{(i+j) \bmod t}^{q-1}, & \text{если } \alpha \neq 0,\ \alpha \neq t; \\ \prod\limits_{i=0}^{\alpha-1} x_i^{q-1}, & \text{если } \alpha = t; \end{cases}$$

где $x_i \in GF(2^r)$.

Величину $B(x, y)$ определим следующим образом:

$$B(x,y) = \sum_{1 \leq j_0 + j_1 + \ldots + j_{t-1} \leq b+1} y_{j_0, j_1, \ldots, j_{t-1}} \prod_{i=0}^{t-1} x_i^{j_i},$$

где $y_{j_0, j_1, \ldots, j_{t-1}}$, $x_i \in GF(2^r)$;

$\prod_{i=0}^{t-1} x_i^{j_i}$ есть моном от $x_0, x_1, \ldots, x_{t-1}$, степень которого между 1 и $b+1$.
При этом на моном накладывается ограничение: $\prod_{i=0}^{t-1} x_i^{j_i} \notin \Delta B(x, y)$, где $\Delta B(x, y)$ определяется как

$$\Delta B(x,y) = \begin{cases} \{x_0^{b+1}, x_1^{b+1}, \ldots, x_{t-1}^{b+1}\}, & \text{если } \alpha = 0,\ b - \text{нечётное;} \\ \{x_1^{b+1}, x_0 x_1^b, \ldots, x_0 x_{t-1}^b, t > 1\}, & \text{если } \alpha = 0,\ b - \text{чётное;} \\ \{x_i^{\beta}, x_{(i+1) \bmod t}^{q-2} \prod\limits_{j=2}^{\alpha} x_{(i+j) \bmod t}^{q-1}, 0 \leq i \leq t-1\}, & \text{если } \alpha \neq 0. \end{cases}$$

Утверждение 2.2 (теорема 4.1 из статьи [83]). *Пусть* $f(x,y) = A(x) + B(x,y)$ *есть* q*–ичный полином с* $y_{j_0,j_1,\ldots,j_{t-1}}$ *в качестве коэффициентов и* $x \in GF(q^t)$ *в качестве* t *переменных, где* $1 \le b \le t(q-1)-2$, $q = 2^r$, *значения* $A(x)$ *и* $B(x,y)$ *приведены выше. Пусть* $b+2 = \alpha(q-1)+\beta$ *и* $b+1 = u(q-1)+v$, $0 \le \alpha,\ u \le t$, $0 \le \beta,\ v \le q-2$. *Предположим, что* $b+2 \ne t(q-1)-1$ *и* b *является нечётным при* $t = 1$. *Тогда код* C, *состоящий из всех векторов* $(y|x|f(x,y))$ *является AMD кодом с* $m = t \cdot r$ *бит, размерностью*

$$k = \left(\sum_{j=0}^{t} (-1)^j \cdot C_t^j \cdot C_{t+b-jq}^{b-jq} - t - 1 \right) r$$

бит и вероятностью необнаружения ошибок

$$P_{undet} \le 1 - (2^r - v)2^{-(u+1)r}.$$

Рассмотрим характеристики кодов, которые могут быть получены при использовании данной конструкции.

Одним из частных случаев является ситуация, когда $r = 1$ бит, $q = 2^r = 2$. Получаем, что $y \in GF(2^k)$, $k = \sum_{i=0}^{b+1} C_t^i - t - 1$ бит, $x \in GF(2^t)$, $f(x,y) \in GF(2)$. Вероятность пропуска ошибок в таком случае составляет $P_{undet} \le 1 - 2^{-(b+1)}$.

Другим частным случаем является ситуация, когда $b \le q-3$. В этом случае размерность кода $k = \left(C_{t+b+1}^t - t - 1\right) r$ бит, а вероятность $P_{undet} \le (b+1)2^{-r}$.

Если $b \le b-3$ есть нечётное число, а $t = 1$, то $f(x,y)$ имеет простой вид:

$$f(x,y) = x^{b+2} + y_0 x + y_1 x^1 + \ldots + y_{b-1} x^b.$$

Данный вид кодирующей функции был представлен ещё в ранних работах по алгебраическим манипуляциям, например, в [19].

По аналогии с надёжными кодами, для AMD кодов на основе полиномов также был разработан метод исправления повторяющихся ошибок за счёт решения системы уравнений в конечных полях [97]. Однако с ростом значений b и t значительно увеличивается объем необходимых для декодирования вычислений. С целью упрощения процедуры коррекции повторяющихся ошибок в статье предлагается выполнять декодирование в два этапа. При этом значение случайной величины x защищается с помощью надёжного кода. На первом этапе декодирования значение случайной величины проверяется на наличие искажений. При их обнаружении используется процедура исправления повторяющихся ошибок с помощью надёжных кодов, описанная в работе [96]. Если коррекция значений случайной величины прошла успешно, то выполняется второй этап декодирования. Он заключается в коррекции повторяющихся ошибок в информационной части слова AMD кода при условии, что значение случайной величины лишено искажений. Для этого необходимо решить систему уравнений в конечных полях. Представленная процедура исправления повторяющихся ошибок обладает высокой вычислительной сложностью, что приводит к значительным аппаратным затратам при её реализации.

Как и надёжные коды, AMD коды на основе кодов РМ зачастую обладают минимальным расстоянием, равным единице. Таким образом, в исходном виде использовать их для исправления ошибок не представляется возможным. В работе [29] представлены две модификации AMD кода на основе обобщённого кода Рида–Маллера, позволяющие увеличить минимальное расстояние кода и, следовательно, исправлять ошибки малой кратности. При этом у кода сохраняется способность обнаружения любых сильных атак. Первая модификация заключается в последовательном кодировании сообщения сначала AMD кодом, а затем кодом Хэмминга. В этом случае минимальное расстояние кода увеличивается до трёх, позволяя исправлять однократную ошибку с помощью классического метода декодирования кода Хэмминга. Вторая модификация также основана на комбинировании линейного и нелинейного кодов, но имеет более сложную структуру. При второй модификации обеспечивается меньшая избыточность, необходимая для увеличения минимального расстояния, однако процедура декодирования требует больше вычислений. Кроме того, при модификациях возможно увеличение минимального расстояния кода до четырёх за счёт добавления общей проверки на чётность. В этом случае код может одновременно исправлять однократные ошибки и обнаруживать двукратные с вероятностью, равной единице, сохраняя вероятность обнаружения остальных помех равной $1 - P_{undet}$. Такие коды, называемые SEC–DED (single error correcting, double error detecting) кодами, обычно используют для защиты блоков памяти.

Согласно принципу построения, данный класс кодов гарантирует заданный уровень защиты от искажений, описываемых сильной моделью алгебраических манипуляций. Поскольку сильная модель является обобщением слабой модели, коды, обнаруживающие алгебраические манипуляции, также обеспечивают защищённость технических систем и от слабых манипуляций. Таким образом, нелинейные кодовые методы на основе AMD кодов позволяют повышать помехоустойчивость вычислительных устройств в условиях дискретного канала с алгебраическими манипуляциями вне зависимости от их типа.

2.2 Выводы

Среди рассмотренных методов повышения помехоустойчивости в условиях искажений, описываемых моделью алгебраический манипуляций, выделяются методы на основе нелинейных кодов. Они позволяют обеспечивать целостность данных с заданной вероятностью в рассматриваемой модели канала, повышая надёжность и качество функционирования технических систем. Таким образом, описанные методы нелинейного кодирования являются достаточно универсальными методами повышения помехоустойчивости для множества каналов со случайной структурой, которые могут быть описаны моделью дискретного канала с алгебраическими манипуляциями.

Однако, существующие коды, обнаруживающие алгебраические манипуляции, (надёжные и AMD) требуют тщательного анализа их характеристик. Не исключено, что могут быть найдены более эффективные методы их построения и алгоритмы декодирования. В следующей главе

будут представлены две границы параметров нелинейных кодов, полученные автором диссертации с помощью теоретико–информационного анализа методов их построения. В четвёртой главе будут предложены как новые методы повышения помехоустойчивости на основе разработанных автором нелинейных кодов, так и модификации существующих. В заключительной — пятой — главе диссертационной работы будут приведены научно–технические предложения по применению предлагаемых нелинейных кодов.

3 Границы на параметры нелинейных кодов

В данной главе диссертационной работы представлены границы, позволяющие оценить экстремальные значения параметров надёжных кодов и сильно защищённых кодов, обнаруживающих алгебраические манипуляции. Данные границы позволяют оценивать оптимальность существующих методов построения кодов, задавая направления для дальнейших исследований.

3.1 Граница длины систематического *R*-равномерно надёжного кода

Исследуем минимальную длину (обозначим ее через n) R–равномерно надёжного кода [1]. Напомним, что R–равномерно надёжным кодом называется код C, у которого

$$R = ||\{c : c \in C, c + e \in C\}|| = const$$

для любых ненулевых значений ошибки e. Пусть k — размерность кода, то есть $M = ||C|| = p^k$. В силу равномерной надёжности кода различных разностей между кодовыми словами ровно $M(M-1)/R$. Каждая из этих разностей является элементом поля $GF(p^n)$, над которым построен код. Следовательно, поле должно содержать не менее $M(M-1)/R$ элементов. Кроме того, для систематического кода p^{n-k} элементов поля не могут быть разностями между кодовыми словами, потому что разность систематических частей кодовых слов не может равняться $0 \in GF(p^k)$. Для такого кода только $p^n - p^{n-k}$ элементов поля могут являться разностями кодовых слов, и их должно быть не менее $M(M-1)/R$. Из этого утверждения можно получить нижнюю границу длины систематического R–равномерно надёжного кода:

$$\begin{aligned}
p^n - p^{n-k} &\geq \frac{M(M-1)}{R}, \\
p^n(1 - p^{-k}) &\geq \frac{M(M-1)}{R}, \\
p^n \left(\frac{M-1}{M}\right) &\geq \frac{M(M-1)}{R}, \\
p^n &\geq \frac{M^2}{R}, \\
n &\geq \log_p \frac{M^2}{R}.
\end{aligned}$$

На практике наиболее востребованы параметры $p = 2$ и $R = 2$. Легко заметить, что граница при этом принимает следующий вид:

$$n \geq 2k - 1.$$

В работе [24] были предложены конструктивные методы построения кодов над $GF(2^n)$ с $R = 2$ для любых k (например, $\{(y|y^3)\}$ и $\{(y|y^{-1})\}$ из раздела 2.1.4). Рассмотрим поле $GF(2^2)$, порождённое полиномом $x^2 + x + 1$. Здесь и далее элементы поля будем записывать в двоичном представлении, если не указано другое. Для $k = 2$ согласно конструкции $\{(y|y^{-1})\}$ может быть построен код

$$\{(00|00), (01|01), (10|11), (11|10)\} \in GF(2^4),$$

где символом «|» разделены информационные и проверочные символы, соответственно.
Данные методы построения кодов обеспечивают скорость 1/2, то есть $n = 2k$. Легко заметить, что данные кодовые конструкции не лежат на выведенной границе.

Можно показать, что существуют коды, лежащие на границе. При $k = 2$ ($n \geq 2k - 1 = 3$) примером такого кода является следующий:

$$C = \{(00|0), (01|1), (10|1), (11|1)\} \in GF(2^3).$$

Для кода C выпишем все разности между кодовыми словами:

$$\begin{aligned}
(00|0) - (01|1) &= (01|1),\\
(01|1) - (00|0) &= (01|1),\\
(00|0) - (10|1) &= (10|1),\\
(10|1) - (00|0) &= (10|1),\\
(00|0) - (11|1) &= (11|1),\\
(11|1) - (00|0) &= (11|1),\\
(01|1) - (10|1) &= (11|0),\\
(10|1) - (01|1) &= (11|0),\\
(01|1) - (11|1) &= (10|0),\\
(11|1) - (01|1) &= (10|0),\\
(10|1) - (11|1) &= (01|0),\\
(11|1) - (10|1) &= (01|0).
\end{aligned}$$

Легко заметить, что одинаковые значения разностей среди всех возможных пар слов данного кода встречаются ровно два раза. Отсюда следует, что при возникновении некоторой ошибки e не более двух слов кода C исказятся в кодовые слова. Пусть привносится ошибка $e = (100)$.

Получаем

$$\begin{aligned}(00|0)+(10|0) &= (10|0),\\ (01|1)+(10|0) &= \textbf{(11|1)},\\ (10|1)+(10|0) &= (00|1),\\ (11|1)+(10|0) &= \textbf{(01|1)},\end{aligned}$$

где жирным шрифтом выделены разрешённые кодовые комбинации. Таким образом, если по каналу передавались слова $(00|0)$ или $(10|1)$, то появление ошибки $e = (100)$ будет обнаружено. При передаче слов $(01|1)$ и $(11|1)$ появление ошибки не может быть обнаружено.

Данный код C являются равномерно надёжным систематическим кодом с $R = 2$ и $P^w_{undet} = 1 - 2/4 = 0.5$, являясь при этом кодом с минимальной возможной длиной (информационная избыточность меньше, чем у существующих кодов из [24] на 1 бит при сохранении той же вероятности необнаружения ошибки P^w_{undet}).

Представленная граница может быть использована для оценки надёжных кодов с точки зрения оптимальности длины кода. Условие применимости данной границы следующие: параметр R должен делить величину $M(M-1)$ нацело и быть меньше мощности кода M. Легко заметить, что данная граница также может быть применена и к неравномерно надёжным кодам. В этом случае происходит усиление неравенства:

$$n > \log_p \left(M^2/R\right).$$

Расхождения между представленной границей и существующими надёжными кодами показывают, что необходимы дальнейшие исследования на предмет существования более эффективных кодов и конструктивных методов их построения.

3.2 Нижняя граница обнаруживающей способности AMD кода на базе кодов Рида–Маллера

Конструкция AMD кодов на базе обобщённых кодов Рида–Маллера (РМ) является наиболее распространённой. Кодирующая функция $f(x, y)$ выбирается таким образом, чтобы уравнение маскирования ошибки (УМО) соответствовало кодовому слову обобщённого кода РМ.

Для вероятности необнаружения ошибки P_{undet} некоторых конструкций AMD кодов, включая рассматриваемую, известна следующая нижняя граница. Для любого q–ичного (k, m, r) AMD кода (k, m, r есть количество информационных, случайных и проверочных битов соответственно, $q = 2^r$) выполняется следующее неравенство:

$$P_{undet} \geq 1 - \frac{d_q(2^m, M)}{2^m}, \tag{3.1}$$

где $d_q(2^m, M)$ — максимальное возможное расстояние Хэмминга q-ичного кода длины 2^m с $M = 2^{k+m+r}$ кодовыми словами [25].

Данную границу можно уточнить для всех кодов, построенных на основе обобщённых кодов РМ [7]. Определим величину $t = m/r$, при условии, что r делит m нацело. Представим значение случайной величины $x \in GF(2^m)$ как вектор из t переменных $x_i \in GF(2^r)$, $i = 1, \ldots, t$, то есть $x = (x_1|x_2|\ldots|x_t)$. Известно, что q–ичный код РМ порядка b с t переменными может быть получен подстановкой всех элементов поля $GF(q^t)$ в полиномы от t q–ичных переменных, чья степень не превышает b [14]. Пусть $p_i(z)$ есть один из таких полиномов, $z \in GF(q^t)$. Тогда соответствующее ему кодовое слово обобщённого кода Рида–Маллера можно представить как

$$c_i = \left(p_i(0)|p_i(\alpha^0)|p_i(\alpha^1)|\ldots|p_i(\alpha^{(q^t-2)})\right),$$

где α есть примитивный элемент поля $GF(q^t)$ [5]. Также известно, что минимальный вес кодового слова (минимальное количество ненулевых элементов) кода порядка b с t переменными равен

$$d_{GRM}(b,t) = (q-v)q^{t-u-1},$$

где $b = (q-1)u+v$. Легко заметить, что количество нулевых компонент в кодовом слове соответствует количеству корней порождающего его полинома $p_i(z)$. Из этого следует, что количество корней полинома степени b от t переменных ограничено сверху следующим значением

$$N_{b,t} \leq q^t - (q-v)q^{t-u-1}.$$

Из принципа построения AMD кодов на базе кодов Рида–Маллера следует, что при

$$k \leq \left(\sum_{j=0}^{t}(-1)^t \cdot C_t^j \cdot C_{t+s-jq}^{s-jq} - t - 1\right) r$$

может быть построен код, степень УМО которого не будет превышать значения s [25]. Известно, что уравнение маскирования ошибки такого кода представляет собой, в общем случае, полином степени b от t q–ичных переменных. При проверке данных на наличие искажений в УМО подставляется значение случайной величины x так, что её компоненты x_i используются как значения t переменных уравнения. Если вычисленное таким образом значение УМО равно нулю, то искажение не выявляется.

Допустим, что существует некий наилучший в смысле обнаруживающей способности (k, m, r) AMD код на базе кодов РМ с таким же ограничением на k, но отличающийся от кода, построенного согласно конструкции из [25]. Рассмотрим его уравнение маскирования ошибки. Различным комбинациям сообщения y и ошибки e, в общем случае, соответствуют УМО различных степеней. Обозначим через $\breve{s}$ наибольшую возможную степень УМО данного наилучшего кода. Из того условия, что наш код обладает минимальной возможной вероятностью необнаружения ошибки, следует, что максимальное количество корней его УМО не больше, чем у существующей конструкции из [25]. Так как количество корней полинома пропорционально его степени, то $\breve{s} \leq s$. Количество корней УМО для рассматриваемого наилучшего кода ограничена следующим образом:

$$N_{\breve{s},t} \leq q^t - (q-\breve{v})q^{t-\breve{u}-1},$$

где $\breve{s} = (q-1)\breve{u} + \breve{v}$.

Отсюда получаем границу на вероятность маскировки ошибки наилучшего кода:

$$P_{undet} \leq \frac{N_{\breve{s},t}}{q^t} \leq \frac{q^t - (q-\breve{v})q^{t-\breve{u}-1}}{q^t} = 1 - (q-\breve{v})q^{-\breve{u}-1},$$

$$P_{undet} \leq 1 - (q-\breve{v})q^{-\breve{u}-1}.$$

При этом, рассматриваемый наилучший код должен удовлетворять границе (3.1). Получаем, что

$$\begin{cases} P_{undet} \geq 1 - \dfrac{d_q(2^m, M)}{2^m} \text{ (теоретическая граница)}, \\ P_{undet} \leq 1 - (q-\breve{v})q^{-\breve{u}-1} \text{ (для наилучшего существующего кода)}. \end{cases}$$

Получаем нижнюю границу минимальной возможной степени УМО — это такая наименьшая степень $\breve{s}$, при которой выполняется неравенство:

$$1 - (q-\breve{v})q^{-\breve{u}-1} \geq 1 - \frac{d_q(2^m, M)}{2^m},$$

$$(q-\breve{v})q^{-\breve{u}-1} \leq \frac{d_q(2^m, M)}{2^m},$$

где по–прежнему $\breve{s} = (q-1)\breve{u} + \breve{v}$.

Таким образом, уточнённая нижняя граница для AMD кодов на базе кодов Рида–Маллера может быть записана следующим образом:

$$\begin{cases} P_{undet} \geq 1 - (q-\breve{v})q^{-\breve{u}-1}, \\ \breve{s} = \min s_i : (q-v_i)q^{-u_i-1} \leq \dfrac{d_q(2^m, M)}{2^m}, \\ s_i = (q-1)u_i + v_i, s_i = 1, 2, \ldots. \end{cases} \tag{3.2}$$

Полученная нижняя граница во многих случаях позволяет уточнить старую, менее точную границу. Она была использована для анализа оптимальности существующих кодов с точки зрения обнаруживающей способности. AMD коды на базе кодов РМ, степень УМО которых равна $\breve{s}$ из формулы (3.2), обладают наилучшей обнаруживающей способностью, которая может быть достигнута с помощью данного метода построения кодов.

Рассмотрим пример. Пусть необходимо построить AMD код на основе кодов РМ с параметрами $k = 4$ бита, $m = 4$ бита, $r = 2$ бита. Применение границы Синглтона [14] к границе (3.1) показывает, что вероятность необнаружения ошибки не может быть менее, чем

$$P_{undet} \geq \left\lceil \frac{k+m}{r} \right\rceil 2^{-m} = \left\lceil \frac{4+4}{2} \right\rceil 2^{-4} = 0.25.$$

Из таблицы лучших линейных кодов[1)] следует, что наибольшее достижимое расстояние четверичного кода длины $2^m = 16$ с 5 информационными символами ($2^{k+m+r} = 2^{10} = 4^5$) равно 9. Таким образом, из границы (3.1) получаем, что

$$P_{undet} \geq 1 - \frac{d_4(n = 2^4, M = 4^5) = 9}{2^4} = 1 - \frac{9}{16} = 0.4375.$$

[1)] http://www.iks.kit.edu/home/grassl/codetables/

Применение выведенной границы (3.2) показывает, что для наилучшего возможного кода на основе кодов РМ $\breve{s} = 2$. Следовательно, для AMD кодов на основе кодов РМ справедлива следующая нижняя оценка вероятности невыявления ошибки:

$$P_{undet} \geq 1 - (q - \breve{v})q^{-\breve{u}-1} = 1 - (4-2)4^{-0-1} = 0.5.$$

В частности, это значит, что максимальное расстояние кода РМ, на основе которого может быть построен AMD с приведёнными параметрами (k, m, r), равно 8.
Код на основе кодов РМ, достигающий эту границу при параметрах $(k = 4, m = 4, r = 2)$, существует и имеет кодирующую функцию $f(x, y) = x_0y_0 + x_1y_1 + x_0^3 + x_1^3$, $x_i, y_i \in GF(2^2)$.

Таким образом, вероятность необнаружения ошибок AMD кодов на основе кодов РМ в некоторых случаях выше теоретической границы для AMD кодов. Из этого следует необходимость поиска новых конструкций кодов с более высокой обнаруживающей способностью, а также дальнейшее исследование их экстремальных характеристик.

3.3 Выводы

Представленные в третьей главе границы на параметры нелинейных кодов показали, что существует расхождение между теоретически и практически достижимыми параметрами кодов. Это говорит о необходимости дальнейшего изучения конструкций нелинейных кодов с целью поиска совершенных и оптимальных.

4 Новые нелинейные кодовые методы повышения помехоустойчивости

В данной главе диссертационной работы приведены новые кодовые методы повышения помехоустойчивости на основе нелинейных кодов, методы их декодирования, а также модификации существующих кодов.

Данные результаты представляют интерес потому, что позволяют уменьшить вычислительную сложность процедур кодирования и декодирования, а также увеличивают вероятность обнаружения алгебраических манипуляций.

4.1 Обобщение надёжных кодов

4.1.1 Конструкция обобщённых систематических надёжных кодов

В данном разделе предлагается кодовый метод повышения помехоустойчивости на основе новой конструкции AMD кодов, основанной на надёжных кодах, обнаруживающих ошибки. Конструкция получила название обобщённого систематического надёжного (ОСН) кода [9].

Как уже было сказано, недостатком надёжных кодов при условии сильной модели манипуляции является детерминированность вычисляемой проверочной части. Наиболее естественным методом её устранения представляется привнесение случайности в кодируемое сообщение. К сообщению добавляется значение некоторой случайной величины, формируя дополненное сообщение. Таким образом, уменьшая фактическую размерность кода (при фиксированных параметрах кода), удаётся сопоставить сообщению множество дополненных сообщений, размер которого равен количеству возможных значений случайной величины. Далее, дополненное сообщение подвергается процедуре кодирования надёжным кодом. Кодирование представляет собой однозначное и обратимое отображение дополненного сообщения в кодовое слово. Описанная процедура приводит к неопределённости результата кодирования, когда каждому исходному сообщению соответствует набор дополненных сообщений, и, следовательно, возможных кодовых слов. За счёт этого устраняется уязвимость к сильным манипуляциям, характерная для надёжных кодов. Данная конструкция является обобщением надёжных кодов на случай искажений, описываемых моделью сильных манипуляций.

Рассмотрим следующий метод построения кода на основе перестановки в конечном поле. Пусть вектор $z \in GF(2^{k+m=r})$ представляет собой конкатенацию информационного сообщения

$y \in GF(2^k)$ и значения случайной величины $x \in GF(2^m)$, то есть является модифицированным сообщением $z = (y|x)$. Для кодирования используется R–надёжный систематический код $C = \{(z|f(z))\}$, где $f(z) : GF(2^r) \to GF(2^r)$ есть некоторая нелинейная функция с показателем нелинейности P_f. Таким образом, кодовое слово рассматриваемого кода можно записать как

$$c = (z|f(z)) = ((y|x)|f(y|x)),$$

а параметры кода имеют следующий вид: размерность кода k, длина $n = 2(k+m) = 2r$, скорость $k/2r$.

Утверждение 4.1. *ОСН код на основе функции перестановки в поле Галуа при $R < 2^m$ является сильно защищённым AMD кодом в широком смысле с вероятностью необнаружения ошибки $P^s_{undet} \leq R/2^m$.*

Доказательство. Как уже было показано, обнаруживающая способность AMD кода напрямую зависит от максимального количества корней его уравнения маскирования ошибки. УМО от переменной z рассматриваемого кода выглядит следующим образом:

$$f(z) + e_f - f(z + e_z) = 0,$$

$$f(z) + e_f = f(z + e_z).$$

Проанализируем количество корней уравнения маскирования ошибки. Из теории нелинейных функций над конечными полями известно, что данное уравнение имеет не более $R = P_f \cdot 2^r$ решений относительно z. То есть для любой конфигурации ошибок $e = (e_z, e_f)$ существует не более R значений z, при которых выполнится равенство в УМО.

Так как вектор z представляет собой конкатенацию векторов y и x, существует не более R их комбинаций, при которых ошибка не будет обнаружена. Следовательно, для каждой конфигурации ошибок e существуют не более R значений x, при которых алгебраическая манипуляция не будет обнаружена. Таким образом, если параметр R надёжного кода, на основе которого строится ОСН код, меньше чем 2^m, то, согласно определению, ОСН код является AMD кодом с вероятностью необнаружения искажений, соответствующих модели сильной алгебраической манипуляции, равной $P^s_{undet} \leq R/2^m$. ■

В качестве кодирующей функции удобно использовать разностно δ-равномерные отображения, введённые К. Найберг в работе [89]. В качестве параметра δ указывается количество решений уравнения $f(z + a) - f(z) = b$ среди всех $0 \neq a \in GF(2^r), b \in GF(2^r)$, то есть параметр R согласно используемым обозначениям. Для совершенных нелинейных функций δ равен единице, для почти совершенных нелинейных функций — двум. Таким образом, при использовании разностно δ-равномерного отображения в качестве кодирующей функции ОСН кода получаем, что вероятность необнаружения сильной манипуляции составляет

$$P^s_{undet} \leq P_f \cdot 2^k = \delta/2^m.$$

Рассмотрим пример ОСН кода на основе перестановки в конечном поле: пусть $k = 4$ бита, $m = 2$ бита, тогда $r = k + m = 6$ бит. В качестве функции кодирования выберем возведение в третью степень в поле Галуа: $f(z) = f(x, y) = (x|y)^3$, являющуюся почти совершенной нелинейной функцией с $\delta = 2$. Таким образом, за основу берётся 2-надёжный код $\{(z|f(z) = z^3)\}$. Полученный ОСН код будет AMD кодом ($\delta < 2^m$) со скоростью $k/(k + m + r) = 1/3$, гарантируя при этом вероятность необнаружения любой сильной манипуляции, равную $P^s_{undet} \leq 0.5$. Продемонстрируем принцип кодирования данным ОСН кодом. Пусть информационное сообщение $y \in GF(2^4)$ приняло значение (0111). На рисунке 4.1 представлены все соответствующие ему дополненные сообщения и кодовые слова. Информационная, случайная и проверочная части кодового слова разделены символом «|», соответственно. Вычисления проверочной части выполнены в поле Галуа $GF(2^6)$ с порождающим полиномом $x^6 + x + 1$.

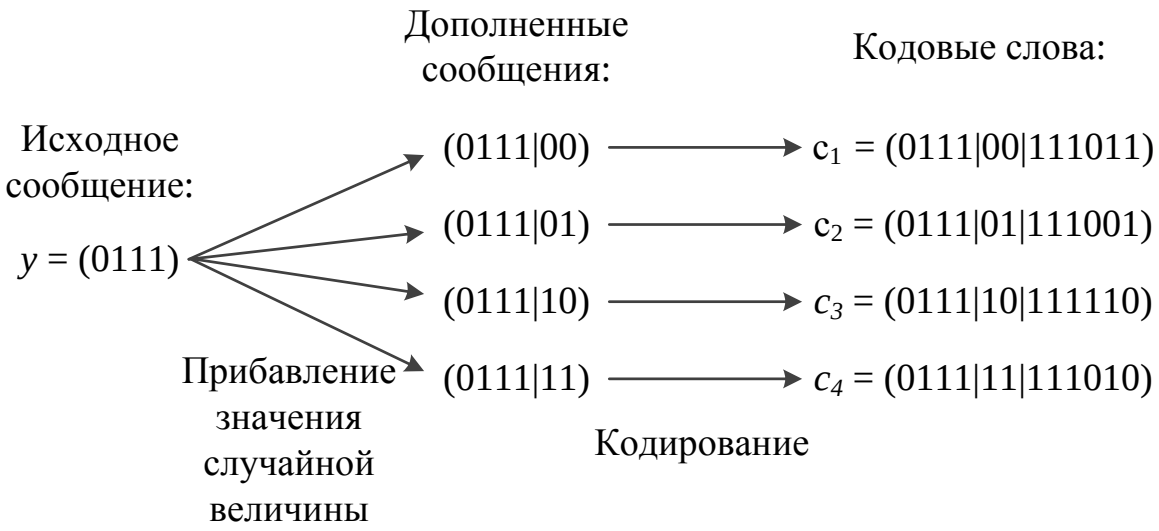


Рисунок 4.1 – Принцип кодирования ОСН кодом с параметрами $k = 4, m = 2, r = 6$ на примере сообщения (0111)

Таким образом, в качестве результата кодирования сообщения $y = (0111)$ выбирается одно из четырёх слов $\{(0111|00|111011), (0111|01|111001), (0111|10|111110), (0111|11|111010)\}$ в зависимости от значения случайной величины $x \in GF(2^2)$.

Утверждение 4.2. *При искажениях, описываемых слабой моделью алгебраических манипуляций, ОСН код на основе перестановки в конечном поле обеспечивает вероятность необнаружения ошибки, равную*

$$P^w_{undet} \leq R/2^r,$$

что в 2^k раз меньше, чем в случае сильной манипуляции.

Доказательство. Это утверждение следует из свойств R–надёжных кодов [24]. Пусть произошла некоторая ошибку e. Среди всех $2^{k+m} = 2^r$ кодовых слов только R будут искажены ошибкой e в другие кодовые слова. Предполагая равномерное распределение кодовых слов, получаем, что

вероятность необнаружения любой ошибки e, описываемой моделью слабой манипуляции, ограничена величиной $P^w_{undet} \leq R/2^r$. ■

Равенство в последнем выражении достигается при использовании R–равномерно надёжного кода (то есть кодирующая функция является δ–равномерным отображением).

Таким образом, рассмотренный в прошлом примере код обеспечивает вероятность пропуска слабой манипуляции $P^w_{undet} = 2^{-6+1} \approx 3 \cdot 10^{-2}$.

Легко заметить, что данная конструкция схожа с кодами, используемыми для кодового зашумления [8]. Фактически, случайная величина в обоих случаях используется для маскирования кодового слова, соответствующего исходному сообщению.

При работе в полях характеристики два для построения ОСН кодов целесообразно использовать 2–надёжные коды $\{(z|z^{-1})\}$ и $\{(z|z^3)\}$, использующие почти совершенные нелинейные функции. Они обеспечивают вероятности $P^s_{undet} \leq 2/2^m = 2^{-m+1}$ и $P^w_{undet} = 2/2^r = 2^{-r+1}$. В случае, если характеристика поля отличается от двух, то рекомендуется использовать 1–надёжный код $\{(z|z^2)\}$, кодирующая функция которого является совершенной нелинейной функцией. В этом случае достигаются вероятности $P^s_{undet} = 1/2^m = 2^{-m}$ и $P^w_{undet} = 1/2^r = 2^{-r}$.

Как уже было сказано, применение границы Синглтона к границе для AMD кодов (3.1) приводит к следующему неравенству:

$$P^s_{undet} \geq \left\lceil \frac{k+m}{r} \right\rceil 2^{-m}.$$

Для ОСН кодов на основе перестановки в поле $k + m = r$, следовательно, граница принимает следующий вид:

$$P^s_{undet} \geq 2^{-m}.$$

Данная граница достигается ОСН кодами при работе в полях характеристики, отличной от двух. Например, на данной границе лежит ОСН код на основе 1–надёжного кода $\{(z|z^2)\}$, $z \in GF(p^n)$, $p \neq 2$. Наименьшая вероятность необнаружения ошибок, достигаемая ОСН кодами в полях характеристики два, отличается от граничного значения в два раза и составляет $P^s_{undet} = 2^{-m+1}$.

Заслуживающим внимания надёжным кодом, на основе которого могут быть построены ОСН коды, является код, основанный на экспоненциальной функции [2]. Данный код позволяет использовать существующие в большинстве криптографических приложений блоки экспоненцирования, что приводит к значительном уменьшению аппаратных затрат. При этом возможно использование кода с большой длиной, что обеспечивает высокую вероятность обнаружения искажений за счёт добавления значительной доли случайных символов. Более подробно данный надёжный код описан в разделе 4.2.

Кроме того, возможен и другой метод построения ОСН кода — на основе операции скалярного умножения в конечном поле, а не перестановки. Рассмотрим метод его построения. Его кодовые слова также состоят из трёх компонентов:

$$c = (y \in GF(2^{ta}), x \in GF(2^{tb}), f(x, y) \in GF(2^r)),$$

но проверочный символ имеет размер $r = a + b$ бит, $y = (y_1, y_2, \ldots, y_t)$, $y_i \in GF(2^a)$, $x = (x_1, x_2, \ldots, x_t)$, $x_i \in GF(2^b)$, а функция кодирования выглядит следующим образом:

$$f(x, y) = \sum_{i=1,3,\ldots}^{t} (y_i, x_i) \cdot (y_{i+1}, x_{i+1}).$$

То есть в качестве слагаемых в выражении скалярного умножения используется конкатенация компонентов информационного и случайного векторов $(y_i, x_i) \in GF(2^r)$.

Утверждение 4.3. *ОСН код на основе операции скалярного умножения в поле Галуа является сильно защищённым AMD кодом в широком смысле с вероятностью необнаружения сильных манипуляций* $P^s_{undet} = 2^{-b}$.

Доказательство. Обозначим каждый элемент (y_i, x_i) через z_i, а (e_{y_i}, e_{x_i}) через e_{z_i}. Тогда кодирующая функция является линейным полиномом:

$$f(z) = z_1 z - 2 + z_3 z_4 + \ldots + z_{t-1} z_t = \sum_{i=1,3,\ldots}^{t-1} z_i z_{i+1}.$$

С помощью простых манипуляций получаем следующее УМО:

$$\sum_{i=1,3,\ldots}^{t-1} z_i e_{z_{i+1}} + z_{i+1} e_{z_i} = \sum_{i=1,3,\ldots}^{t-1} e_{z_i} e_{z_{i+1}} + e_f. \tag{4.1}$$

Таким образом, мы имеем линейный полином с t q–ичными переменными z_i, где $q = 2^{r=a+b}$.

Единственный вариант, при котором исключаются все z_i (и, соответственно, все x_i) из уравнения (4.1). это возникновение ошибки, у которой все $e_{z_i} = 0$. Тогда УМО не будет зависеть от значения случайной величины x:

$$S = {}_f.$$

Но в этом случае e_f должно также равняться нулю (чтобы синдром S был равен нулю), что приводит к нулевому вектору ошибки e. Из этого следует, что не существует ненулевых векторов ошибки $e \in GF(2^{ta+tb+a+b})$, которые были бы необнаруживаемы при любых значениях z.

Легко заметить, что уравнение маскирования ошибки от t q–ичных переменных (4.1) имеет $q^{w-1} = 2^{r(w-1)}$ решений относительно z, где $1 \leq w \leq t$ есть количество ненулевых компонент e_{z_i}. Следовательно, существует $2^{r(w-1)}$ комбинаций векторов (y_i, x_i), $i = 1, \ldots, t$, при которых УМО выполняется (и ошибка не обнаруживается). Но при сильной модели алгебраических манипуляций информационная часть y и ошибка e фиксированные, следовательно, каждый $z_i = (y_i, x_i)$ принимает только 2^b значений вместо $2^{r=a+b}$. Следовательно, максимальное количество решений УМО (4.1) относительно x ограничено сверху величиной $2^{b(w-1)}$. По определению (2.4) данный ОСН код на основе операции скалярного умножения является AMD кодом в широком смысле с вероятностью необнаружения ошибки:

$$P^s_{undet} \leq \max_{y, e \neq 0} \frac{|\{x : S(x) = 0\}|}{|\{x\}|} = \frac{2^{b(w-1)}}{2^{bw}} = 2^{-b}.$$

■

Легко заметить, что по аналогии с ОСН кодами на основе перестановки в поле, данный код также обеспечивает более низкую вероятность необнаружения слабых манипуляций.

Утверждение 4.4. *В канале со слабыми алгебраическими манипуляциями данный ОСН код на основе скалярного умножения обеспечивает вероятностью необнаружения слабых манипуляций, равную* $P^{w}_{undet} = 2^{-r=-(a+b)}$.

Доказательство. Доказательство аналогично предыдущему, за исключением того, что при слабой модели манипуляций информационное сообщение y не фиксировано. Следовательно, УМО имеет $2^{r(w-1)}$ решений относительно z. В соответствии с определением (2.2) получаем следующее выражение для вероятности необнаружения ошибок:

$$P^{W}_{undet} \leq \max_{e \neq 0} \frac{|\{(y,x) : S(x) = 0\}|}{|\{x\}| * |\{x\}|} = \frac{2^{r(w-1)}}{2^{rw}} = 2^{-r}.$$

■

Рассмотрим пример ОСН кода на основе операции скалярного умножения в конечном поле. Пусть $y \in GF(2^{48})$ есть информационное сообщение, а $r = 13$ бит. Тогда $y = (y_1, y_2, y_3, y_4, y_5, y_6)$, $y_i \in GF(2^8)$, $t = 6$. Получаем, что $x \in GF(2^{t(r-a)=30})$ и $x = (x_1, x_2, x_3, x_4, x_5, x_6)$, $x_i \in GF(2^{b=5})$. Следующий AMD код в широком смысле может быть построен:

$$C = \{(y \in GF(2^{48}), x \in GF(2^{30}), f(x,y) \in GF(2^{13}))\}$$

с использование кодирующей функции

$$f(x,y) = (y_1, x_1)(y_2, x_2) + (y_3, x_3)(y_4, x_4) + (y_5, x_5)(y_6, x_6).$$

Скорость кода близка к 0.53, вероятности обнаружения сильных и слабых алгебраических манипуляций равны $P^{s}_{undet} = 2^{-5} \approx 3 * 10^{-3}$ и $P^{s}_{undet} = 2^{-13} \approx 10^{-4}$, соответственно.

4.1.2 Исправление ошибок малой кратности

Увеличение минимального расстояния ОСН кода

Как и большинство AMD кодов, ОСН коды обладают небольшим минимальным кодовым расстоянием (у многих кодов оно равно единице). Однако для некоторых применений кодов (например, для проектирования помехоустойчивой памяти) зачастую требуется возможность исправления ошибок малой кратности, вызываемых естественными причинами (классическими каналами с шумом). Природа этих ошибок заключается в несовершенстве физических характеристик каналов передачи и хранения данных, однако зачастую сводится к влиянию только аддитивной помехи. Для проектирования помехоустойчивой памяти наиболее востребованы коды, исправляющие однократную ошибку (single error correcting code, SEC код), и коды, исправляющие однократную и обнаруживающие двукратную ошибки (single error correcting, double error detecting code, SEC–DED код). Известно, что для исправления однократной ошибки необходимо

минимальное кодовое расстояние, равное трём. Для того, чтобы код мог также гарантированно обнаруживать двукратную ошибку, требуется расстояние, равное четырём [14].

Следовательно, для применения ОСН кодов в задачах, требующих исправления ошибок малой кратности, необходимо модифицировать их конструкцию с целью увеличения минимального кодового расстояния. Наиболее естественным решением этой задачи является использование схемы последовательного кодирования сообщения сначала AMD кодом, а потом линейным (как предлагается в [29]). Однако, к ОСН кодам на основе перестановки в поле применима также и несколько изменённая схема кодирования, когда не всё слово ОСН кода кодируется линейным кодом, а отдельные его компоненты [11]. При этом в процессе декодирования используются дистанционные свойства как линейного кода, так и ОСН кода. Это позволяет снизить необходимую для исправления ошибок информационную избыточность. Кроме того, упростить процедуру декодирования по сравнению с [29] позволяет тот факт, что кодирующая функция является несжимающей и, в некоторых случаях, обратимой.

Рассмотрим пример модификации ОСН кода на основе перестановки в поле $\{(y|x|f(x,y) = (y|x)^{-1})\}$, $y \in GF(2^k)$, $x \in GF(2^m)$, $f(x,y) \in GF(2^{k+m})$ с целью увеличения его минимального расстояния. Напомним, что взятый за основу надёжный код $\{(z|z^{-1})\}$ при нечётном k является 2–надёжным, при чётном $k-4$–надёжным.

Исследуем дистанционные свойства кода $\{(y|x|f(x,y) = (y|x)^{-1})\}$. Кодовые слова в силу систематичности представляют собой конкатенацию информационной и проверочной частей, а также значения случайной величины. Информационная часть представляет собой все элементы поля $GF(2^k)$. Таким образом, информационную часть кодовых слов можно рассматривать как линейный код с минимальным расстоянием один.

Поскольку обратный по умножению элемент в поле существует для каждого элемента (при условии доопределения обратного элемента для нуля: $0^{-1} = 0 \in GF(2^{k+m})$ [24]) и он уникален (то есть не существует двух различных элементов поля, обратные элементы которых совпадают), то проверочная часть кодовых слов также принимает значения всех элементов поля $GF(2^{k+m})$, причём каждый элемент встречается ровно один раз. Отсюда следует, что проверочную часть также можно рассматривать как линейный код с минимальным расстоянием, равным единице. При конкатенации информационной и проверочной частей кодового слова минимальное расстояние итогового кода становится не меньше двух.

Случайная величина может принимать одинаковые значения для различных сообщений y, потому при добавлении её к информационной и проверочной частям слова минимальное расстояние исследуемого ОСН кода не увеличивается.

Поскольку отдельные части кодового слова ОСН кода можно рассматривать как линейные коды, то к ним могут быть применены стандартные методы удлинения и укорачивания кодов [14]. Например, добавление к слову линейного кода общей проверки на чётность увеличивает на единицу как длину кода, так и минимальное расстояние. Отсюда легко могут быть получены

различные модификации исходной конструкции ОСН кодов, например, коды

$$C_1 = \{(y|x|f(x,y)|p(y))\}$$

и

$$C_2 = \{(y|x|f(x,y)|p(y|x)|p(f(x,y))\},$$

где $f(x,y) = (y|x)^{-1} \in GF(2^r)$;

$p(\cdot) \in GF(2)$ есть бит чётности двоичного вектора–аргумента.

Другими словами, $p(y) = wt(y) \bmod 2$, где $wt(\cdot)$ есть вес Хэмминга аргумента.

ОСН код, исправляющий однократную ошибку

Рассмотрим код $C_1 = \{(y|x|f(x,y) = (y|x)^{-1}|p(y))\}$. Его минимальное кодовое расстояние не менее трёх, из чего следует возможность исправления однобитовой ошибки. Пусть произошла ошибка $e = (e_y|e_x|e_f|e_p) \neq 0$, $wt(e) = 1$ (здесь и далее речь идёт о весе Хэмминга векторов, представленных в двоичном виде). Тогда искажённое слово ОСН кода можно записать как

$$\tilde{c} = c + e = \left(y + e_y|x + e_x|(y|x)^{-1} + e_f|p(y) + e_p\right) =$$

$$= \left(\tilde{y}|\tilde{x}|\tilde{f}(x,y)|\tilde{p}(y)\right).$$

Алгоритм декодирования данного слова сводится к вычислению восстановленного сообщения $\dot{y}$ из $(\dot{y}|\dot{x}) = f^{-1}(\tilde{f}(x,y)) = ((y|x)^{-1} + e_f)^{-1}$ и проверке выполнения следующих равенств:

$$(\dot{y}|\dot{x}) = (\tilde{y}|\tilde{x}), \tag{4.2}$$

$$p(\tilde{y}) = \tilde{p}(y), \tag{4.3}$$

$$p(\dot{y}) = \tilde{p}(y). \tag{4.4}$$

Первая проверка служит для обнаружения ошибки любой конфигурации с заданной вероятностью. Эта часть алгоритма декодирования относится к исходному алгоритму обнаружения ошибок с помощью ОСН кода. Легко заметить, что этот этап несколько изменён по сравнению с оригинальной конструкцией: вместо вычисления $f(x + e_x, y + e_y)$ выполняется обратное преобразование $f^{-1}(\cdot)$ от $f(x,y) + e_f$. При условии отсутствия ошибок его результат должен совпасть с $(y|x)$. Необходимо отметить, что однобитовая ошибка в y, x или $f(x,y)$ гарантированно будет обнаружена, так как не существует такой комбинации y, x и e, $wt(e_y|e_x|e_f) = 1$, при которой выполнится УМО $((y|x)^{-1} + e_f)^{-1} = (y + e_y|x + e_x)$. Вторая и третья проверки служат для определения местоположения ошибки: информационная часть, проверочная или бит чётности. В таблице 4.1 представлен анализ воздействия однобитовой ошибки на слово рассматриваемого кода. Выполнение равенств обозначается через логическую единицу (1), невыполнение — через логический ноль (0). При отсутствии ошибок все три равенства выполняются.

На основе представленной таблицы может быть построен алгоритм декодирования данного кода, позволяющий гарантированно исправлять однобитовые ошибки и обнаруживать все

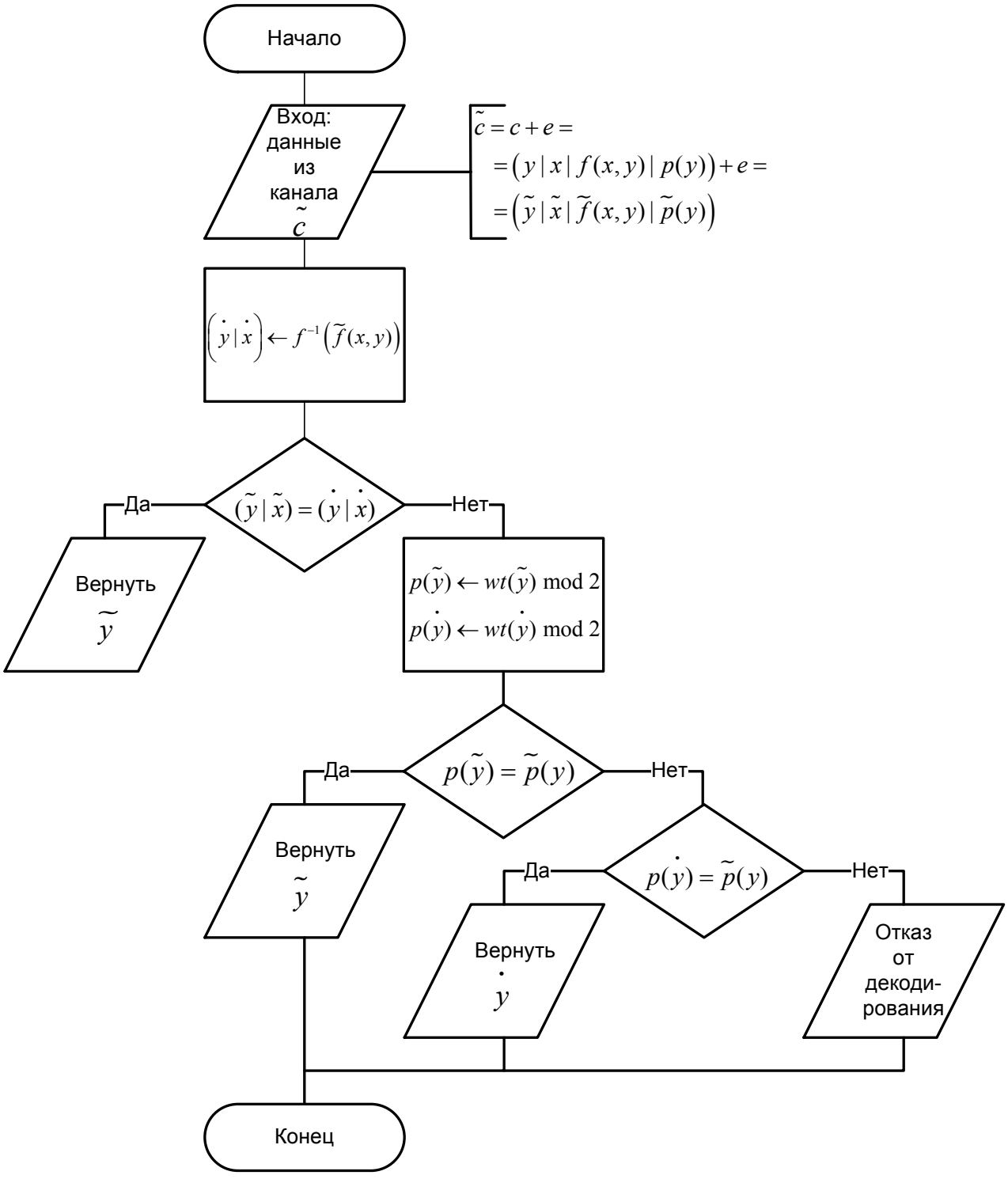


Рисунок 4.2 – Схема алгоритма декодирования кода C_1 с исправлением однобитовой ошибки

Таблица 4.1 – Анализ воздействия однобитовой ошибки на слово ОСН кода C_1 с проверкой на чётность информационной части

Позиция однобитовой ошибки	Результат проверки 4.2	Результат проверки 4.3	Результат проверки 4.4	Результат декодирования
Информационная часть y	0	0	1	$\dot{y}$
Случайная величина x	0	1	1	$\tilde{y}, \dot{y}$
Проверочная часть $f(x, y)$	0	1	0	$\tilde{y}$
Бит чётности $p(y)$	1	0	0	$\tilde{y}$, $\dot{y}$

остальные с заданными ненулевыми вероятностями $1 - P^{w}_{undet}$ для слабых манипуляций и $1 - P^{s}_{undet}$ — для сильных. Схема алгоритма декодирования проиллюстрирована на рисунке 4.2.

Алгоритм, оформленный в виде псевдокода, приведён ниже. Комментарии к выполняемым действиям обозначены символом «#».

Вход: принятый вектор $\tilde{c} = c + e = \left(\tilde{y}|\tilde{x}|\tilde{f}(x,y)|\tilde{p}(y)\right)$

$(\dot{y}|\dot{x}) \leftarrow f^{-1}\left(\tilde{f}(x,y)\right)$

\# Проверка наличия искажений в слове ОСН кода

если $(\tilde{y}|\tilde{x}) = (\dot{y}|\dot{x})$ **тогда**

    \# Информационная часть не искажена

    вернуть $\tilde{y}$

иначе

    \# Вычислить проверку на чётность вектора $\tilde{y}$

    $p(\tilde{y}) \leftarrow wt(\tilde{y}) \bmod 2$

    \# Вычислить проверку на чётность вектора $\dot{y}$

    $p(\dot{y}) \leftarrow wt(\dot{y}) \bmod 2$

    \# Определить позицию ошибки

    если $p(\tilde{y}) = \tilde{p}(y)$ **тогда**

        \# Информационная часть не искажена

        вернуть $\tilde{y}$

    иначе если $p(\dot{y}) = \tilde{p}(y)$ **тогда**

        \# Проверочная часть не искажена

        вернуть $\dot{y}$

    иначе

        \# Кратность ошибки превышена

        вернуть Отказ от декодирования

    конец если

конец если

Выход: Корректное информационное сообщение или сигнал об ошибке

Алгоритм 4.1 – Алгоритм декодирования ОСН кода с исправление однобитовой ошибки, представленный в виде псевдокода

Таким образом, для исправления ошибок достаточно добавления одного бита чётности. Исходный алгоритм декодирования ОСН кодов дополняется вычислением двух битов чётности и двумя сравнениями. Выполняемые при декодировании вычисления включают:

– поиск обратного элемента по умножению в поле $GF(2^r)$;

– два вычисления весов векторов;

– три проверки выполнения равенств.

Для сравнения, один из алгоритмов исправления ошибок для AMD кодов на основе полиномов (обладающий меньшей вычислительной сложностью) из [29] требует выполнения следующих операций:

– вычисление проверочных символов кода Хэмминга;

– выполнение $b+1$ сложений в поле $GF(2^m)$;

– вычисление синдрома кода Хэмминга и локатора ошибки ϵ;

– вычисление веса локатора ошибки ϵ;

– вычисление синдрома AMD кода по формуле $S = f(x,y) + f(x+e_x, y+e_y) + e_f$;

– осуществление двух проверок выполнения равенств;

– сравнение значения S с величинами $\epsilon \cdot u^j$ для $j \in \{1,2,\ldots,b\}$ до совпадения (таким образом, количество проверок выполнения равенства может достигать величины $b+2$).

Таким образом, сложность алгоритма исправления однократной ошибки и затраты на его реализацию значительно уступают аналогичным показателям решения, предложенного в [29].

Рассмотрим пример работы представленного выше алгоритма декодирования кода C_1, исправляющего однократную ошибку. Пусть в системе используется код конструкции $C_1 = \{(y|x|f(x,y)|p(y))\}$ с параметрами $k=4$, $m=2$, $r=k+m=6$. Вычисление проверочной части осуществляется в поле $GF(2^6)$ над полиномом x^6+x+1. Пусть в канал поступили следующие кодовые слова:

$$\begin{aligned} c_1 &= (0000|00|000000|0), \\ c_2 &= (0111|10|101101|1), \\ c_3 &= (0111|11|000110|1). \end{aligned}$$

В результате искажений из канала были получены следующие слова (жирным шрифтом выделены искажённые биты):

$$\begin{aligned} \widetilde{c_1} &= (0000|00|000000|\mathbf{1}), \\ \widetilde{c_2} &= (0111|10|1011\mathbf{1}1|1), \\ \widetilde{c_3} &= (0\mathbf{0}11|11|000110|1). \end{aligned}$$

Рассмотрим декодирование слова $\widetilde{c_1}$. На первом шаге вычисляем прообраз значения кодирующей функции $(\dot{y}|\dot{x}) \leftarrow 000000^{-1} = 000000$. Далее выполняем проверку (4.2) на наличие искажений в части слова, являющейся словом ОСН кода. Выполнение равенства $(\tilde{y}|\tilde{x}) = (\dot{y}|\dot{x})$ говорит о том, что с вероятностью $P^s_{undet} \leq 2^{-3}$ в данной части слова отсутствуют искажения, соответствующие сильным алгебраическим манипуляциям, и с вероятностью $P^w_{undet} = 2^{-5}$ — соответствующие слабым алгебраическим манипуляциям. Следовательно, в соответствии с алгоритмом декодирования принимается решение, что информационная часть слова не искажена. Значение $\tilde{y} = 0000$ является результатом декодирования.

Рассмотрим декодирование слова $\widetilde{c_2}$. Прообраз значения кодирующей функции, вычисленный на первом шаге алгоритма, равен $(\dot{y}|\dot{x}) \leftarrow 101111^{-1} = 101001$. Далее выполняем проверку (4.2)

на наличие искажений в части слова, являющейся словом ОСН кода. Невыполнение проверки ($(\tilde{y}|\tilde{x}) \neq (\dot{y}|\dot{x})$) говорит о том, что в слове обнаружены искажения. На следующем шаге алгоритма вычисляются две проверки на чётность: $p(\tilde{y}) \leftarrow wt(0011) \bmod 2 = 0$, $p(\dot{y}) \leftarrow wt(1010) \bmod 2 = 0$. Далее определяется позиция ошибки. Выполнение проверки (4.3) ($\tilde{p}(y) = p(\tilde{y})$) является показателем того, что с заданной вероятностью информационная часть слова не была искажена. Декодированным информационным сообщением является значение $\tilde{y} = 0111$.

Рассмотрим декодирование слова $\widetilde{c_3}$. На первом шаге вычисляем прообраз значения кодирующей функции $(\dot{y}|\dot{x}) \leftarrow 000110^{-1} = 011111$. Далее выполняем проверку (4.2) на наличие искажений в части слова, являющейся словом ОСН кода. Невыполнение проверки ($(\tilde{y}|\tilde{x}) \neq (\dot{y}|\dot{x})$) говорит о том, что в слове обнаружены искажения. На следующем шаге алгоритма вычисляются две проверки на чётность: $p(\tilde{y}) \leftarrow wt(0011) \bmod 2 = 0$, $p(\dot{y}) \leftarrow wt(0111) \bmod 2 = 1$. Далее определяется позиция ошибки. Невыполнение проверки (4.3) ($\tilde{p}(y) \neq p(\tilde{y})$) и выполнение проверки (4.4) ($p(\dot{y}) = \tilde{p}(y)$) является признаком того, что с заданной вероятностью проверочная часть слова ОСН кода не была искажена. Результатом декодирования является значение $\dot{y} = 0111$.

ОСН код, исправляющий однократную и обнаруживающий двукратную ошибку

Рассмотрим код $C_2 = \{(y|x|f(x,y))|p(y|x)|p(f(x,y)))\}$. Его кодовое расстояние не менее четырёх, из чего следует возможность исправления однобитовой ошибки и гарантированного обнаружения двукратной ошибки (SEC–DED код).

Пусть произошла ошибка $e = (e_y|e_x|e_f|e_{p_1}|e_{p_2}) \neq 0$, $wt(e) \leq 2$. Тогда искажённое слово ОСН кода можно записать как

$$\tilde{c} = c + e = (y + e_y|x + e_x|f(x,y) + e_f|p(y|x) + e_{p_1}|p(f(x,y)) + e_{p_2}) =$$

$$= \left(\tilde{y}|\tilde{x}|\tilde{f}(x,y)|\tilde{p}(y|x)|\tilde{p}(f(x,y))\right).$$

Алгоритм декодирования заключается в вычислении восстановленного сообщения $\dot{y}$ из $(\dot{y}|\dot{x}) = f^{-1}(\tilde{f}(x,y)) = ((y|x)^{-1} + e_f)^{-1}$ и проверке выполнения следующих равенств:

$$(\dot{y}|\dot{x}) = (\tilde{y}|\tilde{x}), \tag{4.5}$$

$$p(\tilde{y}|\tilde{x}) = \tilde{p}(y|x), \tag{4.6}$$

$$p(\tilde{f}(x,y)) = \tilde{p}(f(x,y)). \tag{4.7}$$

Выбор декодированного сообщения при однократной ошибке осуществляется аналогично рассмотренному ранее алгоритму декодирования ОСН кода с проверкой на чётность информационной части.

Обнаружение двукратной ошибки осуществляется по следующему принципу. Легко заметить, что равенство 4.5 может выполниться только при

$$wt(e_y) = 1 \text{ и } wt(e_x) = 0 \text{ и } wt(e_f) = 1$$

или

$$wt(e_y) = 0 \text{ и } wt(e_x) = 1 \text{ и } wt(e_f) = 1$$

или

$$wt(e_f) = 0 \text{ и } wt(e_x) = 0 \text{ и } wt(e_y) = 0.$$

В первых двух случаях $e_{p_1} = 0$ и $e_{p_2} = 0$ (так как суммарная кратность ошибки равна двум). Это приводит к невыполнению равенств (4.6) и (4.7). В последнем случае (при $wt(e_f) = 0$, $wt(e_x) = 0$, $wt(e_y) = 0$) $e_{p_1} = 1$ и $e_{p_2} = 1$. При этом также не выполняются равенства (4.6) и (4.7). Таким образом, не существует ошибки e кратности два, при которой все проверки будут выполнены или же произойдёт ошибочное декодирование в другое слово. Следовательно, двукратная ошибка гарантированно выявляется с помощью данного кода.

Принцип декодирования кода C_2 представлен в таблице 4.2. Алгоритм декодирования SEC–DED кода C_2 сводится к проверке выполнения трёх равенств (4.5), (4.6) и (4.7), после чего принимается решение о значении декодированного сообщения в соответствии с таблицей 4.2. Выполняемые вычисления включают поиск обратного элемента по умножению в поле $GF(2^r)$, два вычисления весов векторов, а также три проверки выполнения равенств.

Таблица 4.2 – Принцип работы предлагаемого декодера SEC–DED кода C_2

Результат проверки 4.5	Результат проверки 4.6	Результат проверки 4.7	Результат декодирования
0	0	0	Обнаружение ошибки
0	0	1	$\dot{y}$
0	1	0	$\tilde{y}$
0	1	1	Обнаружение ошибки
1	0	0	Обнаружение ошибки
1	0	1	$\tilde{y}, \dot{y}$
1	1	0	$\tilde{y}, \dot{y}$
1	1	1	Нет ошибок

Рассмотрим пример работы представленного выше алгоритма декодирования кода C_2, исправляющего однократную ошибку и обнаруживающего двукратную. Пусть в системе используется код конструкции $C_2 = \{(y|x|f(x,y))|p(y|x)|p(f(x,y)))\}$ с параметрами $k = 4$, $m = 2$, $r = k + m = 6$. Вычисление проверочной части осуществляется в поле $GF(2^6)$ над полиномом $x^6 + x + 1$. Пусть в канал поступили следующие кодовые слова:

$$\begin{aligned} c_1 &= (0000|00|000000|0|0), \\ c_2 &= (0111|10|101101|0|0), \\ c_3 &= (0111|11|000110|1|0). \end{aligned}$$

В результате искажений из канала были получены следующие слова (жирным шрифтом выделены искажённые биты):

$$\begin{aligned}\widetilde{c_1} &= (0000|0\mathbf{1}|00000\mathbf{1}|0|0),\\ \widetilde{c_2} &= (011\mathbf{0}|10|101101|0|\mathbf{1}),\\ \widetilde{c_3} &= (\mathbf{1}111|11|000110|1|0).\end{aligned}$$

Рассмотрим декодирование слова $\widetilde{c_1}$. На первом шаге вычисляем прообраз значения кодирующей функции $(\dot{y}|\dot{x}) \leftarrow 000001^{-1} = 000001$. Далее вычисляются две проверки на чётность: $p(\tilde{y}|\tilde{x}) \leftarrow wt(000001) \bmod 2 = 1$, $p(\dot{y}|\dot{x}) \leftarrow wt(000001) \bmod 2 = 1$. На следующем шаге алгоритма проверяются условия (4.5) ($(\dot{y}|\dot{x}) = (\tilde{y}|\tilde{x})$), (4.6) ($p(\tilde{y}|\tilde{x}) \neq \tilde{p}(y|x)$) и (4.7) ($p(\tilde{f}(x,y)) \neq \tilde{p}(f(x,y))$). В соответствии с таблицей 4.2 принимается решение об отказе от декодирования вследствие высокой кратности ошибок.

Для слова $\widetilde{c_2}$ прообраз значения кодирующей функции равен $(\dot{y}|\dot{x}) \leftarrow 101101^{-1} = 011110$. Проверки (4.5), (4.6) и (4.7) не выполняются. Следовательно, производится отказ от декодирования.

При декодировании слова $\widetilde{c_3}$ также начинаем с вычисления прообраза значения кодирующей функции $(\dot{y}|\dot{x}) \leftarrow 000110^{-1} = 011111$. Проверки (4.5) и (4.6) не выполняются, проверка (4.7) выполняется. В соответствии с таблицей 4.2 результатом декодирования выбирается $\dot{y} = 0111$.

Выводы по исправлению ошибок малой кратности с помощью ОСН кодов

Таким образом, за счёт добавления незначительного количества избыточности ОСН коды могут быть использованы в качестве SEC и SEC–DED кодов. Принцип, лежащий в основе предлагаемого метода декодирования ОСН кодов, заключается в выборе результата декодирования из информационной части кодового слова и прообраза значения проверочной функции в зависимости от выполнения дополнительных провероки (например проверки на чётность). Использование линейных кодов с большим расстоянием вместо кода с проверкой на чётность приводит к дальнейшему увеличению минимального расстояния получаемого кода. Это позволяет исправлять ошибки большей кратности. Информационная избыточность, требуемая для исправления ошибок малой кратности, меньше, чем у кодов из [29].

Алгоритмы, предложенные в данном разделе, используют дистанционные характеристики как линейных кодов, так и самих ОСН кодов, что приводит к снижению требуемой информационной избыточности. Вычислительная сложность алгоритмов снижается за счёт обратимости кодирующей функции, что позволяет использовать её прообраз в процессе декодирования. Вычислительная сложность представленных алгоритмов ниже, чем у алгоритмов из работы [29].

Важной особенностью использования AMD кодов для исправления ошибок является то, что необходимо различать ситуации, когда возникающие искажения вызваны естественными причинами, и когда они описываются моделью алгебраических манипуляций. Если вычислительное устройство принимает все искажения за естественные и пытается их исправлять, то это может негативно сказаться на целостности данных. Если же все искажения трактуются системой

как алгебраические манипуляции, то возникновение искажений, вызванных вкладом классических каналов с шумом, не приведёт к их исправлению. Таким образом, необходимо введение некоторого порога, который будет разграничивать естественные искажения малой кратности от искажений, описываемых моделью алгебраических манипуляций.

Особенное значение данный вопрос приобретает при искусственной природе возникающих помех. Так, в работе [29] предлагается рассматривать последовательно возникающие искажения как следствие искусственных помех. Это объясняется как высокой тактовой частотой современных устройств, так и ограниченностью временного разрешения применяемых инструментов привнесения помех. В работе выводятся нижняя и верхняя границы для пороговой величины количества последовательных искажённых данных. В случае, если количество ошибочных данных превышает этот порог, то системой принимается решение, что осуществляется искусственное привнесение помехв. В противном случае предполагается, что искажения имеют естественную природу.

В качестве альтернативного метода разделения искажений по их природе можно использовать следующий: для текущего используемого секретного ключа создать счётчик обнаруженных искажений. Если значение счётчика достигло некоторого порогового значения, то принимается решение о возможности компрометации ключа, и система производит его смену. Значение пороговой величины может выбираться на основе данных о требуемом количестве внедрённых ошибок для взлома используемого алгоритма шифрования, которые приводятся в соответствующих работах. Очевидно, что данный метод позволяет обеспечивать гарантированную защиту даже от злоумышленников, обладающих высокоточными методами внедрения помех.

В заключении стоит указать на тот факт, что не все кодирующие функции $f(x, y)$ удобно использовать для исправления ошибок. Например, ОСН код $\{(y|x|(y|x)^3)\}$ не всегда обладает минимальным кодовым расстоянием больше единицы. Это объясняется тем, что если проверочная функция является возведением в степень i в конечном поле $GF(2^{k+m})$, и i делит $k+m$, то проверочные части кодовых слов будут повторяться (например, $i = 3$, $k + m = 9$ бит). Это обуславливается наличием циклических подгрупп по умножению в поле. Кроме того, это приводит к тому, что функция не всегда обратима. Следовательно, увеличение расстояния кода с помощью добавления проверки на чётность будет применимо только к информационной части и даст код с расстоянием не менее двух.

4.1.3 Исправление повторяющихся ошибок

Использование предлагаемых кодов позволяет корректировать повторяющиеся ошибки. По аналогии с надёжными кодами, исправляющими ошибки [92], в случае повторения одинаковой конфигурации ошибок $e \neq 0 \in GF(2^n)$ в течение трёх и более тактов работы устройства можно исправить эту ошибку.

Рассмотрим алгоритм исправления повторяющихся ошибок при использовании 2–надёжного кода $C = \{(z|f(z) = z^3)\}$, $z \in GF(2^k)$ из [92]. Пусть ошибка $e = (e_z|e_f) \neq 0$, $e_z, e_f \in GF(2^k)$,

произошла на трёх необязательно последовательных тактах работы устройства и была обнаружена при декодировании. Искажённые кодовые слова поступают в блок, исправляющий повторяющиеся ошибки за счёт решения системы уравнений. Обозначим элементы первого и второго искажённых слов как

$$z_1 + e_z = \tilde{z}_1, \tag{4.8}$$

$$z_1^3 + e_f = \tilde{f}_1^3, \tag{4.9}$$

$$z_2 + e_z = \tilde{z}_2, \tag{4.10}$$

$$z_2^3 + e_f = \tilde{f}_2^3. \tag{4.11}$$

Значение z_1 из 4.8 может быть подставлено в 4.9, аналогично, z_2 из 4.10 может быть подставлено в 4.11. Сложив два полученных выражения и выполнив математические преобразования, получаем следующее выражение:

$$z_2^2 + z_2(\tilde{z}_1 + \tilde{z}_2) + (\tilde{z}_1^2 + \tilde{z}_2^2) + \frac{\tilde{f}_1 + \tilde{f}_2}{\tilde{z}_1 + \tilde{z}_2} = 0. \tag{4.12}$$

Напомним, что искажённые значения $\tilde{z}_1$, $\tilde{z}_2$, $\tilde{f}_1$ и $\tilde{f}_2$ являются частями декодируемых слов и известны блоку исправления ошибок. Уравнение 4.12 является квадратным в $GF(2^k)$ относительно z_2. Следовательно, существует не более двух решений, одно из которых соответствует корректному значению z_1, а другое — значению z_2. На данном этапе невозможно сопоставить решения уравнения величинам z_1 и z_2 (в работе [92] такая ситуация называется слабым надёжным декодированием), поэтому необходимо добавить ещё одно ошибочное слово.

Повторив аналогичные действия для второго и третьего искажённых слов, получаем

$$z_2^2 + z_2(\tilde{z}_2 + \tilde{z}_3) + (\tilde{z}_2^2 + \tilde{z}_3^2) + \frac{\tilde{f}_2 + \tilde{f}_3}{\tilde{z}_2 + \tilde{z}_3} = 0. \tag{4.13}$$

Из уравнений 4.12 и 4.13 можно получить линейное уравнение

$$z_2(\tilde{z}_1 + \tilde{z}_3) + (\tilde{z}_1^2 + \tilde{z}_3^2) + \frac{\tilde{f}_2 + \tilde{f}_3}{\tilde{z}_2 + \tilde{z}_3} + \frac{\tilde{f}_1 + \tilde{f}_2}{\tilde{z}_1 + \tilde{z}_2} = 0, \tag{4.14}$$

которое имеет единственное решение для z_2. Аналогичные уравнения могут быть составлены для z_1 и z_3:

$$z_1(\tilde{z}_2 + \tilde{z}_3) + (\tilde{z}_2^2 + \tilde{z}_3^2) + \frac{\tilde{f}_1 + \tilde{f}_3}{\tilde{z}_1 + \tilde{z}_3} + \frac{\tilde{f}_1 + \tilde{f}_2}{\tilde{z}_1 + \tilde{z}_2} = 0, \tag{4.15}$$

$$z_3(\tilde{z}_1 + \tilde{z}_2) + (\tilde{z}_1^2 + \tilde{z}_2^2) + \frac{\tilde{f}_2 + \tilde{f}_3}{\tilde{z}_2 + \tilde{z}_3} + \frac{\tilde{f}_1 + \tilde{f}_3}{\tilde{z}_1 + \tilde{z}_3} = 0. \tag{4.16}$$

С помощью данных уравнений могут быть вычислены корректные значения z_1, z_2 и z_3, а также значения ошибки $e = (e_z|e_f)$. Если все три вычисленные значения ошибок совпадают, то декодированные значения z_1, z_2 и z_3 поступают на выход устройства. Несовпадение вычисленных значений ошибки является следствием того, что имела место изменяющаяся во времени ошибка, а не повторяющаяся. В этом случае выдаётся отказ от декодирования.

Тот факт, что конструкция ОСН кодов основана на надёжных кодах, приводит к возможности исправления повторяющихся ошибок ОСН кодами. Алгоритм исправления ошибок полностью совпадает с алгоритмом для надёжных кодов. При этом, в качестве сообщения z используется конкатенация информационного сообщения y и значения случайной величины x: $z = (y|x)$. После декодирования корректное значение информационного сообщения извлекается из декодированного $\widehat{z}$.

Рассмотрим пример исправления повторяющихся ошибок с помощью ОСН кода. Пусть в системе используется код $C = \{(z|f(z) = z^3)\}$, где $z = (y|x) \in GF(2^6)$, $y \in GF(2^4)$, $x \in GF(2^2)$, $f(x, y) \in GF(2^6)$, поле $GF(2^6)$ порождено полиномом $x^6 + x + 1$. Пусть кодовые слова

$$\begin{aligned} c_1 &= (0000|00|000000), \\ c_2 &= (0111|10|111110), \\ c_3 &= (0111|00|111011). \end{aligned}$$

были искажены на величину $e = (1000|11|000001)$. Таким образом, из канала были получены следующие слова (жирным шрифтом выделены искажённые биты):

$$\begin{aligned} \widetilde{c_1} &= (\mathbf{1}000|\mathbf{11}|00000\mathbf{1}), \\ \widetilde{c_2} &= (\mathbf{1}111|\mathbf{01}|11111\mathbf{1}), \\ \widetilde{c_3} &= (\mathbf{1}111|\mathbf{11}|11101\mathbf{0}). \end{aligned}$$

При декодировании данных слов необходимо вычислить значения z_1, z_2 и z_3 с помощью формул (4.14), (4.14) и (4.16), соответственно. По формуле (4.14) получаем, что $\widehat{z_1} = 000000$. Аналогично, по формулам (4.14) и (4.16) получаем, что $\widehat{z_2} = 011110$ и $\widehat{z_3} = 011100$. Легко заметить, что результаты декодирования $\widehat{z_i} = (\widehat{y_i}|\widehat{x_i})$ совпадают с исходными значениями $z_i = (y_i|x_i)$ рассматриваемых кодовых слов. Соответственно, результатом декодирования являются сообщения $\widehat{y_1} = 0000$, $\widehat{y_2} = 0111$ и $\widehat{y_3} = 0111$. Таким образом, декодирование ошибки $e = (1000|11|000001)$, исказившей три кодовых слова, было выполнено успешно.

Необходимо указать, что в [97] описывается процедура исправления повторяющихся ошибок с использованием AMD кодов на базе обобщённых кодов РМ. Однако с ростом значений b и t значительно увеличивается объем необходимых вычислений. Для упрощения процедуры коррекции ошибок в статье предлагается защищать случайную величину x от ошибок с помощью надёжных кодов $\{(x|x^3)\}$ или $\{(x|x^{-1})\}$. Таким образом, требуемая аппаратная избыточность значительно увеличивается по сравнению с предлагаемой конструкцией.

4.1.4 Гибридный кодек, обнаруживающий алгебраические манипуляции

Принцип построения ОСН кодов позволяет спроектировать схему гибридного кодека, обнаруживающего алгебраические манипуляции. Гибридный кодек работает по следующему принципу: используется кодек надёжного кода, однако в качестве кодируемого сообщения поступает сочетание информационных и случайных символов, при этом их соотношение может изменяться [10].

Принципиальная схема кодека, реализующего ОСН код на основе перестановок в поле, представлена на рисунке 4.3. В качестве исходного надёжного кода используется 2–надёжный код $\{(z|z^3)\}$. Кроме того, в данной схеме используется блок исправления повторяющихся ошибок, который может быть заменён на блок исправления ошибок малой кратности.

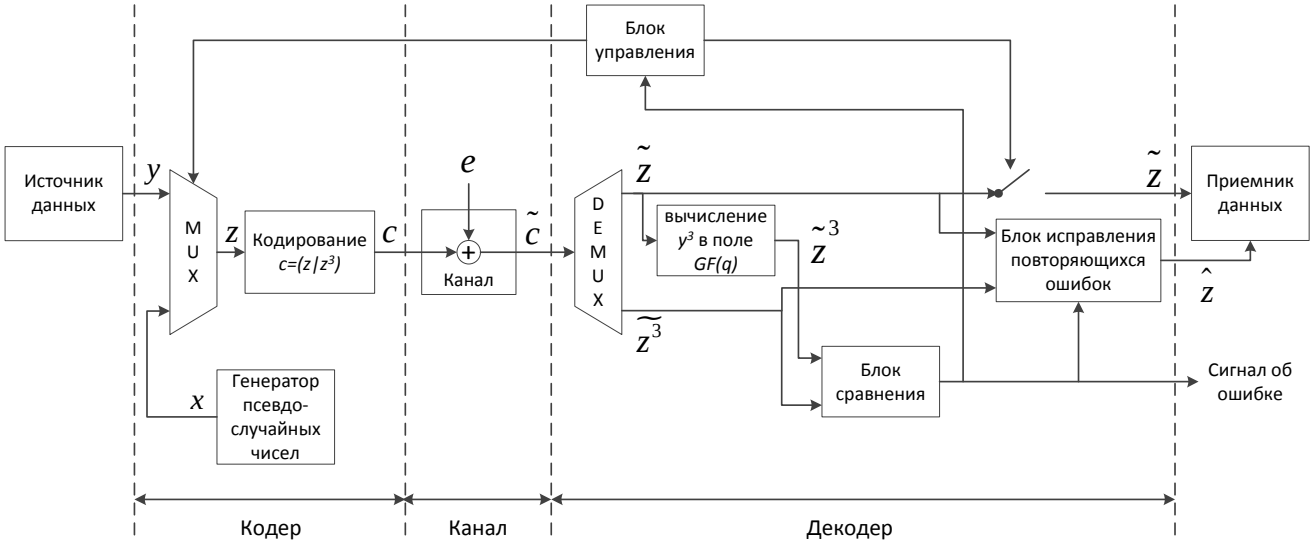


Рисунок 4.3 – Схема гибридного кодека, обнаруживающего алгебраические манипуляции

Описанная конструкция позволяет динамически менять скорость кода за счёт увеличения/уменьшения доли случайных символов в зависимости от ситуации в канале или же от значимости данных. При этом фиксированной остаётся длина кода, которая сохраняется для всех кодов, реализованных одним кодеком.

Применение данного кодека может значительно снизить требуемые аппаратные затраты на защиту технической системы. Кодек способен обеспечивать требуемый уровень обнаружения искажений для различных блоков устройства и может заменить несколько различных кодеков сильных и слабых ОСН кодов.

Рассмотрим пример. Допустим, криптографический модуль имеет три блока памяти, к целостности данных в которых предъявляются различные требования. В первом блоке требуется обнаружение только слабых манипуляций, при этом допускается невыявление ошибки с вероятностью не более 10^{-3}, для остальных блоков требует обнаружения сильных манипуляций: для второго — с вероятностью необнаружения ошибки не выше 0.5, для третьего — не выше 10^{-3}. В этом случае для обеспечения защиты устройства от искажений, описываемых моделью алгебраических манипуляций, может быть использован гибридный кодек с длиной 2–надёжного кода $n = 32$.

При работе с первым блоком памяти случайные символы не будут поступать на вход мультиплексора, следовательно, будет использоваться 2–надёжный код с параметрами $n = 32$, $k = 16$, $m = 0$, $r = 16$, скорость кода 1/2. Код обеспечивает вероятность необнаружения слабых манипуляций $P^w_{undet} = 2/2^k \approx 3 \cdot 10^{-5}$. Данная вероятность P^w_{undet} будет обеспечиваться всеми кодами,

получаемыми из исходного за счёт добавления случайных символов (то есть и для кодов, которые будут использоваться для защиты второго и третьего блоков).

Второй блок памяти требует защиты от сильных манипуляций. Следовательно, необходимо добавление случайных символов в исходное сообщение. Для достижения вероятности необнаружения сильных манипуляций 0.5 достаточно добавление $m = 2$ случайных битов. Получаем ОСН код с параметрами $n = 32$, $k = 14$, $m = 2$, $r = 16$, скорость кода 0.44. Вероятность невыявления сильных манипуляций $P^s_{undet} = 2/2^m = 0.5$, слабых — $P^w_{undet} \approx 3 \cdot 10^{-5}$.

Для достижения вероятности необнаружения сильных манипуляций 10^{-3} при защите третьего блока требуется $m = 10$ случайных бит. Параметры получаемого ОСН кода следующие: $n = 32$ бита, $k = 6$ бит, $m = 10$ бит, $r = 16$ бит, скорость кода 0.2. Получаем $P^s_{undet} = 2/2^m \approx 10^{-3}$, $P^w_{undet} \approx 3 \cdot 10^{-5}$.

Таким образом, использование гибридного кодека вместо трёх отдельных кодеков в данном примере позволило значительно снизить аппаратные расходы на защиту блоков памяти от ошибок, обеспечивая при этом требуемый уровень помехоустойчивости.

4.1.5 Сравнение с основными существующими конструкциями

Конструкция на основе обобщённых кодов РМ

В данном разделе приводится сравнение описываемой кодовой конструкции на основе перестановок в поле с AMD кодами из [25], которые основаны на обобщённых кодах Рида–Маллера (раздел 2.1.4). Обе сравниваемые конструкции являются AMD кодами в широком смысле, то есть гарантируют обнаружение любых ошибок с заданной вероятностью (включая конфигурации ошибок, не искажающих информационную часть).

При фиксированных параметрах k и m коды из [25] обеспечивают вероятность необнаружения ошибки $P_{undet} \leq (\lceil k/m \rceil + 1)/2^m$. Из этого следует, что при $k \leq m$ сравниваемые коды обладают одинаковой вероятностью необнаружения ошибок. В случае $k > m$ предлагаемый код обеспечивает более низкую вероятность необнаружения ошибки. В обоих случаях избыточность ОСН кода больше на k бит.

Рассмотрим пример для $k > m$. Пусть размер информационного сообщения y составляет $k = 8$ бит, а размер значения случайного числа — $m = 4$ бит. Тогда может быть построен AMD код на основе кодов РМ со следующими параметрами: размер проверочной части равен $r = 4$ бита, вероятность необнаружения манипуляций $P_{undet} = 4/2^4 = 0.25$.

ОСН код на основе перестановок в поле будет иметь $r = 12$ бит и следующие вероятности необнаружения манипуляций: $P^w_{undet} = 2^{-11} \approx 5 \cdot 10^{-4}$ и $P^s_{undet} = 2^{-3} = 0.125$.

ОСН код на основе операции скалярного умножения в поле будет иметь $r = 6$ бит и следующие вероятности необнаружения манипуляций: $P^w_{undet} = 2^{-6} \approx 0.016$ и $P^s_{undet} = 2^{-2} = 0.25$.

Таким образом, предлагаемые коды позволяют достичь меньшей вероятности необнаружения искажений, описываемых моделью алгебраических манипуляций, чем AMD коды на основе кодов РМ для случаев, когда $k > m$.

Сравнение кодов с одинаковой скоростью. Пусть есть ОСН код с $y \in GF(2^k)$, $x \in GF(2^m)$ и $f(x, y) \in GF(2^{k+m})$,

$$P^s_{undet} \leq 2/2^m.$$

Согласно [25] можно построить код (увеличив размер случайного числа и проверочной части так, чтобы скорость кодов совпадали) с $y \in GF(2^k)$, $x \in GF(2^{m+\lfloor k/2 \rfloor})$ и $f(x, y) \in GF(2^{m+\lfloor k/2 \rfloor})$, вероятность необнаружения ошибки которого ограничена сверху величиной

$$P_{undet} \leq (\lfloor k/(m + \lfloor k/2 \rfloor) \rfloor + 1)/2^{m+\lfloor k/2 \rfloor} < 2/2^{m+\lfloor k/2 \rfloor}.$$

Из этого следует, что при заданной скорости кода конструкция на основе кодов РМ демонстрирует более высокий уровень защиты от алгебраических манипуляций.

Стоит отметить, что при слабой модели манипуляций предлагаемая в диссертационной работе конструкция ОСН кодов пропускает ошибки с вероятностью в 2^k раз меньше, чем коды на основе обобщённых кодов РМ.

Конструкция ОСН кодов на основе перестановок в поле позволяет исправлять повторяющиеся ошибки и ошибки малой кратности с меньшей сложностью, чем конструкция AMD кодов на основе обобщённых кодов РМ. Кроме того, для ОСН кодов возможно динамическое изменение скорости кода и обнаруживающей способности.

ОСН коды на основе операции скалярного умножения требуют случайную величину и проверочный символ большего размера для достижения такой же обнаруживающей способности, как коды на основе полиномов, что приводит к меньшей скорости ОСН кода. При этом, обнаружение слабых манипуляций ОСН кодами осуществляется со значительно большей вероятность. Основное преимущество конструкции ОСН кодов на основе операции скалярного умножения в поле заключается в более простой кодирующей функции, являющейся полиномом первой степени (у кодов на основе полиномов — минимум третьей степени). Таким образом, использование ОСН кода может быть эффективно в приложениях, имеющих ограничения по аппаратной избыточности и вычислительной сложности.

Конструкции на основе линейных помехоустойчивых кодов и операции умножения в поле

Код на основе ЛПК является AMD кодом в узком смысле и обеспечивает $P_{undet} = 2^{-m}$ при $k = m = r$ [27]. Код на основе операции умножения информационного и случайного компонентов кодового слова также является AMD кодом в узком смысле и обеспечивает аналогичную вероятность $P_{undet} = 2^{-m}$ при $k = m = r$ [27]. Эту вероятность достигают недвоичные ОСН коды. При этом, длина ОСН больше на m символов. Необходимо отметить, что конструкция ОСН кодов обеспечивает гибкость в выборе параметров кода, что выгодно отличает её от конструкций на основе ЛПК и операции умножения.

4.1.6 Заключение по кодовой конструкции

Данная кодовая конструкция представляет теоретический интерес, поскольку является обобщением надёжных кодов на случай искажений, описываемых сильной моделью алгебраических манипуляций. Фактически, используется нелинейный надёжный код размерности $k + m$ из пространства, размерность которого $2(k+m)$. При этом каждому из 2^k информационных сообщений y соответствуют 2^m кодовых слов. Выбор одного из этих слов осуществляется за счёт случайной величины x. Если $m = 0$, то присутствует однозначное соответствие информационного сообщения кодовому слову. В данном случае полученный код является нелинейным надёжным кодом, обнаруживающим только слабые манипуляции. Если $m > 0$, то каждое сообщение в результате кодирования может быть преобразовано более чем в одно кодовое слово исходного надёжного кода. Привнесение случайности в процесс кодирования приводит к возможности обнаружения любых сильных манипуляций. Следовательно, полученный код является сильно защищённым AMD кодом. С увеличением размера m случайного числа обеспечивается рост вероятности обнаружения искажений, соответствующих сильным манипуляциям.

Кроме того, данная конструкция является вторым известным AMD кодом в широком смысле (первая — коды на основе полиномов). Других конструктивных методов построения кодов широком смысле на данный момент не существует.

В силу полиномиальной зависимости сложности кодирующей функции от размера проверочной части ОСН коды могут быть эффективно использованы для кодирования данных небольшого размера, например, во флэш–памяти типа NAND [31]. Несмотря на то, что кодовая конструкция из [25] требует меньшей избыточности, обобщённые надёжные коды обладают рядом практических преимуществ:

1. Возможность и простота динамического изменения скорости кода за счёт изменения соотношения количества информационных и случайных битов в кодируемом сообщении. Фактически, кодек обобщённых надёжных кодов может содержать в себе некоторый набор кодов с разными характеристиками. Данная особенность позволяет использовать один кодек для различных типов памяти или различных режимов работы устройства.

2. Модификации ОСН кодов с целью увеличения минимального кодового расстояния. Привносимая при этом информационная избыточность меньше избыточности, необходимой для AMD кодов на основе кодов РМ [29]. Наличие вычислительно простых алгоритмов исправления ошибок малой кратности для модификаций ОСН кодов.

3. Процедура исправления повторяющихся ошибок, описанная для надёжных кодов в [92]. С её использованием предлагаемый код способен исправлять ошибки, которые проявляются на 3 и более тактах работы устройства, за счёт решения системы уравнений. Подобная процедура для кодов из [25] требует значительных аппаратных, информационных и вычислительных затрат [97].

4. Высокая помехоустойчивость при искажениях, описываемых моделью слабой алгебраической манипуляции.

5. Возможность уменьшения информационной избыточности за счёт использования модификации на основе разбиения информационного вектора (модификация описана в разделе 4.3.3).

Описанный обобщённый надёжный систематический код совмещает в себе как R–надёжный, так и сильно защищённый AMD коды, обеспечивая вероятности необнаружения ошибок, равные $P^w_{undet} \leq R/2^{k+m}$ и $P^s_{undet} \leq R/2^m$ для перестановок в поле, и $P^w_{undet} \leq 2^- a + b$ и $P^s_{undet} \leq 2^{-b}$ для операции скалярного умножения, соответственно.

4.2 Надёжный код на основе экспоненциальной почти совершенной нелинейной функции

4.2.1 Экспоненциальная нелинейная функция

В статье [89] Найберг показала, что функция $f(y) = u^y \bmod p$, где y — элемент абелевой группы $G = \{0, ..., p-1\}$ (p — простое число), а u — элемент порядка q из поля $GF(p)$, является разностно $((p-1)/q+1)$–равномерным отображением. Исследуем степень нелинейности этой функции. Пусть $a, b \in G$ и $a \neq 0$. Тогда уравнение

$$u^{(y+a) \bmod p} - u^y = b \tag{4.17}$$

эквивалентно

$$\begin{cases} u^{(y+a)} - u^y = b \text{ и } 0 \leq y \leq p-a-1 \\ \text{или} \\ u^{(y+a-p)} - u^y = b \text{ и } p-a \leq y \leq p-1. \end{cases} \tag{4.18}$$

Так как решение y уравнения

$$u^{y+a} - u^y = b$$

уникально по модулю q, то из этого следует, что первое уравнение из (4.18) имеет не более $\lfloor (p-a)/q \rfloor$ корней в G. Аналогично, второе уравнение из (4.18) имеет не более $\lfloor a/q \rfloor$ корней. Следовательно, уравнение (4.17) имеет не более

$$\left\lfloor \frac{p-a}{q} \right\rfloor + \left\lfloor \frac{a}{q} \right\rfloor = \frac{p-1}{q} + 1$$

решений в G.

Выбирая в качестве u примитивный элемент поля $GF(p)$, мы получаем из (4.17) уравнение вида $f(y+a) - f(y) = b$, которое имеет не более двух корней. Оно аналогично проверочному выражению надёжных кодов с точностью до использованных обозначений (раздел 2.1.4). С помощью этого соотношения блок обнаружения ошибок выявляет наличие искажений. Стоит

отметить, что при использовании в качестве u примитивного элемента поля, $f(y) = u^y \bmod p$ становится почти совершенно нелинейной функцией с $P_f = 2/p$. Параметр P_f определяет степень нелинейности функции (раздел 2.1.4).

4.2.2 Конструкция кода

Согласно Теореме 2 из [24], конкатенация информационной части $y \in GF(2^k)$ и проверочной части $f(y) \in GF(2^r)$, образует кодовые слова $(y|f(y))$ надёжного систематического кода, параметры которого определяются следующим образом: $R = 2^k \cdot P_f$, $n = k + r$, $M = 2^k$. Если для вычисления проверочных символов использовать экспоненциальную функцию $f(y) = u^y \bmod p$, где в качестве u взят примитивный элемент поля, то при объединении информационной и проверочной частей получаем систематический надёжный код со следующими параметрами: $k = \lceil \log_2(p) \rceil$, $n = 2k$, $R = 2$ [2].

Рассмотрим пример работы предлагаемого кода при $p = 31$ и $u = 3$, $k = \lceil \log_2(31) \rceil = 5$ бит. Пусть в системе возникает помеха, приводящие к искажению обрабатываемых данных на величину $e = (e_y|e_f) = (5|11)$. Среди всех возможных кодовых слов кода $C = \{(y|f(y) = u^y)\}$ существуют всего два слова $c_1 = (13|24)$ и $c_2 = (27|23)$, результатом прибавления ошибки к которым станут кодовые слова $c_3 = c_1 + e = (18|4)$ и $c_4 = c_2 + e = (1|3)$. Прибавление ошибки e к другим кодовым словам приведёт к запрещённым комбинациям, что является признаком ошибки. Таким образом, при предположении о равновероятности кодовых слов получаем, что вероятность необнаружения слабой модели манипуляции ограничена вероятностью $P^w_{undet} \leq 2/p = 2/31 \approx 0.07$.

4.2.3 Применимость кодовой конструкции

Предложенные на данный момент 2–надёжные коды в качестве почти совершенно нелинейных функций используют степенные функции и функцию инвертирования в поле [24]. Эти коды являются надёжными кодами с заданной вероятностью обнаружения ошибок при аддитивной модели помехи.

Однако не всегда ошибки в устройстве могут быть описаны аддитивной моделью. Как показывает практика, для Flash–памяти, оптических дисков и сетей характерна достаточно большая разница между вероятностями переходов $0 \to 1$ и $1 \to 0$ [98]. Зачастую используется допущение, что в таких системах возможен только один тип переходов. Такие ошибки получили название асимметричных. Более того, исследования показали, что для некоторых LSI/VLSI ROM и RAM характерны однонаправленные ошибки. Они отличаются от асимметричных тем, что оба перехода $0 \to 1$ и $1 \to 0$ возможны, но в каждом отдельном слове встречается только один тип перехода — один тип асимметричной ошибки. Математически такая ошибка может быть представлена как побитовая конъюнкция/дизъюнкция кодового слова c и некоторого вектора w, состоящего из нулей и единиц: переход $0 \to 1$ - $c \vee w$; переход $1 \to 0$ - $c \wedge w$.

При искусственном происхождении ошибки — наведении помех криптоаналитиком — также возможны асимметричные и однонаправленные модели ошибок (как в примере атаки на DES

с недифференциальным анализом помех из раздела 1.2.3). Возможность осуществлять переход всех битов в 0 даёт злоумышленнику заведомо успешный метод внедрения необнаруживаемых ошибок даже при использовании существующих надёжных кодов.

Рассмотрим пример. Для защиты блока памяти используется 2–надёжный код $(y|f(y) = y^3)$ над полем $GF(p^n)$. Допустим, произошла ошибка, которая привела к переходу $0 \to 1$, $1 \to 1$ для всех битов кодового слова. Тогда проверочное уравнение $f(y \wedge 0) = f(y) \wedge 0$ при $f(y) = y^3$ будет удовлетворено при любых y, и искажение не будет обнаружено. Аналогичная ситуация и для функции инвертирования в поле, так как она доопределяется тем условием, что $0^{-1} = 0$ [24].

Предлагаемый код обеспечивает защиту даже при однонаправленных ошибках. Это обеспечивается тем, что в общем случае $u^0 \neq 0$ и $u^{p-1} \neq p-1$. Наиболее эффективным сценарием возникновения ошибки при данном надёжном коде представляется приведение данных к кодовым словам наименьшего или наибольшего веса Хемминга (в зависимости от направления ошибки). Например, наименьшим весом обладает слово $(0|u^0 = 1)$. Например, при искусственной природе искажений злоумышленник может обнулить все биты информационной части кодового слова, а у проверочной части оставить нетронутым только младший бит. Далее, в зависимости от значения младшего бита выполнится или нет проверочное соотношение (0 — не выполнится, 1 — выполнится). Таким образом, вероятность необнаружения однонаправленной ошибки ограничена сверху значением 0.5 (при равномерном распределении сообщений). Следовательно, предлагаемый код обеспечивает защиту не только от аддитивных ошибок, описываемых слабой моделью алгебраических манипуляций, но и обладает лучшей обнаруживающей способностью относительно однонаправленных ошибок по сравнению с аналогами.

Очевидно, что процедуры кодирования и декодирования предлагаемого кода обладают более высокой вычислительной сложностью, нежели существующие кодовые конструкции. Однако, если криптографический модуль использует алгоритм Диффи–Хеллмана для генерации секретного ключа между двумя устройствами, то блоки, выполняющие возведение в степень по модулю простого числа, могут быть использованы и для процедур кодирования и декодирования. Более того, в обоих случаях требуется возведение фиксированного основания в степень, значение которой является переменной величиной. Поэтому могут быть применены эффективные алгоритмы возведения в степень по модулю простого числа, например, метод Евклида для фиксированного основания [87]. Также возможно эффективное использование данного надёжного кода в криптографических модулях, использующих возведение в степень при шифровании и дешифровании. Примерами таких алгоритмов являются шифры RSA и Эль–Гамаля, а также электронные цифровые подписи на их основе. Соответственно, использование уже реализованных в устройстве блоков сводит аппаратные затраты к минимуму.

4.2.4 Заключение по кодовой конструкции

Предлагаемая кодовая конструкция надёжного кода на основе экспоненциальной функции позволяет обеспечивать гарантированное обнаружение искажений, описываемых слабой моде-

лью атаки. Данный код обеспечивает вероятность необнаружения ошибки $P^{w}_{undet} \leq 2/p$, которая при определённых параметрах (значение p близко к степени двойки слева) близка к обнаруживающей способности существующих кодовых конструкций. При этом распространённость операции экспоненцирования в криптографических приложениях позволяет во многих случаях уменьшить аппаратные затраты на реализацию данного кода за счёт повторного использования существующих блоков возведения в степень. Кроме того, преимуществом данного кода является тот факт, что он обеспечивает базовую защищённость данных от асимметричных ошибок, что не характерно для остальных надёжных кодов. Данный код может быть эффективно использован для построения на его основе обобщённого систематического надёжного кода.

4.3 Модификации AMD кода на основе операции умножения информационного и случайного компонентов

4.3.1 Код на основе операции умножения информационного и случайного компонентов

Рассмотрим код $C = \{(y|x|f(x,y) = x \cdot y)\}$ на основе операции умножения, описанный в разделе 2.1.4. УМО кода выглядит следующим образом:

$$x \cdot y + e_f = (x + e_x) \cdot (y + e_y).$$

Легко заметить, что максимальное количество решений данного УМО относительно x при фиксированных y и e равно единице, так как:

$$e_f = x \cdot e_y + y \cdot e_x + e_x \cdot e_y,$$

$$x = \frac{e_f + e_y \cdot e_x + y \cdot e_x}{e_y}.$$

Правая часть последнего выражения при фиксированной величине ошибки представляет собой константу; если значение случайной величины x равно этой константе, то привнесённая ошибка $e = (e_y|e_x|e_f)$ останется необнаруженной. Случайная величина x распределено равномерно, следовательно, вероятность необнаружения любой ошибки e при фиксированном y равна

$$P_{undet} = \frac{|\{x : f(x,y) + e_f = f(x + e_x, y + e_y)\}|}{|\{x\}|} = \frac{1}{2^m} = 2^{-k}. \tag{4.19}$$

Последний переход справедлив потому, что для данной конструкции $k = m$.

Из УМО следует, что данный код является AMD кодом в узком смысле, то есть неприменим при такой ошибке e, у которой $e_y = 0$.

Продемонстрируем принцип кодирования AMD кодом на основе операции умножения в поле. Пусть $k = 2$ битам, а информационное сообщение $y \in GF(2^2)$ приняло значение (10). На рисунке 4.4 представлены все соответствующие ему дополненные сообщения и кодовые слова.

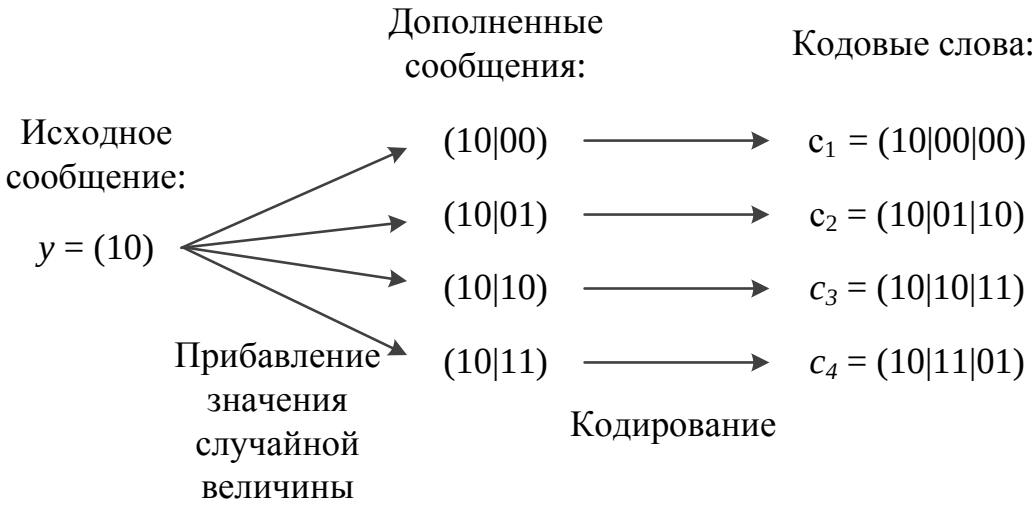


Рисунок 4.4 – Принцип кодирования AMD кодом на основе операции умножения в поле с параметрами $k = m = r = 2$ на примере сообщения (10)

Информационная, случайная и проверочная части кодового слова разделены символом «|», соответственно. Вычисления проверочной части выполнены в поле Галуа $GF(2^2)$ с порождающим полиномом $x^2 + x + 1$.

Стоит отметить, что о целесообразности использования умножения в поле для обнаружения ошибок писали ещё В. И. Коржик и Л. М. Финк в [15]. В их книге предлагался универсальный метод кодирования для систем связи с обратной связью в каналах со случайной структурой. Опишем предложенный ими метод в терминах, используемых в данной диссертационной работе. Пусть есть информационное сообщение $y \in GF(2^k)$ и случайная величина $x \in GF(2^m)$, $k < m$. При этом нулевое значение случайной величины является запрещённым, то есть $x \neq 0 \in GF(2^m)$. Канал, по которому передаются данные, считается каналом с аддитивным шумом. Метод кодирования заключается в выполнении двух действий. Прежде всего информационное сообщение дополняется нулями до m бит. После чего выполняется его умножение на значение случайной величины x. Результат передаётся по каналу связи. На приёмной стороне полученный из канала блок подвергается обратным преобразованиям. При приёме возможны две ситуации: блок может быть принят без ошибки с вероятностью P_{corr}, или же блок может быть принят с ошибкой, вероятность чего равна $1 - P_{corr}$. В последнем случае информационный блок с равной вероятностью переходит в любой другой блок (доказательство приводится в книге).

Ошибка остаётся необнаруженной в том случае, если ошибочно принятый блок принадлежит коду. Если блок был принят без ошибок, то после деления на x последние $m - k$ разрядов будут нулевыми. Это и является признаком безошибочного приёма. Признаком ошибки принимается наличие ненулевых символов в этих разрядах. Таким образом, работа декодера сводится к отбрасыванию последних разрядов восстановленного блока, если они равны нулю, и отбраковыванию блока, если в этих разрядах имеется хотя бы один ненулевой символ. Отсюда легко получить вероятность необнаружения ошибки. Так как при наличии ошибки все возможные значения восстановленного блока равновероятны, и из 2^m блоков лишь 2^k принадлежат коду (один из которых

является корректным), получаем следующую вероятность необнаружения ошибки

$$P_{undet} = (1 - P_{corr})\frac{2^k - 1}{2^m - 1}.$$

Полагая $P_{corr} = 0$ (то есть возникновение искажение гарантировано), получаем:

$$P_{undet} = \frac{2^k - 1}{2^m - 1}.$$

Таким образом, предложенный В. И. Коржиком и Л. М. Финком универсальный метод кодирования позволяет организовывать передачу данных в каналах связи с обратной связью, обеспечивая высокую степень помехоустойчивости. При обнаружении ошибки в принятом блоке по обратной связи осуществляется, например, переспрос на его повторную передачу. Универсальный метод кодирования для каналов со случайной структурой обеспечивает более высокую вероятность пропуска ошибок, чем исследуемый AMD код. Кроме того, в модели из [15] подразумевается, что приёмник и передатчик имеют точно синхронизированную случайную величину x, что ограничивает область применения данного метода кодирования. Использование AMD кода, основанного на операции умножения, позволяет обеспечить минимальную достижимую вероятность необнаружения ошибки $P_{undet} = 1/2^m$. Значение случайной величины x является частью кодового слова, передаётся по каналу и не требует дополнительной синхронизации передатчика и приёмника.

Основным недостатком кодовой конструкции является отсутствие гибкости при выборе параметров кода [19]. Фактически, размер k информационного сообщения полностью определяет длину кода $n = 3k$, размер случайного числа — k бит, а также вероятность необнаружения ошибки $P_{undet} = 2^{-k}$. Ниже будут представлены два варианта модификаций данной конструкции, которые предоставляют большую гибкость при выборе параметров кода [3].

4.3.2 Модификация на основе расширения случайной величины

Наиболее естественным методом модификации описанной конструкции является использование случайной величины из меньшего поля Галуа, то есть $x \in GF(2^m)$, $m < k$. Для выполнения умножения формируется вектор $\hat{x}$ путём дополнения двоичного представления элемента x $k - m$ нулями, то есть выполняется отображение $g(x) = \hat{x}$, $\hat{x} = (0, ..., 0, x_{m-1}, x_{m-2}, ..., x_0)$, где $(x_{m-1}, x_{m-2}, ..., x_0)$ есть двоичное представление элемента поля x. Полученный элемент большего поля $\hat{x} \in GF(2^k)$ используется для выполнения операции кодирования согласно оригинальной конструкции. Кодовое слово выглядит следующим образом:

$$c = (y \in GF(2^k) | x \in GF(2^m) | \hat{x} \cdot y \in GF(2^k)),$$

то есть длина кода уменьшается на $k - m$ бит.

На приёмной стороне перед проверкой УМО полученное из канала значение случайной величины $x + e_x$ ещё раз подвергается отображению $g(x + e_x) = \widehat{x + e_x}$.

Утверждение 4.5. *Вероятность необнаружения сильной алгебраической манипуляции ограничена сверху следующим выражением:*

$$P_{undet} \leq 2^{-m}.$$

Доказательство. Рассмотрим формулу 2.4. Мощность множества значений случайной величины x, стоящая в знаменателе дроби, равна 2^m. Необходимо определить значение числителя дроби из 2.4. Легко заметить, что для данной модификации исходной кодовой конструкции числитель дроби приобретает следующий вид:

$$|\{x : f(\hat{x}, y) + e_f = f(\widehat{x + e_x}, y + e_y)\}, g(x) = \hat{x}| =$$

$$|\{\hat{x} : f(\hat{x}, y) + e_f = f(\widehat{x + e_x}, y + e_y)\}, \exists g^{-1}(x)|,$$

то есть добавляется условие, что для получаемого решения УМО должен существовать прообраз среди $x \in GF(2^m)$. Дополнительное ограничение на мощность множества приводит к тому, что обнаруживающая способность кода становится неравномерной. Если для оригинальной конструкции значение числителя было равно 1 для всех возможных y и $e \neq 0$, то при данной модификации некоторая часть комбинаций y и $e \neq 0$ приведёт к решениям УМО, которые не имеют прообраза относительно отображения $g(x)$, то есть значение числителя будет равно 0. Таким образом, часть ошибок не будет обнаруживаться с вероятностью $P_{undet}(e) = 0/2^m = 0$, в то время как остальные — с $P_{undet}(e) = 1/2^m = 2^{-m}$, где через $P_{undet}(e)$ обозначена вероятность необнаружения конкретной ошибки e, соответствующей сильной модели манипуляции.

Таким образом, вероятность невыявления сильной манипуляции для данной кодовой конструкции составляет $P_{undet} \leq 2^{-m}$. ■

4.3.3 Модификация на основе разбиения информационного сообщения

Вторым вариантом модификации оригинальной кодовой конструкции является следующий: пусть $k = m \cdot u$ (в противном случае либо уменьшается разрядность m случайного числа x, либо размер k информационного вектора y увеличивается до необходимого значения за счет добавления, например, нулей). Далее информационный вектор делится на u частей по m бит: $y = (y_1|y_2|...|y_u)$, где $y_i \in GF(2^m)$, $i = 1, 2, ..., u$. Каждый y_i на основе случайного числа m подвергается процедуре кодирования согласно оригинальной конструкции, то есть получаем набор из u кодовых слов:

$$c_i = (y_i \in GF(2^m)|x \in GF(2^m)|y_i \cdot x \in GF(2^m)), i = 1, 2, ..., u.$$

Таким образом выполняется u процедур кодирования векторов y_i на основе фиксированного случайного числа x. Полученные промежуточные кодовые слова c_i объединяются в одно слово c следующим образом:

$$c = (y|x|f(x, y)),$$

$$c = (y_1|y_2|...|y_u|x|y_1 \cdot x|y_2 \cdot x|...|y_u \cdot x).$$

Итоговое кодовое слово имеет длину $n = 2k + m$. На приёмнике слово c раскладывается в u кодовых слов c_i, $i = 1, 2, ..., u$, после чего выполняется u проверок их УМО. При обнаружении ошибки хотя бы в одном кодовом слове весь набор $y = (y_1|y_2|...|y_u)$ признается ошибочным. Рассмотрим вероятность обнаружения сильных манипуляций с помощью данной модификации AMD кода.

Утверждение 4.6. *Вероятность необнаружения искажения, описываемого сильной моделью алгебраической манипуляции, ограничена сверху величиной* $P_{undet} \leq 2^{-m}$.

Доказательство. Очевидно, что чем больше блоков y_i (и, соответственно, кодовых слов c_i) подвергается искажению, тем меньше вероятность того, что искажение не будет выявлено (предполагается, что вероятность возникновения необнаруживаемой ошибки меньше единицы). Таким образом, большей вероятностью остаться необнаруженой обладает ошибка, искажающая только один из u блоков y_i. При этом вероятность остаться необнаруженной будет максимальна. Рассмотрим эту вероятность, так как именно она определяет верхнюю границу на P_{undet}.

Искажение одного из блоков y_i можно рассматривать как искажение одного из кодовых слов оригинальной конструкции с параметрами: $y \in GF(2^m)$, $x \in GF(2^m)$, $f(x, y) \in GF(2^m)$, $n = 3m$. Получаем, что вероятность необнаружения сильной манипуляции одного блока равна $P_{undet} = 2^{-m}$. Отсюда, вероятность необнаружения искажения кодового слова, описываемого моделью сильной алгебраической манипуляции, ограничена снизу $P_{undet} \leq 2^{-m}$. ■

Необходимо отметить, что тот факт, что все u кодовых слов c_i получены на основе одного и того же случайного числа x, не способствует увеличению вероятности необнаруженного возникновения ошибки в слове c при данном методе декодирования. Если бы решение о наличии ошибок принималось независимо на уровне каждого блока, а не на уровне целого слова, то в этом случае возникновение искажений оставалось бы необнаруженным с большей вероятностью. Рассмотрим пример с искусственной природой происхождения помех. В этом случае злоумышленник был бы способен проводить атаку с более высокой вероятностью успеха. Для этого он бы для каждого из c_i случайно выбирал некоторые неповторяющиеся числа $\dot{x}_i \in GF(2^m)$, которые предполагал бы соответствующими значению x. Исходя из $\dot{x}_i$ рассчитывались бы такие значения ошибок e_i, чтобы выполнялось УМО. Таким образом, злоумышленник смог бы перебрать u возможных значений случайного числа. В этом случае вероятность успешного внедрения ошибки в один из u блоков (то есть успешное искажение кодового слова c при таком методе декодирования) равнялась бы $P_{undet} = u/2^m$ при $u < 2^m$ и $P_{undet} = 1$ при $u \geq 2^m$.

4.3.4 Заключение по модификациям

Как видно, приведённые модификации кодовой конструкции AMD кодов, основанной на операции умножения в поле, предоставляют возможность гибкого выбора длины кода и соответствующей вероятности обнаружения искажений, описываемых сильной моделью манипуляций. Зависимость вероятности обнаружения сильной манипуляции от размера случайной величины

x аналогична оригинальной конструкции. Таким образом, представленные модификации нелинейного кодового метода на основе кода на базе операции умножения информационного и случайного компонентов могут быть эффективно использованы в технических средствах с целью повышения их помехоустойчивости и качества функционирования.

Аппаратная сложность реализации первой модификации соответствует сложности оригинальной кодовой конструкции. Сложность реализации умножения в конечном поле имеет квадратичную зависимость от степени расширения поля. Благодаря этому вторая предложенная модификация позволяет значительно снизить сложность кодека за счёт того, что умножение вычисляется в меньшем поле. Схема одного из вариантов реализации кодера приведена на рисунке 4.5.

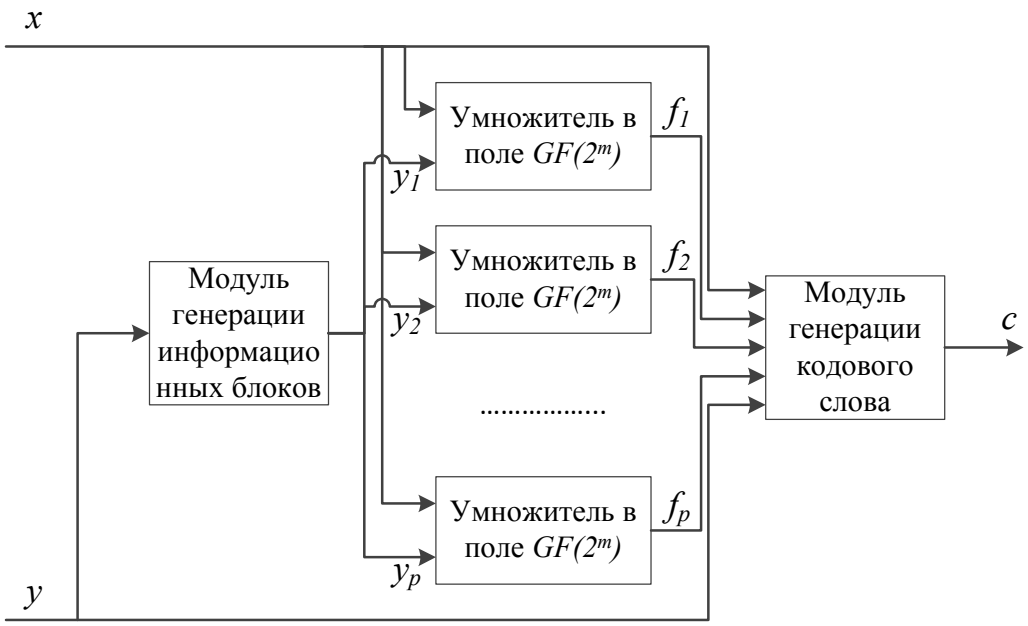


Рисунок 4.5 – Общая схема кодера для модификации AMD кода на основе разбиения информационного сообщения

На схеме изображено p умножителей в поле $GF(2^m)$. Их количество может варьироваться от 1 до u, в зависимости от требуемой производительности. При $p < u$ умножители будут использоваться повторно для различных y_i, что приведёт к увеличению временных затрат кодирования, но уменьшит аппаратную избыточность. При $p = u$ сложность реализации кодера будет максимальна и составит порядка $u \cdot O(m^2)$, в то время как в других конструкциях, где умножение выполняется в $GF(2^k)$, она имеет порядок $O(k^2)$, что сопоставимо со сложностью $u^2 \cdot O(m^2)$ данной модификации.

4.4 Код на основе операции скалярного умножения компонентов информационного сообщения и значения случайной величины

4.4.1 Конструкция

Данная глава диссертационной работы завершается описанием конструкции сильно защищённого AMD кода в узком смысле, основанного на операции скалярного умножения [4]. Эта конструкция обеспечивает минимальную вероятность обнаружения искажений при заданном размере проверочной части кодового слова и обладает меньшей вычислительной сложностью, чем существующие конструкции.

Рассмотрим код

$$C = \{(y \in GF(2^{sr}) | x \in GF(2^{sr}) | f(y,x) \in GF(2^r))\},$$

где $f(x,y) = \sum_{i=1}^{s} x_i \cdot y_i$. Таким образом, сообщение y и значение случайной величины x рассматриваются как вектора, компоненты x_i, y_i которых лежат в конечном поле $GF(2^r)$. В качестве кодирующей функции используется результат скалярного произведения данных векторов.

Утверждение 4.7. *Данный код является AMD кодом в узком смысле и обеспечивает вероятность необнаружения искажений, описываемых моделью алгебраических манипуляций, равную* $P_{undet} = 2^{-r}$.

Доказательство. Как следует из (2.4), обнаруживающая способность AMD кода определяется максимальным количеством корней его УМО относительно x среди всех y и e. Рассмотрим синдромное выражение предлагаемой кодовой конструкции:

$$S = f(y,x) + f(y + e_y, x + e_x) + e_f =$$

$$\sum_{i=1}^{s} x_i \cdot y_i + \sum_{i=1}^{s} (x_i + e_{x_i}) \cdot (y_i + e_{y_i}) + e_f =$$

$$\sum_{i=1}^{s} (x_i \cdot e_{y_i} + y_i \cdot e_{x_i} + e_{x_i} \cdot e_{y_i}) + e_f.$$

Приравняв синдром нулю и перенеся все свободные члены в правую часть равенства, получаем следующее УМО:

$$\sum_{i=1}^{s} x_i \cdot e_{y_i} = \sum_{j=1}^{s} (y_j \cdot e_{x_j} + e_{x_j} \cdot e_{y_j}) + e_f. \tag{4.20}$$

УМО является уравнением первой степени от, в общем случае, s переменных $x_i \in GF(2^r)$, $i = 1, \ldots, s$. Фактическое количество переменных x_i, от которых зависит УМО, определяется количеством ненулевых e_{y_i}. Обозначим это количество через w. Утверждается, что данный код

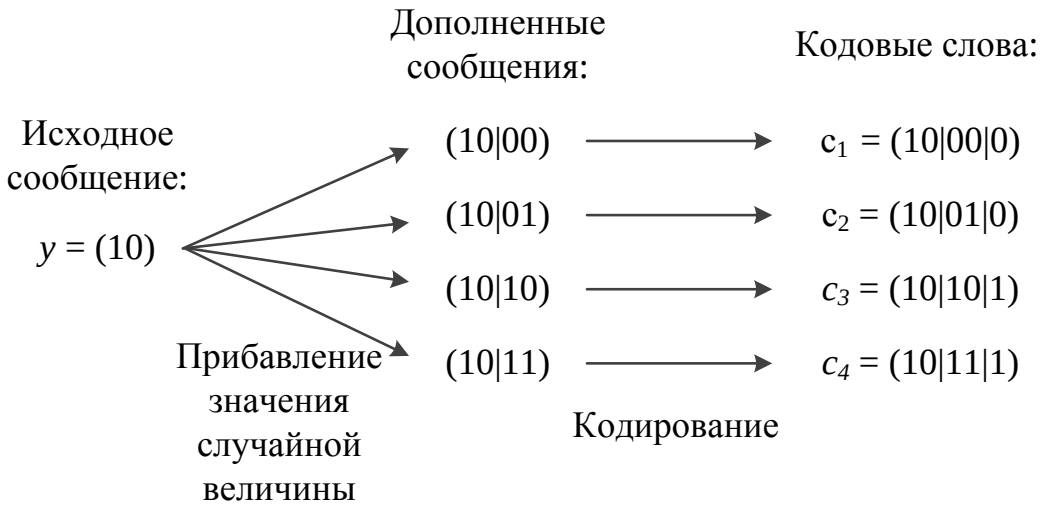


Рисунок 4.6 – Принцип кодирования AMD кодом на основе операции скалярного умножения с параметрами $k = m = 2$, $r = 1$ на примере сообщения (10)

является AMD кодом в узком смысле, следовательно, он обязан обнаруживать ошибки с $e_y \neq 0$. Таким образом, полагаем, что как минимум один e_{y_i} не равен нулю, то есть $1 \leq w \leq s$.

Очевидно, что не существует такой пары y и $e : e_y \neq 0$, при которой в (4.20) выполнится равенство при всех значения случайной величины x. Для фиксированных y и $e : e_y \neq 0$ правая часть выражения (4.20) представляет собой некоторую константу. Легко заметить, что левая часть выражения при условии, что хотя бы один $e_{y_i} \neq 0$, не может быть равна этой константе при всех значениях x. Из этого следует то, что данный код является AMD кодом в узком смысле в соответствии с определением (2.3).

Далее, исследуем количество решений УМО. Известно, что линейное уравнение от w переменных, каждая из которых принимает одно из 2^r значений, имеет ровно $2^{r(w-1)}$ корней. Таким образом, при $2^{r(w-1)}$ значениях случайной величины x синдром принятой последовательности будет равен нулю, и ошибка не будет обнаружена. Отсюда получаем, что вероятность успешного проведения слабой и сильной манипуляций равна $P_{undet} = 2^{r(w-1)}/2^{wr} = 2^{-r}$. ■

Продемонстрируем принцип кодирования AMD кодом на основе операции скалярного умножения компонентов информационного сообщения и значения случайной величины. Пусть $k = m = 2$, $r = 1$, а информационное сообщение $y \in GF(2^2)$ приняло значение (10). Кодирующая функция имеет вид $f(x, y) = x_1y_1 + x_2y_2$, $x_i, y_i \in GF(2)$, $i = 1, 2$. На рисунке 4.6 представлены все соответствующие ему дополненные сообщения и кодовые слова. Информационная, случайная и проверочная части кодового слова разделены символом «|», соответственно. Вычисления проверочной части выполнены в поле Галуа $GF(2)$.

Рассмотрим пример. Пусть $y \in GF(2^{32})$, $x \in GF(2^{32})$, $r = 8$ бит. Функция кодирования имеет следующий вид: $f(y, x) = x_1y_1 + x_2y_2 + x_3y_3 + x_4y_4$, $x_i, y_i \in GF(2^8)$. Скорость кода равна $k/(k + m + r) \approx 0.44$. Использование предлагаемого кода позволяет обеспечить вероятность необнаружения манипуляций, равную $P_{undet} = 2^{-8} \approx 4 \cdot 10^{-3}$.

AMD код, основанный на операции умножения (раздел 4.3.1 и [19]) информационного и случайного компонентов, является частным случаем данной конструкции при $s = 1$ ($k = m = r$) и скорости кода $1/3$.

Стоит отметить, что может быть использована модификация данного кода, основанная на расширении случайного числа (раздел 4.3.2). При этом размер случайной величины может быть уменьшен до $s \cdot h$ бит, $h < r$, что приведёт к вероятности необнаружения ошибки $P_{undet} \leq 2^{-h}$. Данная модификация позволяет увеличивать скорость кода за счёт уменьшения вероятности выявления ошибок, а также обеспечивать защиту данных различных размеров при условии фиксированного размера случайного числа.

Приведём пример модификации предлагаемого кода. Пусть $y \in GF(2^{32})$, $r = 8$ бит. Величину h выберем равную 5 битам, следовательно, $x \in GF(2^{20})$. При кодировании значение случайной величины x будет разбито на 4 блока $x_i \in GF(2^5)$, каждый из которых будет расширен за счёт дополнения нулями до размера, равного 8 битам. Функция кодирования аналогична функции из предыдущего примера. Скорость кода в этом случае равна $32/(32 + 20 + 5) \approx 0.53$. Вероятность необнаружения алгебраической манипуляции составляет $P_{undet} = 2^{-5} \approx 3 \cdot 10^{-2}$.

4.4.2 Сравнение с основными существующими конструкциями

Конструкция на основе кодов РМ

Минимальная вероятность необнаружения ошибок, обеспечиваемая AMD кодами на основе кодов РМ, равна $P_{undet} \leq (b + 1)/2^r$, при этом функция кодирования имеет вид: $f(y, x) = xy_1 + \cdots + x^b y_b + x^{b+2}$, где $b = k/m$, $r = m$. Очевидно, что предлагаемый код обеспечивает меньшую вероятность необнаружения алгебраических манипуляций (вероятность вплоть до в $b + 1$ раз меньше). Кроме того, функция кодирования имеет первую степень, что гарантирует меньшую вычислительную сложность процедур кодирования и декодирования, чем конструкция на основе кодов РМ (минимальная степень функции кодирования которых равна трём).

Необходимо отметить, что конструкция на основе кодов РМ является AMD кодом в широком смысле, обеспечивая обнаружения ошибок во всём кодовом слове. Кроме того, эта конструкция использует, в общем случае, случайное число меньшего размера.

Обобщённые систематические надёжные коды

ОСН коды являются AMD кодами в широком смысле и обеспечивают вероятность необнаружения искажений $P^s_{undet} \leq 2^{1-m}$, при этом размер проверочной части кодового слова равен $r = k + m$ (раздел 4.1). При $s = 1$ ($k = m = r$) предлагаемый код гарантирует меньшую вероятность необнаружения помех $P_{undet} = 2^{-m}$, при этом размер проверочной части кодового слова равен $r = k = m$. При $s > 1$ предлагаемая конструкция требует большего размера случайного числа для достижения заданной P_{undet}, но при этом обеспечивает меньший размер r проверочной части кодового слова.

Стоит отметить, что ОСН коды обеспечивают более низкую вероятность необнаружения слабых атак $P^{w}_{undet} \leq 2^{1-r}$ и имеют вычислительно эффективные алгоритмы исправления ошибок.

Конструкция на основе линейных помехоустойчивых кодов

Код на основе ЛПК является AMD кодом в узком смысле и обеспечивает $P_{undet} = 2^{-m}$ при $k = m = r$ [27]. Данные характеристики достигаются предлагаемой конструкцией при $s = 1$. В отличие от конструкции на основе ЛПК, предлагаемый код позволяет варьировать размер проверочной части r, скорость кода и, соответственно, обеспечиваемую P_{undet} при $s > 1$. Кроме того, использование метода расширения случайной величины снимает ограничение на её размер (то есть $m \leq k$ вместо $m = k$).

4.4.3 Заключение по кодовой конструкции

Нелинейный кодовый метод на основе предлагаемого кода обеспечивает минимальную вероятность необнаружения искажений среди существующих AMD кодов, равную $P_{undet} = 2^{-r}$. Кроме того, метод также характеризуется уменьшенной вычислительной сложностью по сравнению со многими используемыми кодами.

Стоит также отметить, что возможно использование модификации данной конструкции, основанной на расширении случайной величины (раздел 4.3.2). Модификация позволяет варьировать скорость кода и вероятность пропуска ошибок P_{undet}, и может быть использована для кодирования данных различного размера при фиксированном размере случайного числа. Таким образом, использование данного метода повышения помехоустойчивости является эффективным и позволяет повысить надёжность и качество функционирования технических систем.

4.5 Выводы

В представленной главе диссертационной работы были приведены как новые конструкции нелинейных кодов, разработанные автором диссертации, так и модификации существующих. Использование предлагаемых методов обеспечения целостности данных позволяет во многих случаях уменьшить вероятность обнаружения алгебраических манипуляций по сравнению с существующими кодами, а также уменьшить вычислительную сложность процедур кодирования и декодирования. Кроме того, использование ОСН кодов позволяет как значительно увеличить вероятность выявления искажений, соответствующих слабым манипуляциям, так и применить вычислительно простые алгоритмы исправления ошибок малой кратности и повторяющихся ошибок. Использование гибридного кодека, включающего в себя заданное количество ОСН кодов, приводит к значительному уменьшению требуемой аппаратной избыточности.

Данные по параметрам AMD кодов и их особенностям представлены в сводной таблице 4.3. Цветом выделены коды и их модификации, предлагаемые в диссертационной работе. В случае, если P^{w}_{undet} и P^{s}_{undet} совпадают, то в соответствующей графе указывается одно значение. Буквами

«У» и «Ш» в графе «Тип кода» обозначены коды в узком и широком смыслах, соответственно. Необходимо отметить, что для конструкции на основе кодов РМ приведена наименьшая достижимая вероятность пропуска ошибки P_{undet}. Она справедлива лишь при определённом выборе параметров, в общем случае вероятность пропуска ошибки выше (раздел 2.1.4). Под сложностью понимается вычислительная сложность реализации процедур кодирования и декодирования.

Таблица 4.3 – Характеристики конструкций AMD кодов с $y \in GF(2^k)$, $x \in GF(2^m)$, $f(x,y) \in GF(2^r)$

Конструкция AMD кода	Тип кода	Параметры	P^w_{undet} P^s_{undet}	Особенности
на основе кодов РМ	Ш	$k \geq m \geq r$	2^{-r+1}	исправление ошибок; гибкость в выборе параметров; высокая/средняя сложность;
ОСН коды (перестановки)	Ш	$k > 1$, $m \geq 0$, $r = k + m$	2^{-r+1} 2^{-m+1}	исправление ошибок; гибридный кодек; низкая вероятность P^w_{undet}; высокая/средняя сложность; гибкость в выборе параметров;
ОСН коды (скалярн. умн.)	Ш	$k = ta$, $m = tb$, $r = a + b$	2^{-r+1} 2^{-b+1}	гибкость в выборе параметров; низкая вероятность P^w_{undet}; высокая/средняя сложность;
на основе операции умножения	У	$k = m = r$	2^{-m}	отсутствие гибкости в выборе параметров; средняя сложность;
на основе ЛПК	У	$k = m = r$	2^{-m}	отсутствие гибкости в выборе параметров; средняя сложность;
модификации кода на основе операции умножения	У	$k = r \geq m$	2^{-m}	гибкость в выборе параметров; средняя сложность;
на основе операции скалярного умножения	У	$k = sr$, $m = sh$, $h \leq r$	2^{-r}	гибкость в выборе параметров; низкая сложность.

5 Научно–технические предложения по применению нелинейных кодовых методов

5.1 Области применения исследуемых нелинейных кодов

Описанные в диссертационной работе нелинейные кодовые методы применяются:

– для проектирования помехозащищённой памяти, используемой для хранения информации, наличие искажений в которой является критичным (например, при хранении секретных ключей криптографических алгоритмов или управляющих команд) [25, 29, 83];

– для проектирования аппаратных схем, устойчивых к привносимым помехам (например, криптографических модулей в виду угрозы, представляемой вычислительными ошибками) [25, 29, 83];

– в системах передачи данных с обратной связью [15];

– для организации секретной передачи сообщений [27];

– в надёжных нечётких экстракторах [19];

– в системах надёжного разделения секрета [19];

– при организации анонимной квантовой связи [99].

5.2 Предложения по применению разработанных методов

Далее представлено два предложения по использованию методов повышения помехоустойчивости на основе кодов, обнаруживающих алгебраические манипуляции:

– для повышения надёжности информации, обрабатывающейся бортовыми вычислительными комплексами космических аппаратов;

– для защиты реализации шифра AES от привносимых помех.

5.2.1 Повышение достоверности данных в космических аппаратах

Как уже было сказано, длительное нахождение космических аппаратов на орбите (10–15 лет) может приводить к непредсказуемым искажениям в работе бортовых вычислительных комплексов. Для их защиты от помех зачастую используются методы линейного помехоустойчивого кодирования и дублирование. Например, в спутниках FASat–Bravo, Thai Phutt и UoSAT используется тройное модульное резервирование статического оперативного запоминающего устройства (СОЗУ) [100]. При чтении данных все три модуля памяти задействованы, прочитанные значения сравниваются между собой, и по мажоритарному принципу принимается решение о значении выходного байта. Таким образом, за счёт использования 200% структурной избыточности достигается гарантированное исправление однократных ошибок. Кроме того, данная схема предлагает надёжную защиту от многократных ошибок в одном из модулей памяти.

Однако, дублирование уязвимо к алгебраическим манипуляциям (раздел 2.1.1). Более того, в отчёте Суррейского космического центра приводится информация о том, что среди многократных ошибок часто встречается инверсия битов на одной и той же логической позиции у отдельных байтов [100]. Предположительно, это связано с тем, что ячейки, находящиеся относительно близко, подвергаются воздействию тяжелых заряженных частиц, «пролетающих» космический аппарата насквозь. Соответственно, при относительно высокой вероятности таких ошибок тройное дублирование может оказаться не совсем эффективно. Легко заметить, что, рассматривая данную архитектуру организации памяти относительно модели канала с алгебраическими манипуляциями, количество необнаруживаемых конфигураций ошибок из $2^{3*8} - 1$ возможных равно $2^8 - 1$.

Более того, уменьшение размеров транзисторов и снижение рабочего напряжения также значительно увеличивает вероятность появления ошибок высокой кратности. К примеру, в [101] описывается тот факт, что для СОЗУ, произведённой по технологии 65 нм, вероятность многобитовой ошибки в 10 раз превышает аналогичную вероятность для СОЗУ по технологии 90 нм (около 55% ошибок, вызванных бомбардировкой нейтронами, имели кратность два и выше). Кроме того, в качестве минусов использования тройного дублирования можно также указать значительное увеличение энергопотребления, что является весьма критичным параметром для некоторых космических аппаратов (например, малых спутников).

Устранить недостатки защиты СОЗУ с помощью тройного дублирования можно, применив коды, обнаруживающие алгебраические манипуляции. Рассмотрим возможность применения ОСН кодов на основе перестановок в конечном поле для защиты 8–битного СОЗУ, используемого на перечисленных выше спутниках. Использование именно ОСН кодов позволяет нам сохранить возможность гарантированного исправления однократной ошибки, а также обеспечивает относительно высокую вероятность исправления многократных ошибок в отдельных частях кодового слова (информационной, случайной, проверочной). Кроме того, возможна реализация алгоритма исправления повторяющихся ошибок, представляющихся весьма вероятными для данной постановки задачи. Метод построения кода и алгоритмы декодирования описаны в разделе

4.1.2. Напомним, что используемый код имеет следующую структуру:

$$C = \{(y|x|f(x,y) = (x|y)^{-1}|p(y) = wt(y) \bmod 2)\}.$$

В таблице 5.1 представлены параметры ОСН кодов, исправляющих однократную ошибку, которые могут быть использованы в качестве альтернативы тройному дублированию. Представлены те коды, избыточность которых не превосходит избыточности тройного дублирования (200%).

Таблица 5.1 – Параметры ОСН кодов, исправляющих однократную ошибку, для применения в СОЗУ малых спутников

Номер кода	k (бит)	m (бит)	r (бит)	Избыточность	P^w_{undet}	P^s_{undet}
1	8	0	9	112,5%	$2^{-6} \approx 2 \cdot 10^{-2}$	1
2	8	1	10	137,5%	$2^{-8} \approx 4 \cdot 10^{-3}$	1
3	8	2	11	162,5%	$2^{-8} \approx 4 \cdot 10^{-3}$	0.5
4	8	3	12	187,5%	$2^{-10} \approx 1 \cdot 10^{-4}$	0.25

Из таблицы видно, что использование предлагаемых кодов позволит обеспечить вероятность необнаружения любых искажений в диапазоне от 2^{-6} до 2^{-10} для слабых манипуляций и достичь вероятности обнаружения 0.25 для сильных при меньшей избыточности, чем требует тройное дублирование. При этом сохранится возможность исправления однобитовых ошибок и появится возможность исправления повторяющихся ошибок.

5.2.2 Защита архитектуры шифра AES от вычислительных ошибок

Ниже будут приведены предложения по реализации архитектуры шифра AES–128 (AES с длиной ключа 128 бит), устойчивой к привносимым помехам. Компоненты аппаратной реализации шифра защищены с помощью предлагаемых в диссертации AMD кодов, что обеспечивает целостность результатов вычислений и значения ключа шифрования.

Напомним, что шифрование в AES состоит из 10 раундов преобразований над данными размера 128 бит, последний раунд имеет на одно преобразование меньше, а первому раунду предшествует прибавление ключа раунда (Round key addition) (раздел 1.2.3). В каждом из 9 одинаковых раундов осуществляются 4 преобразования: подстановка (SBox), сдвиг строк (Shift Rows), перемешивание столбцов (Mix Columns) и прибавление ключа раунда (Add Round Key). Последний раунд отличается от остальных тем, что в нём нет перестановки столбцов. Блок подстановки фактически состоит из двух преобразований: вычисления обратного мультипликативного элемента в $GF(2^8)$ и афинного преобразования, которое включает умножение на матрицу над $GF(2)$ с последующим прибавлением константного вектора. За исключением операции обращения элемента в конечном поле все операции являются линейными.

Рассматривая один раунд, можно заметить, что 128–битный блок обработки данных может быть разделен на 4 идентичных блока размерности 32 бита. Кроме того, в каждом из этих блоков

вычисление обратного элемента вычисляется над блоком данных размера 8 бит. Таким образом, нелинейная часть преобразований одного раунда может быть представлена в виде 16 идентичных блоков, а линейная — в виде 4 блоков (рисунок 5.1).

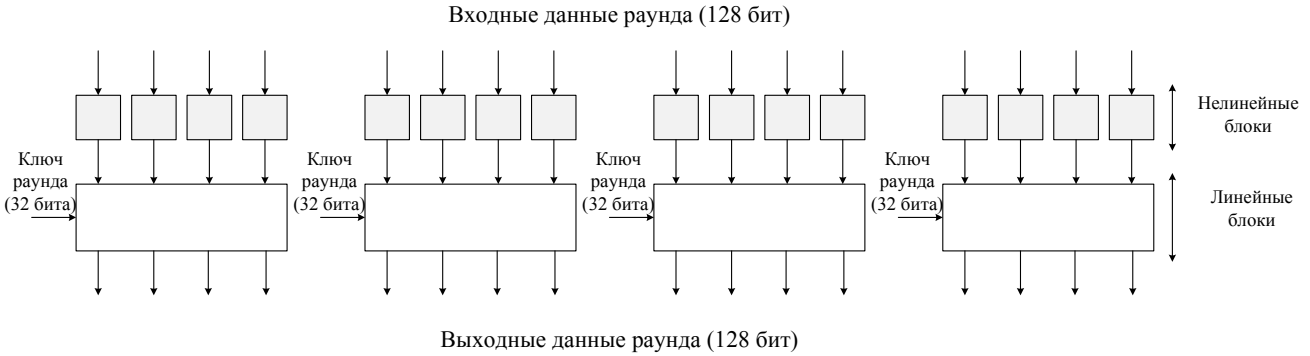


Рисунок 5.1 – Разделение раунда AES на линейные и нелинейные блоки

Защита нелинейных блоков AES может быть осуществлена, например, с использованием изящного метода, предложенного профессором М. Карповским в [28]. Он основан на почти совершенной нелинейности операции инвертирования в поле и обеспечивает заданный уровень обнаружения слабых манипуляций. Для защиты нелинейных блоков от сильных манипуляций рекомендуется использовать физические методы, например, экранирование и установку сенсоров воздействий. Размер нелинейных блоков относительно мал по сравнению с линейной частью шифра, поэтому затраты на внешнюю защиту этих блоков будут несущественными.

Защита линейной части шифра AES осуществляется с помощью кодов, обнаруживающих алгебраические манипуляции. Пусть требуется обеспечить целостность данных с вероятностью необнаружения сильной манипуляции порядка 10^{-3}. Рассмотрим защиту линейных блоков с применением двух кодовых конструкций, предлагаемых в данной диссертационной работе: ОСН кодов и кодов, основанных на операции скалярного умножения компонентов информационного сообщения и значения случайной величины.

Рассмотрим вариант использования ОСН кодов для обеспечения целостности результатов вычислений линейных блоков шифра AES. Каждый из 4 32–битных блоков линейных преобразований защищается с помощью ОСН кодов. Используется код со следующими параметрами: размер информационного сообщения $k = 32$ бита, размер случайного числа $m = 11$ бит, в качестве кодирующей функции используется возведение в третью степень в конечном поле: $f(x, y) = (y|x)^3$. Для уменьшения вносимой избыточности используется модификация кода, основанная на разбиении информационного сообщения на 2 блока по 16 бит. Таким образом, на основе одного случайного элемента $x \in GF(2^{11})$ осуществляется кодирование двух блоков $y_1, y_2 \in GF(2^{16})$ с вычислением проверочных символов $(y_i|x)^3 \in GF(2^{27})$, $i = 1, 2$. Используемый код может быть представлен следующим образом:

$$C = \{(y \in GF(2^{16})|x \in GF(2^{11})|(y|x)^3 \in GF(2^{27}))\}.$$

Вероятность необнаружения искажений, описываемых слабой моделью манипуляций, с помощью данного кода составляет $P^{w}_{undet} \leq 2/2^{27} = 2^{-26} \approx 1.5 \cdot 10^{-8}$, вероятность обнаружения искажений, описываемых сильной моделью, — $P^{s}_{undet} \leq 2/2^{11} = 2^{-10} \approx 10^{-3}$. Схема защиты 32–битного линейного блока шифра AES с помощью ОСН кода приведена на рисунке 5.2.

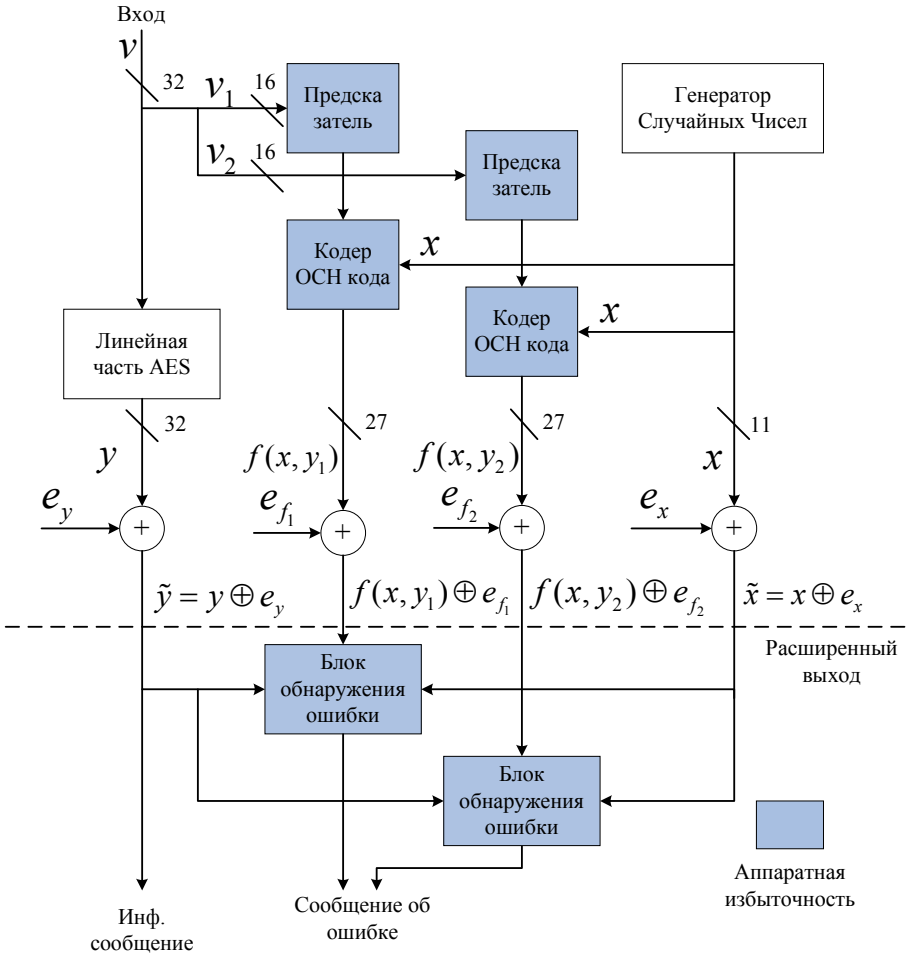


Рисунок 5.2 – Схема параллельного обнаружения ошибок в линейной части шифра AES с использованием ОСН кодов

Рассмотрим вариант использования AMD кода, основанного на операции скалярного умножения компонентов информационного сообщения и значения случайной величины, для обеспечения целостности результатов вычислений линейных блоков шифра AES. Защита блоков линейных преобразований может быть реализована различным образом в зависимости от ограничений на вносимую избыточность и требуемый уровень обнаружения искажений. Блоки могут быть защищены как одним кодом размерности 128 бит, так и кодами меньшей размерности, защищающими пары блоков или отдельные блоки. Параметры используемых при этом кодов приведены в таблице 5.2.

Таблица 5.2 – Варианты параметров AMD кодов, основанных на операции скалярного умножения, используемых для защиты блоков линейных преобразований шифра AES, где R — скорость используемого кода $(R = (k/(k+m+r)))$, N — количество кодеков, r^* — общее количество проверочных символов, а R^* — общая скорость итоговой помехоустойчивой схемы $(R^* = 128/(128 + r^*))$

k (бит)	m (бит)	r (бит)	R	$P^w_{undet} = P^s_{undet}$	N	r^* (бит)	R^*
128	128	32	0.44	$2^{-32} \approx 2.3 \cdot 10^{-10}$	1	160	0.44
128	128	16	0.47	$2^{-16} \approx 1.5 \cdot 10^{-5}$	1	144	0.47
128	128	8	0.49	$2^{-8} \approx 4 \cdot 10^{-3}$	1	136	0.49
64	64	32	0.4	$2^{-32} \approx 2.3 \cdot 10^{-10}$	2	128	0.5
64	64	16	0.44	$2^{-16} \approx 1.5 \cdot 10^{-5}$	2	96	0.57
64	64	8	0.47	$2^{-8} \approx 4 \cdot 10^{-3}$	2	80	0.62
32	32	32	0.33	$2^{-32} \approx 2.3 \cdot 10^{-10}$	4	160	0.44
32	32	16	0.4	$2^{-16} \approx 1.5 \cdot 10^{-5}$	4	96	0.57
32	32	8	0.44	$2^{-8} \approx 4 \cdot 10^{-3}$	4	64	0.67
16	16	16	0.33	$2^{-16} \approx 1.5 \cdot 10^{-5}$	8	144	0.47
16	16	8	0.4	$2^{-8} \approx 4 \cdot 10^{-3}$	8	80	0.62
8	8	8	0.33	$2^{-8} \approx 4 \cdot 10^{-3}$	16	136	0.49

Общее количество проверочных символов r^* подсчитывается по следующему принципу. Количество кодеков, указанное в таблице, говорит о том, что исходный 128–битный блок разбивается на количество подблоков, равное количеству требуемых кодеков, каждый из которых будет кодироваться и декодироваться отдельным кодеком. При этом каждый кодек будет вычислять проверочные символы размера r бит, следовательно, для вычисления всей информационной избыточности, необходимо размер проверочных символов умножить на количество кодеков. Кроме того, к размеру вносимой избыточности необходимо прибавить размер случайного вектора. В этом случае умножение на количество кодеков не требуется, так как один случайный вектор размера m бит может быть использован во всех кодеках. Второй характеристикой данных вариантов является сложность реализации кодеков. Известно, что сложность реализации операции умноже-

ния в поле пропорциональная квадрату степени расширения поля, поэтому, чем меньше размер проверочной функции, тем меньше аппаратной избыточности потребуется на реализацию кодека. Кроме того, сложность реализации предлагаемой схемы помехоустойчивого кодирования зависит также и от количества блоков умножения в конечном поле, которое равно величине $128/r$. Количество блоков умножения может быть снижено за счёт их переиспользования при реализации многотактовой схемы кодирования/декодирования.

Защищенная архитектура линейных блоков AES с использованием AMD кода, основанного на операции скалярного умножения, с параметрами $k = 128$, $m = 128$, $r = 32$ приведена на рисунке 5.3.

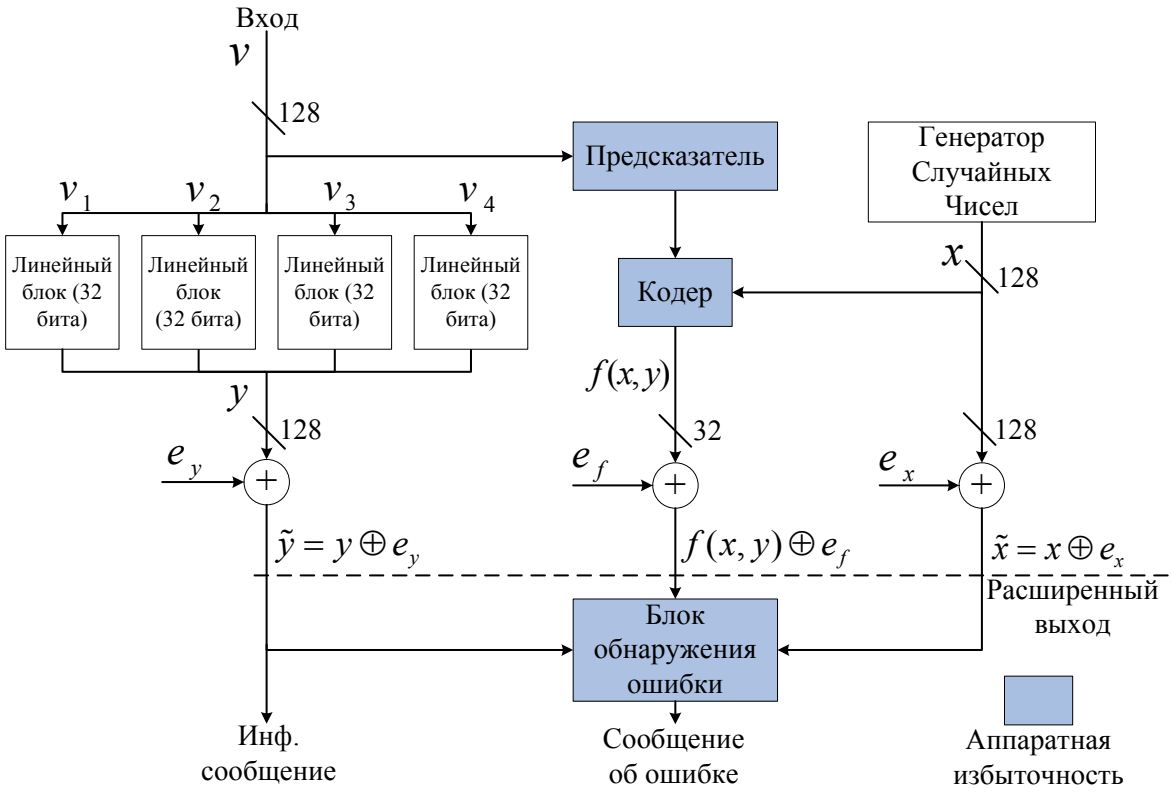


Рисунок 5.3 – Схема параллельного обнаружения ошибок в линейной части шифра AES с использованием AMD кода на основе операции скалярного умножения компонентов информационного сообщения и значения случайной величины

Помимо защиты аппаратной схемы, реализующей шифр AES, также необходимо обеспечить целостность секретного ключа, хранящегося в памяти устройства. Для этого требуется использование защищённой памяти, устойчивой привнесению помех. Рассмотрим архитектуру помехоустойчивой памяти на основе ОСН кодов.

Размер секретного ключа равен $k = 128$ битам. Размер случайного числа $m = 11$ бит. В качестве кодирующей функции используется вычисление обратного элемента по умножению в конечном поле: $f(y, x) = (y|x)^{-1}$. Для уменьшения избыточности используется модификация ОСН кода на основе разбиения информационного сообщения на 8 блоков $y_i \in GF(2^{16})$, $i = 1, ..., 8$. Таким образом, в процессе кодирования на основе одного случайного числа x вычисляется 8

проверочных символов для 8 блоков информационного сообщения y_i: $(y_i|x)^{-1} \in GF(2^{27})$. Кроме того, для каждой конкатенации $(y_i|x)$ и для каждого проверочного символа $(y_i|x)^{-1}$ вычисляется бит проверки на чётность. Таким образом, расстояние кода, используемого для кодирования каждого информационного блока, равно 4. Итоговые параметры кода следующие: $k^* = 128$ бит, $m^* = 11$ бит, $r^* = 232$ бита. Код может быть представлен следующим образом:

$$C = \{(y \in GF(2^{16})|x \in GF(2^{11})|(y|x)^{-1} \in GF(2^{27})|p(y_i|x) \in GF(2)|p((y_i|x)^{-1}) \in GF(2))\}.$$

Итоговая скорость кода приблизительно равна $1/3$. Схема защищённой архитектуры блока памяти приведена на рисунке 5.4.

Данная схема защиты памяти гарантирует вероятность необнаружения слабых манипуляций $P^w_{undet} \leq 2^{-26} \approx 1.5 \cdot 10^{-8}$, сильных — $P^s_{undet} \leq 2^{-10} \approx 10^{-3}$. При этом, в каждом из 8 блоков y_i гарантируется исправление однократной ошибки и обнаружение двукратной ошибки. Таким образом, код используется как SEC–DED код, что повышает достоверность ключа шифрования. При условии, если запись и чтение ключа могут осуществляться в течение нескольких тактов, количество кодеков может быть сокращено за счёт повторного использования одного кодека на каждом такте.

Проведённые расчёты показывают, что предлагаемая защищённая архитектура шифра AES обеспечивает целостность данных с вероятностью необнаружения слабых алгебраических манипуляций $P^w_{undet} \approx 1.5 \cdot 10^{-8}$, сильных — $P^s_{undet} \approx 10^{-3}$. При этом обеспечивается исправление ошибок малой кратности в ключе шифрования.

5.3 Выводы

Проведённые расчёты показали, что использование предлагаемых методов помехоустойчивого кодирования позволяет гарантировать заданный уровень надёжности обработки и хранения информации при условии непредсказуемых потоков ошибок, характерных для данных приложений исследуемой модели канала. Таким образом, за счёт использования предлагаемых методов нелинейного кодирования задача повышения надёжности и качества функционирования технических систем успешно решена.

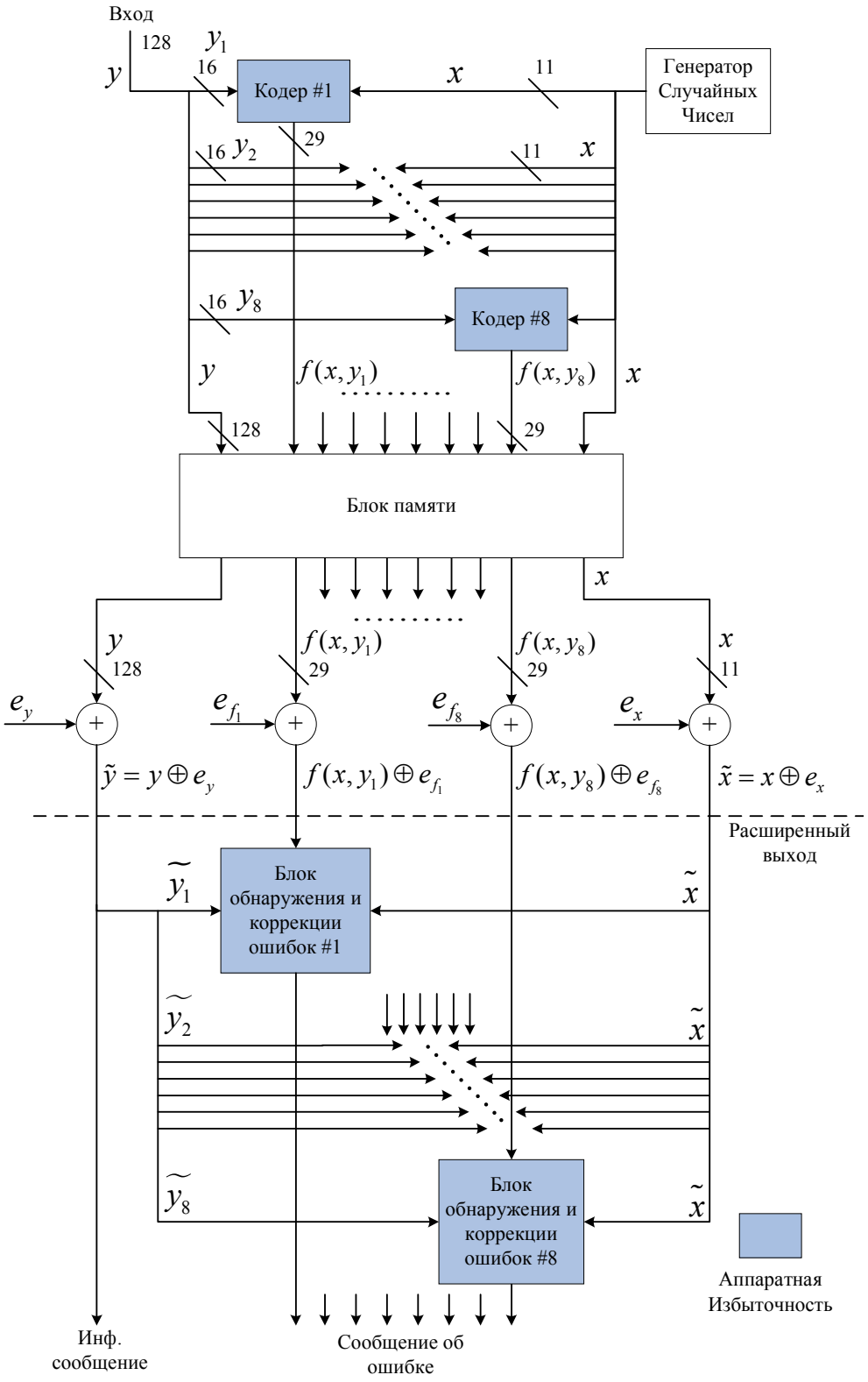


Рисунок 5.4 – Архитектура помехоустойчивой памяти с использованием ОСН кода. Блок кодера выполняет вычисление как проверочных символов ОСН кода, так и битов проверки на четность

Заключение

Основные результаты, полученные в диссертационной работе:

1. В работе предложены новые кодовые методы обеспечения целостности данных для каналов, описываемых моделью канала с алгебраическими манипуляциями. Показано, что использование предлагаемых методов и алгоритмов позволяет увеличить вероятность обнаружения манипуляций и уменьшить вычислительную сложность процедур кодирования и декодирования, что, в свою очередь, повышает достоверность обработки и передачи информации вычислительными устройствами.

2. Полученные результаты можно сформулировать следующим образом:

   – Разработка и исследование кодового метода повышения помехоустойчивости на основе класса обобщённых систематических надёжных кодов, обнаруживающих алгебраические манипуляции.

   – Разработка алгоритма обнаружения и исправления ошибок малой кратности с помощью обобщённых систематических надёжных кодов.

   – Разработка и исследование кодового метода повышения помехоустойчивости, основанного на операции скалярного умножения компонентов информационного сообщения и значения случайной величины.

   – Осуществление модификаций кодового метода, основанного на операции умножения информационного и случайного компонентов, с целью уменьшения информационной избыточности.

   – Вывод оценок экстремальных значений параметров нелинейных кодов, обнаруживающих алгебраические манипуляции.

   Таким образом, решены все задачи, поставленные для достижения сформулированной в работе цели.

3. Показано, что преобразование данных с использованием предлагаемых кодовых методов зачастую обеспечивает более высокий уровень защиты от алгебраических манипуляций, нежели использование существующих решений. Кроме того, алгоритмы декодирования некоторых кодов обладают меньшей вычислительной сложностью, что приводит к снижению затрат на их реализацию по сравнению с аналогами. Использование предлагаемых

методов помехоустойчивого кодирования позволяет повысить надёжность обработки информации и качество функционирования технических систем. Таким образом, цель работы достигнута.

4. На основе решённых в диссертации задач можно определить следующие направления дальнейших исследований:

   – Разработка методов построения систематических надёжных кодов, обладающих минимальной избыточностью.

   – Разработка алгоритма исправления ошибок малой кратности для кодов, основанных на операции скалярного умножения компонентов информационного сообщения и значения случайной величины.

   – Исследование возможности построения систем связи с обратной связью на основе надёжных кодов.

   – Разработка и исследование нелинейных кодовых методов с простыми алгоритмами кодирования и декодирования (с низким энергопотреблением) для бортовых систем обработки информации аэрокосмических систем и комплексов.

Список литературы

[1] *Алексеев, М. О.* Нижняя граница длины систематических равномерно надежных кодов / М. О. Алексеев // *Известия ВУЗов. Приборостроение.* — 2013. — август. — № 8. — С. 14–16.

[2] *Алексеев, М. О.* Новая конструкция систематического надежного кода / М. О. Алексеев // *Известия ВУЗов. Приборостроение.* — 2013. — август. — № 8. — С. 24–27.

[3] *Алексеев, М. О.* Об обнаружении алгебраических манипуляций с помощью операции умножения / М. О. Алексеев // *Информационно-управляющие системы.* — 2014. — июнь. — № 3. — С. 103–108.

[4] *Алексеев, М. О.* Защита от алгебраических манипуляций на основе операции скалярного умножения / М. О. Алексеев // *Проблемы информационной безопасности. Компьютерные системы.* — 2014. — № 2. — С. 47–53.

[5] *Громова, А. Н.* Вариант алгоритма нахождения ошибок для БЧХ–кодов / А. Н. Громова, М. О. Алексеев // *Программные продукты и системы. – Тверь: МНИИПУ.* — 2010. — май. — № 2. — С. 56–58.

[6] *Алексеев, М. О.* Об обнаружении ошибок с помощью нелинейных кодов / М. О. Алексеев, Е. Т. Мирончиков // *Научная сессия ГУАП: Сб. докл.: В 3 ч. Ч. I. Технические науки / СПб.: ГУАП.* — 2011. — С. 40–43.

[7] *Алексеев, М. О.* Уточненная нижняя граница обнаруживающей способности amd кодов / М. О. Алексеев // *Научная сессия ГУАП: Сб. докл.: В 3 ч. Ч. I. Технические науки / СПб.: ГУАП.* — 2012. — С. 61–64.

[8] *Алексеев, М. О.* Пакетная передача с кодовым зашумлением. Теоретические основы и практическое применение / М. О. Алексеев. — LAP LAMBERT Academic Publishing, 2012.

[9] *Алексеев, М. О.* Обобщение надёжных кодов / М. О. Алексеев, А. В. Еганян // *СПИСОК–2013: Материалы всероссийской научной конференции по проблемам информатики.* — 2013. — С. 263–268.

[10] *Алексеев, М. О.* О гибридном кодеке, обнаруживающем алгебраические манипуляции / М. О. Алексеев // *Научная сессия ГУАП: Сб. докл.: В 3 ч. Ч. I. Технические науки / СПб.: ГУАП.* — 2013. — С. 3–6.

[11] *Алексеев, М. О.* Об исправлении ошибок малой кратности обобщёнными систематическими надёжными кодами / М. О. Алексеев // *Теория и практика современной науки : материалы XV Междунар. научно–практической конф., г. Москва, 8–9 октября 2014 г. / Науч.–инф. издат. центр «Институт стратегических исследований». – М.: Изд-во «Институт стратегических исследований».* — 2014. — С. 47–53.

[12] *Alekseev, M.* Two Algebraic Manipulation Detection Codes Based on a Scalar Product Operation / M. Alekseev // Proc. of the Workshop on Coding and Cryptography (WCC 2015). — 2015.

[13] *Харкевич, А. А.* Борьба с помехами / А. А. Харкевич. — М.: Физматгиз, 1963. — 267 с.

[14] *MacWilliams, F.J.* The Theory of Error–Correcting Codes / F.J. MacWilliams, N.J.A. Sloane. North–Holland mathematical library. — North–Holland Publishing Company, 1977.

[15] *Коржик, Ф. М.* Помехоустойчивое кодирование дискретных сообщений в каналах со случайной структурой / Ф. М. Коржик, Л. М. Финк. — Связь, 1975. — 272 с.

[16] *Sklar, Bernard*. Digital Communications: Fundamentals and Applications / Bernard Sklar. — Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988.

[17] *Shannon, C. E.* A mathematical theory of communication / C. E. Shannon // *Bell System Technical Journal*. — 1948. — July, October. — Vol. 27. — P. 379–423. http://cm.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf.

[18] *Blackwell, David*. The capacities of certain channel classes under random coding / David Blackwell, Leo Breiman, A. J. Thomasian // *The Annals of Mathematical Statistics*. — 1960. — Vol. 31, no. 3. — P. pp. 558–567.

[19] Detection of algebraic manipulation with applications to robust secret sharing and fuzzy extractors / Ronald Cramer, Yevgeniy Dodis, Serge Fehr et al. // Advances in Cryptology–EUROCRYPT 2008. — Springer, 2008. — P. 471–488. http://link.springer.com/chapter/10.1007/978-3-540-78967-3_27.

[20] *Vasil'ev, J. L.* On nongroup close-packed codes / J. L. Vasil'ev // *Probl. Kiberneti. (in Russian)*. — 1962. — no. 8. — P. 375–378.

[21] *Solov'eva, F. I.* On binary nongroup codes / F. I. Solov'eva // *Methodi Diskr. Analiza (in Russian)*. — 1981. — no. 37. — P. 65–76.

[22] *Phelps, K. T.* A Combinatorial Construction of Perfect Codes / K. T. Phelps // *Siam Journal on Algebraic and Discrete Methods*. — 1983. — Vol. 4.

[23] *Phelps, Kevin T.* Kernels of nonlinear hamming codes / Kevin T. Phelps, Mike LeVan // *Designs, Codes and Cryptography*. — 1995. — Vol. 6. — P. 247–257.

[24] Design of Cryptographic Devices Resilient to Fault Injection Attacks Using Nonlinear Robust Codes / Kahraman D. Akdemir, Zhen Wang, Mark G. Karpovsky, Berk Sunar // Fault Analysis in Cryptography / Ed. by M. Joye, M. Tunstall. — Springer Berlin Heidelberg, 2012. — Information Security and Cryptography. — P. 171–199.

[25] *Karpovsky, Mark*. Design of strongly secure communication and computation channels by nonlinear error detecting codes / Mark Karpovsky, Zhen Wang // *IEEE Transactions on Computers*. — 2013. — Vol. 99, no. PrePrints.

[26] *Wang, Zhen*. Replacing linear Hamming codes by robust nonlinear codes results in a reliability improvement of memories / Zhen Wang, M.G. Karpovsky, K.J. Kulikowski // Dependable Systems & Networks, 2009. DSN '09. IEEE/IFIP International Conference on. — 2009. — P. 514–523.

[27] *Jongsma, E*. Algebraic Manipulation Detection Codes: Ph.D. thesis. — 2008. — 6 maart. https://www.math.leidenuniv.nl/scripties/JongsmaBachelor.pdf.

[28] *Karpovsky, Mark*. Differential fault analysis attack resistant architectures for the advanced encryption standard / Mark Karpovsky, Konrad J Kulikowski, Alexander Taubin // Smart Card Research and Advanced Applications VI. — Springer, 2004. — P. 177–192. http://link.springer.com/chapter/10.1007/1-4020-8147-2_12.

[29] Secure memories resistant to both random errors and fault injection attacks using nonlinear error correction codes / Shizun Ge, Zhen Wang, Pei Luo, Mark Karpovsky // Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy. — HASP '13. — New York, NY, USA: ACM, 2013. — P. 1–8.

[30] *Wang, Zhen*. Reliable MLC NAND flash memories based on nonlinear t-error-correcting codes / Zhen Wang, M. Karpovsky, A. Joshi // Dependable Systems and Networks (DSN), 2010 IEEE/IFIP International Conference on. — 2010. — P. 41–50.

[31] Отчет о научно–исследовательской работе «Разработка и исследование надёжных методов хранения информации в аэрокосмических системах и комплексах» / промежуточный / № госрегистрации 1141031400626 / номер темы С8, код проекта 2716. — Санкт Петербург, 2015, — 164 с.

[32] *Гобчанский, О.* Устойчивость IBM PC совместимых контроллеров к радиационным сбоям на орбитах космических аппаратов / О. Гобчанский, Н. Кузнецов // *Современные технологии автоматизации*. — 2005. — Т. 3. — С. 46–51.

[33] *Shamir, Adi*. How to share a secret / Adi Shamir // *Communications of the ACM*. — 1979. — Vol. 22, no. 11. — P. 612–613.

[34] *Акишин, А. И.* Воздействие окружающей среды на материалы космических аппаратов / А. И. Акишин, Л. С. Новиков. — М.: Знания, 1983. — 64 с.

[35] Микроэлектроника для космоса и военных [Электронный ресурс] : Хабрахабр / Авторы Хабрахабр // Коллективный блог Хабрахабр. – Электрон. дан. — Москва: ООО «Хабр», 2015. — Режим доступа: http://habrahabr.ru/post/156049/.

[36] *Коршунов, Ф. П.* Радиационные эффекты в полупроводниковых приборах / Ф. П. Коршунов, Г. В. Гатальский, Г. М. Иванов. — Минск: Наука и Техника, 1976. — 232 с.

[37] *Вавилов, В. С.* Радиационные дефекты в полупроводниках и полупроводниковых приборах / В. С. Вавилов, Н. А. Ухин. — М.: Атомиздат, 1969. — 312 с.

[38] Characterizing flash memory: Anomalies, observations, and applications / Laura M. Grupp, Adrian M. Caulfield, Joel Coburn et al. // Proceedings of the 42Nd Annual IEEE/ACM International Symposium on Microarchitecture. — MICRO 42. — New York, NY, USA: ACM, 2009. — P. 24–33.

[39] *Blakley, George R.* Safeguarding Cryptographic Keys / George R. Blakley // Proceedings of the 1979 AFIPS National Computer Conference. — Vol. 48. — 1979. — P. 313–317.

[40] *Schneier, Bruce*. Applied Cryptography (2Nd Ed.): Protocols, Algorithms, and Source Code in C / Bruce Schneier. — New York, NY, USA: John Wiley & Sons, Inc., 1995.

[41] Схема разделения секрета Шамира [Электронный ресурс] : Материал из Википедии — свободной энциклопедии / Авторы Википедии // Википедия, свободная энциклопедия. – Электрон. дан. — Сан–Франциско: Фонд Викимедиа, 2015. — Режим доступа: http://ru.wikipedia.org/?oldid=70640381.

[42] *Zhou, YongBin*. Side–channel attacks: Ten years after its publication and the impacts on cryptographic module security testing. / YongBin Zhou, DengGuo Feng // *IACR Cryptology ePrint Archive*. — 2005. — 388 p.

[43] *Anderson, R.* Tamper resistance - a cautionary note / R. Anderson, M. Kuhn // Proc of the 2nd USENIX Workshop on Electronic Commerce. — 1996. — 05. — P. 1–11.

[44] *Anderson, Ross J.* Low Cost Attacks on Tamper Resistant Devices / Ross J. Anderson, Markus G. Kuhn // Security Protocols Workshop / Ed. by Bruce Christianson, Bruno Crispo, T. Mark A. Lomas, Michael Roe. — Vol. 1361 of *Lecture Notes in Computer Science*. — Springer, 1997. — P. 125–136.

[45] Атака по сторонним каналам [Электронный ресурс] : Материал из Википедии — свободной энциклопедии / Авторы Википедии // Википедия, свободная энциклопедия. – Электрон. дан. — Сан–Франциско: Фонд Викимедиа, 2015. — Режим доступа: http://ru.wikipedia.org/?oldid=67028511.

[46] *Anderson, Ross*. Low cost attacks on tamper resistant devices / Ross Anderson, Markus Kuhn // Security Protocols / Springer. — 1998. — P. 125–136.

[47] *Biham, Eli*. Differential fault analysis of secret key cryptosystems / Eli Biham, Adi Shamir // Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology. — CRYPTO '97. — London, UK, UK: Springer–Verlag, 1997. — P. 513–525.

[48] The sorcerer's apprentice guide to fault attacks / H. Bar-El, H. Choukri, D. Naccache et al. // Proceedings of the IEEE. — No. 2. — 2006. — P. 370–382.

[49] *Trichina, E.* Multi fault laser attacks on protected CRT-RSA / E. Trichina, R. Korkikyan // Workshop on Fault Diagnosis and Tolerance in Cryptography. — 2010. — P. 75–86.

[50] Low voltage fault attacks on the RSA cryptosystem / A. Barenghi, G. Bertoni, E. Parrinello, G. Pelosi // Workshop on Fault Diagnosis and Tolerance in Cryptography. — 2009. — P. 23–31.

[51] *Skorobogatov, Sergei*. Optical fault masking attacks / Sergei Skorobogatov // Fault Diagnosis and Tolerance in Cryptography (FDTC), 2010 Workshop on / IEEE. — 2010. — P. 23–29. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5577358.

[52] *Kulikowski, Konrad J*. Comparative analysis of robust fault attack resistant architectures for public and private cryptosystems / Konrad J Kulikowski, Zhen Wang, Mark G Karpovsky // Fault Diagnosis and Tolerance in Cryptography, 2008. FDTC'08. 5th Workshop on / IEEE. — 2008. — P. 41–50.

[53] *Malkin, T.* A comparative cost/security analysis of fault attack countermeasures / T. Malkin, F.-X. Standaert, M. Yung // Fault Diagnosis and Tolerance in Cryptography / Ed. by L. Breveglieri, I. Koren, D. Naccache, J.-P. Seifert. — Springer Berlin / Heidelberg, 2006. — Vol. 4236 of *Lecture Notes in Computer Science*. — P. 159–172.

[54] Design of reliable and secure multipliers by multilinear arithmetic codes / Z. Wang, M. Karpovsky, B. Sunar, A. Joshi // Information and Communications Security. — Vol. 5927 of *Lecture Notes in Computer Science*. — 2009. — P. 47–62.

[55] *Wang, Zhen*. Multilinear codes for robust error detection / Zhen Wang, M. Karpovsky, B. Sunar // On-Line Testing Symposium, 2009. IOLTS 2009. 15th IEEE International. — 2009. — P. 164–169.

[56] *Gaubatz, Gunnar*. Sequential circuit design for embedded cryptographic applications resilient to adversarial faults / Gunnar Gaubatz, Erkay Savas, Berk Sunar // *Computers, IEEE Transactions on*. — 2008. — Vol. 57, no. 1. — P. 126–138. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4358236.

[57] *Skorobogatov, S.P.* Optical Fault Induction Attacks / S.P. Skorobogatov, R. J. Anderson // Cryptographic Hardware and Embedded Systems Workshop (CHES-2002), LNCS 2523. — Springer-Verlag, 2002. — P. 2–12.

[58] *Derouet, Odile*. Secure smartcard design against laser fault injection / Odile Derouet // 4th Workshop on Fault Diagnostic and Tolerance in Cryptography, Vienne, Autriche. — 2007. — P. 87–96.

[59] *Boneh, Dan*. On the importance of checking cryptographic protocols for faults / Dan Boneh, Richard A. DeMillo, Richard J. Lipton // Proceedings of the 16th Annual International Conference on Theory and Application of Cryptographic Techniques. — EUROCRYPT'97. — Berlin, Heidelberg: Springer-Verlag, 1997. — P. 37–51.

[60] *Giraud, Christophe*. DFA on AES / Christophe Giraud // Advanced Encryption Standard - AES, 4th International Conference, AES 2004. — Springer, 2003. — P. 27–41.

[61] *Dusart, Pierre*. Differential fault analysis on aes / Pierre Dusart, Gilles Letourneux, Olivier Vivolo // Applied Cryptography and Network Security / Springer. — 2003. — P. 293–306. http://link.springer.com/chapter/10.1007/978-3-540-45203-4_23.

[62] *Piret, Gilles*. A differential fault attack technique against spn structures, with application to the aes and khazad / Gilles Piret, Jean-Jacques Quisquater // Cryptographic Hardware and Embedded Systems-CHES 2003. — Springer, 2003. — P. 77–88.

[63] *Moradi, Amir*. A generalized method of differential fault attack against aes cryptosystem / Amir Moradi, Mohammad T. Manzuri Shalmani, Mahmoud Salmasizadeh // Cryptographic Hardware and Embedded Systems - CHES 2006, 8th International Workshop. — Vol. 4249 of *Lecture Notes in Computer Science*. — Springer, 2006. — P. 91–100. http://www.iacr.org/cryptodb/archive/2006/CHES/08/08.pdf.

[64] *Bogdanov, Andrey*. Biclique cryptanalysis of the full aes / Andrey Bogdanov, Dmitry Khovratovich, Christian Rechberger // Proceedings of the 17th International Conference on The Theory and Application of Cryptology and Information Security. — ASIACRYPT'11. — Berlin, Heidelberg: Springer-Verlag, 2011. — P. 344–371.

[65] *Chen, Chien N.* Differential fault analysis on AES key schedule and some countermeasures / Chien N. Chen, Sung M. Yen // ACISP 2003 / Ed. by Safavi R. Naini, J. Seberry. — Vol. 2727 of *Lecture Notes in Computer Science*. — Springer-Verlag, 2003. — P. 118–129.

[66] *Takahashi, Junko*. DFA Mechanism on the AES Key Schedule / Junko Takahashi, Toshinori Fukunaga, Kimihiro Yamakoshi // Proceedings of the Workshop on Fault Diagnosis and Tolerance in Cryptography. — FDTC '07. — Washington, DC, USA: IEEE Computer Society, 2007. — P. 62–74.

[67] *Kim, Chong Hee*. Improved differential fault analysis on aes key schedule / Chong Hee Kim // IEEE Transactions on Information Forensics and Security. — Vol. 7. — 2012. — P. 41–50.

[68] *Lenstra, Arjen K.* Memo on RSA signature generation in the presence of faults / Arjen K. Lenstra. — manuscript. http://infoscience.epfl.ch/record/164524/files/nscan20.PDF.

[69] *Sellers, F.F.* Error Detecting Logic for Digital Computers / F.F. Sellers, M. Hsiao, L.W. Bearnson. — McGraw-Hill, 1968.

[70] *Kraft, George D*. Microprogrammed control and reliable design of small computers / George D Kraft, Wing N Toy. — Englewood Cliffs, NJ: Prentice-Hall, 1981.

[71] Fault-tolerance design of the ibm enterprise system/9000 type 9021 processors. / C. L. (Jim) Chen, Nandakumar N. Tendolkar, Arthur J. Sutton et al. // *IBM Journal of Research and Development*. — 1992. — Vol. 36, no. 4. — P. 765–780.

[72] *Pradhan, D. K.* Fault-tolerant Computer System Design / D. K. Pradhan. — Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1996.

[73] *Spainhower, Lisa*. Ibm s/390 parallel enterprise server g5 fault tolerance: A historical perspective. / Lisa Spainhower, Thomas A. Gregg // *IBM Journal of Research and Development*. — 1999. — Vol. 43, no. 5. — P. 863–874.

[74] Concurrent error detection schemes for fault-based side-channel cryptanalysis of symmetric block ciphers. / Ramesh Karri, Kaijie Wu, Piyush Mishra, Yongkook Kim // *IEEE Trans. on CAD of Integrated Circuits and Systems*. — 2002. — Vol. 21, no. 12. — P. 1509–1517.

[75] *Mitra, Subhasish*. Which concurrent error detection scheme to choose? / Subhasish Mitra, Edward J. McCluskey // Proceedings of the 2000 IEEE International Test Conference. — ITC '00. — Washington, DC, USA: IEEE Computer Society, 2000. — P. 985–994.

[76] *Shedletsky, John J.* Error correction by alternate-date retry / John J. Shedletsky // IEEE Transactions on Computers. — Vol. 27. — 1978. — February. — P. 106–112.

[77] *Patel, J. H.* Concurrent error detection in alu's by recomputing with shifted operands / J. H. Patel, L. Y. Fung // *IEEE Trans. Comput.* — 1982. — july. — Vol. 31, no. 7. — P. 589–595.

[78] Concurrent error detection of fault-based side-channel cryptanalysis of 128-bit symmetric block ciphers / Ramesh Karri, Kaijie Wu, Piyush Mishra, Yongkook Kim // Proceedings of the 38th annual Design Automation Conference / ACM. — 2001. — P. 579–584.

[79] Vinci: Secure test of a vlsi high-speed encryption system / H Bonnenberg, Andreas Curiger, Norbert Felber et al. // Test Conference, 1993. Proceedings., International / IEEE. — 1993. — P. 782–790. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=470624.

[80] *Karri, Ramesh*. Parity-based concurrent error detection of substitution-permutation network block ciphers / Ramesh Karri, Grigori Kuznetsov, Michael Goessel // Cryptographic Hardware and Embedded Systems-CHES 2003. — Springer, 2003. — P. 113–124. http://link.springer.com/chapter/10.1007/978-3-540-45238-6_10.

[81] Error analysis and detection procedures for a hardware implementation of the advanced encryption standard / Guido Bertoni, Luca Breveglieri, Israel Koren et al. // *IEEE Transactions on Computers*. — 2003. — Vol. 52.

[82] On the VLSI implementation of the international data encryption algorithm IDEA / S. Wolter, H. Matz, A. Schubert, R. Laur // Circuits and Systems, 1995. ISCAS '95., 1995 IEEE International Symposium on. — Vol. 1. — 1995. — P. 397–400.

[83] *Wang, Zhen*. Algebraic manipulation detection codes and their applications for design of secure cryptographic devices / Zhen Wang, Mark G. Karpovsky // IOLTS. — 2011. — P. 234–239.

[84] *Wang, Zhen*. Design of memories with concurrent error detection and correction by nonlinear sec-ded codes / Zhen Wang, Mark Karpovsky, Konrad J. Kulikowski // *J. Electron. Test.* — 2010. — October. — Vol. 26, no. 5. — P. 559–580.

[85] *Verdel, Thomas*. Duplication-based concurrent error detection in asynchronous circuits: shortcomings and remedies / Thomas Verdel, Yiorgos Makris // Defect and Fault Tolerance in VLSI Systems, 2002. DFT 2002. Proceedings. 17th IEEE International Symposium on / IEEE. — 2002. — P. 345–353.

[86] *Mitra, Subhasish*. Diversity techniques for concurrent error detection / Subhasish Mitra, Edward J. McCluskey // ISQED. — 2001. — P. 249–250. http://csdl.computer.org/comp/proceedings/isqed/2001/1025/00/10250249.pdf.

[87] *Menezes, Alfred J.* Handbook of Applied Cryptography / Alfred J. Menezes, Scott A. Vanstone, Paul C. Van Oorschot. — 1st edition. — Boca Raton, FL, USA: CRC Press, Inc., 1996.

[88] *Тужилин, М. Э.* Почти Совершенные Нелинейные Функции / М. Э. Тужилин // *Прикладная Дискретная Математика*. — 2009. — № 3(5). — С. 14 – 20.

[89] *Nyberg, Kaisa*. Differentially uniform mappings for cryptography / Kaisa Nyberg // Advances in Cryptology - EUROCRYPT'93. — Vol. 765 of *Lecture Notes in Computer Science*. — Springer-Verlag, 1994.

[90] *Carlet, Claude*. Highly Nonlinear Mappings / Claude Carlet, Cunsheng Ding // *Journal of Complexity*. — 2004. — April. — Vol. 20, no. 2-3. — P. 205–244.

[91] *Nyberg, Kaisa*. Provable Security against Differential Cryptanalysis / Kaisa Nyberg, Lars R. Knudsen // Advances in Cryptology - CRYPTO '92 / Ed. by E. F. Brickell. — Lecture

Notes in Computer Science no. 740. — Berlin-Heidelberg-New York: Springer-Verlag, 1993. — P. 566–574.

[92] *Kulikowski, Konrad J.* Robust correction of repeating errors by non-linear codes / Konrad J. Kulikowski, Mark G. Karpovsky // *IET Communications*. — 2011. — P. 2317–2327.

[93] *Jungnickel, Dieter*. Difference Sets: An Introduction / Dieter Jungnickel, Alexander Pott // Difference Sets, Sequences and their Correlation Properties / Ed. by A. Pott, P.V. Kumar, T. Helleseth, D. Jungnickel. — Springer Netherlands, 1999. — Vol. 542 of *NATO Science Series*. — P. 259–295.

[94] *Beth, Thomas*. Design theory / Thomas Beth, Dieter Jungnickel, Hanfried Lenz. — Cambridge University Press, 1999. — Vol. 69.

[95] *Kulikowski, Konrad J.* Comparative analysis of robust fault attack resistant architectures for public and private cryptosystems. / Konrad J. Kulikowski, Zhen Wang, Mark G. Karpovsky // FDTC / Ed. by Luca Breveglieri, Shay Gueron, Israel Koren et al. — IEEE Computer Society, 2008. — P. 41–50.

[96] *Kulikowski, K.J.* Robust correction of repeating errors by non-linear codes / K.J. Kulikowski, M.G. Karpovsky // *IET Communications*. — 2011. — November. — Vol. 5. — P. 2317–2327(10).

[97] *Wang, Zhen*. Reliable and secure memories based on algebraic manipulation correction codes. / Zhen Wang, Mark G. Karpovsky // IOLTS. — IEEE Computer Society, 2012. — P. 146–149.

[98] *Ahlswede, R.* Unidirectional error control codes and related combinatorial problems / R. Ahlswede, H. Aydinian, L.H. Khachatrian // Proceedings of Eight International Workshop on Algebraic and Combinatorial Coding Theory. — 2002. — P. 6–9.

[99] Anonymous quantum communication / Gilles Brassard, Anne Broadbent, Joseph Fitzsimons et al. // ASIACRYPT / Ed. by Kaoru Kurosawa. — Vol. 4833 of *Lecture Notes in Computer Science*. — Springer, 2007. — P. 460–473.

[100] *Underwood, Craig Ian*. Single event effects in commercial memory devices in the space radiation environment: Ph.D. thesis / University of Surrey. — 1996.

[101] Investigation of increased multi-bit failure rate due to neutron induced seu in advanced embedded srams / Georg Georgakos, Peter Huber, Martin Ostermayr et al. // 2007 IEEE Symposium on VLSI Circuits. — 2007.

[102] *Жуков, А. Е.* Криптоанализ по побочным каналам (side channel attacks) / А. Е. Жуков // *Защита информации. Инсайд : информационно-методический журнал*. — 2010. — № 5. — С. 28–33.

[103] *Skorobogatov, S.* Side-channel attacks: new directions and horizons / S. Skorobogatov // ECRYPT2 School on Design and Security of Cryptographic Algorithms and Devices. — Albena near Varna, Bulgaria: 2011.

[104] *Wright, P*. Spycatcher: The Candid Autobiography of a Senior Intelligence Officer. / P Wright. — Viking, New York, 1987.

[105] *Skorobogatov, S.* Physical Attacks on Tamper Resistance: Progress and Lessons / S. Skorobogatov // 2nd ARO Special Workshop on HW Assurance. — Washington DC: 2011.

[106] *Skorobogatov, S.* Fault attacks on secure chips: from glitch to flash / S. Skorobogatov // ECRYPT2 School on Design and Security of Cryptographic Algorithms and Devices. — Albena near Varna, Bulgaria: 2011.

[107] *Kocher, Paul C*. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems / Paul C Kocher // Advances in Cryptology–CRYPTO'96 / Springer. — 1996. — P. 104–113. http://link.springer.com/chapter/10.1007/3-540-68697-5_9.

[108] *Brumley, David*. Remote timing attacks are practical / David Brumley, Dan Boneh // *Computer Networks*. — 2005. — Vol. 48, no. 5. — P. 701–716. http://www.sciencedirect.com/science/article/pii/S1389128605000125.

[109] *Song, Dawn Xiaodong*. Timing analysis of keystrokes and timing attacks on ssh / Dawn Xiaodong Song, David Wagner, Xuqing Tian // Proceedings of the 10th Conference on USENIX Security Symposium - Volume 10. — SSYM'01. — Berkeley, CA, USA: USENIX Association, 2001.

[110] *Kocher, P.* Differential Power Analysis / P. Kocher, J. Jaffe, B. Jun // Proc of Advances in Cryptology (CRYPTO'99), LNCS 1666. — 1999. — P. 388–397.

[111] *Sommer, R. M.* Smartly alayzing the simplicity and the power of simple power alaysis on smartcards / R. M. Sommer // CHES 2000, LNCS 1965. — 2000. — P. 78–92.

[112] *Messerges, T. S.* Examining smart-card security under the threat of power alalysis attacks / T. S. Messerges, E. A. Dabbish, R. H. Sloan // *IEEE Trans. Computers*. — 2002. — Vol. 51(5). — P. 541–552.

[113] *Novak, R.* SPA-based adaptive chosen-ciphertext attack on RSA implementation / R. Novak // Public Key Cryptography 2002, LNCS 2274. — 2002. — P. 252–262.

[114] *Black, J.* Side-channel attacks on symmetric encrypton schemes: the case for authenticated encryption / J. Black, H. Urtubia // Proc of 11th USINEX Security Symposium. — 2002. — P. 327–338.

[115] *Kocher, Paul C.* Differential power analysis / Paul C. Kocher, Joshua Jaffe, Benjamin Jun // Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology. — CRYPTO '99. — London, UK, UK: Springer-Verlag, 1999. — P. 388–397.

[116] *Quisquater, Jean-Jaques*. Side channel attacks: State of the art: Tech. rep. / Jean-Jaques Quisquater, Franois Koene: 2002.

[117] *Blonk, J.* Introduction to side channel attacks and non invasive attacks / J. Blonk // FIPS Physical security workshop. — Hawaii: 2005.

[118] *Messerges, T. S.* Investigations of power analysis attacks on smartcards / T. S. Messerges, E.A. Dabbish, R. H. Sloan // Proc. USENIX Workshop on Smartcard Technology. — 1999.

[119] *den Boer, Bert*. A DPA attack against the modular reduction within a CRT implementation of RSA / Bert den Boer, Kerstin Lemke, Guntram Wicke // Cryptographic Hardware and Embedded Systems-CHES 2002. — Springer, 2003. — P. 228–243.

[120] Power analysis, what is now possible... / M.-L. Akkar, R. Bevan, P. Dischamp, D. Moyart // Advances in Cryptology - ASIACRYPT' 00, LNCS / Ed. by T. Okamoto. — No. 1976. — Springer-Verlag, 2000.

[121] *Van Eck, Wim*. Electromagnetic radiation from video display units: an eavesdropping risk? / Wim Van Eck // *Computers & Security*. — 1985. — Vol. 4, no. 4. — P. 269–286.

[122] *Kuhn, M. G.* Electromagnetic Eavesdropping Risks of Flat-Panel Displays / M. G. Kuhn // Proc. 4th Workshop on Privacy Enhancing Technologies, LNCS 3424. — Toronto: 2004.

[123] The em side—channel / Dakshi Agrawal, Bruce Archambeault, Josyula R. Rao, Pankaj Rohatgi // Cryptographic Hardware and Embedded Systems-CHES 2002. — Springer, 2003. — P. 29–45.

[124] Glitch and laser fault attacks onto a secure AES implementation on a SRAM-based FPGA / Gaetan Canivet, Paolo Maistri, Régis Leveugle et al. // *Journal of cryptology*. — 2011. — Vol. 24, no. 2. — P. 247–268.

[125] *Schmidt, J.-M.* A practical fault attack on square and multiply / J.-M. Schmidt, C. Herbst // Proceedings of the 2008 5th Workshop on Fault Diagnosis and Tolerance in Cryptography / Ed. by editor; IEEE Computer Society. — 2008. — P. 53–58.

[126] *Kim, C.* Fault attacks for CRT based RSA: New attacks, new results, and new countermeasures / C. Kim, J.-J. Quisquater // Information Security Theory and Practices. Smart Cards, Mobile and Ubiquitous Computing Systems / Ed. by D. Sauveron, K. Markantonakis, A. Bilas, J.-J. Quisquater. — Springer Berlin / Heidelberg. — Vol. 4462 of *Lecture Notes in Computer Science*. — P. 215–228.

[127] *Skorobogatov, S. P.* Optical fault induction attacks / S. P. Skorobogatov, R. J. Anderson // Revised Papers from the 4th International Workshop on Cryptographic Hardware and Embedded Systems. — Springer-Verlag. — P. 2–12.

[128] Case study of a fault attack on asynchronous DES crypto-processors / Y. Monnet, M. Renaudin, R. Leveugle et al. // Fault Diagnosis and Tolerance in Cryptography / Ed. by L. Breveglieri, I. Koren, D. Naccache, J.-P. Seifert. — Springer Berlin / Heidelberg, 2006. — Vol. 4236 of *Lecture Notes in Computer Science*. — P. 88–97.

[129] *Schmidt, J. M.* Optical and EM fault-attackson CRT-based RSA: concrete results / J. M. Schmidt, M. Hutter // Austrochip 2007: 15th Austrian Workshop on Microelectronics. — 2007. — P. 61–67.

[130] On a new way to read data from memory / David Samyde, Sergei Skorobogatov, Ross Anderson, J-J Quisquater // Security in Storage Workshop, 2002. Proceedings. First International IEEE / IEEE. — 2002. — P. 65–69.

[131] Breaking public key cryptosystems on tamper resistant devices in the presence of transient faults / Feng Bao, Robert H Deng, Yongfei Han et al. // Security Protocols / Springer. — 1998. — P. 115–124.

[132] *Kuhn, M.* Optical Time-Domain Eavesdropping Risks of CRT Displays / M. Kuhn // Proc. of the 2002 Symposium on Security and Privacy. — 2002. — P. 3–18.

[133] *Loughry, J.* Information leakage from optical emanations / J. Loughry, D. Umphress // ACM Transactions on Information and System Security. — Vol. 5. — 2002. — P. 262–289.

[134] *Shamir, Adi*. Acoustic cryptanalysis – on nosy people and noisy machines / Adi Shamir, Eran Tromer // Rump Session at the 23th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT O4). — Interlaken (Switzerland): 2004.

[135] *Herath, Isuru*. Side channel attacks: Measures and countermeasures / Isuru Herath, Roshan G Ragel // 14th Annual Conference of the IET Sri Lanka Network. — Sri Lanka: 2007. — October.

[136] *Page, Dan*. Theoretical use of cache memory as a cryptanalytic side-channel. / Dan Page // *IACR Cryptology ePrint Archive*. — 2002. — Vol. 2002. — P. 169.

[137] Side channel cryptanalysis of product ciphers / J. Kelsey, B. Schneier, D. Wagner, C. Hall // Proc. of the 5th European Symposium on Research in Computer Security, LNCS 1485. — 1998. — P. 97–110.

[138] Cryptanalysis of Block Ciphers Implemented on Computers with Cache / Y. Tsuonoo, E. Tsujihara, K. Minematsu, H. Miyauchi // International Symposium on Information Theory and Its Applications. — 2002. — P. 803–806.

[139] Cryptanalysis of DES implemented on computers with cache / Yukiyasu Tsunoo, Teruo Saito, Tomoyasu Suzaki et al. // Cryptographic Hardware and Embedded Systems-CHES 2003. — Springer, 2003. — P. 62–76.

[140] *Osvik, Dag Arne*. Cache attacks and countermeasures: the case of aes / Dag Arne Osvik, Adi Shamir, Eran Tromer // Topics in Cryptology–CT-RSA 2006. — Springer, 2006. — P. 1–20. http://link.springer.com/chapter/10.1007/11605805_1.

[141] *Tiu, C. C.* A New Frequency-Based Side Channel Attack for Embedded Systems. — 2005.

[142] *Yang, Bo*. Scan based side channel attack on dedicated hardware implementations of data encryption standard / Bo Yang, Kaijie Wu, Ramesh Karri // Test Conference, 2004. Proceedings. ITC 2004. International / IEEE. — 2004. — P. 339–344.

[143] FIPS PUB 140-2 Security Requirements for Cryptographic Modules: Tech. rep.: National Institute of Standards and Technology. http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf.

[144] *Vaudenay, S.* Security Flaws Induced by CBC Padding - Applications to SSL, IPSEC, WTLC / S. Vaudenay // EUROCRYPT 2002, LNCS 2332. — 2002. — P. 543–545.

[145] *Skorobogatov, Sergei*. Low temperature data remanence in static RAM: Tech. Rep. UCAM-CL-TR-536 / Sergei Skorobogatov: University of Cambridge, Computer Laboratory, 2002. — June. http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-536.pdf.

Список использованных сокращений

AES – advanced encryption standard;
AMD – algebraic manipulation detection;
APN – almost perfect nonlinear;
CED – concurrent error detection;
DED – double error detecting;
DEMA – differential electromagnetic attack;
DES – digital encryption standard;
DFA – differential fault attack;
DPA – differential power attack;
PN – perfect nonlinear;
RSA – Rivest, Shamir, Adleman;
SEC – single error correcting;
SEMA – simple electomagnetic attack;
SPA – simple power attack;

ВАК – Высшая Аттестационная Комиссия;
БОО – блок обнаружения ошибок;
ЛПК – линейный помехоустойчивый код;
НПК – нелинейный помехоустойчивый код;
ОСН – обобщённый систематический надёжный;
ОЗУ – оперативное запоминающее устройство;
ПЗУ – постоянное запоминающее устройство;
РМ – Рид–Маллер;
СОЗУ – статическое оперативное запоминающее устройство;
УМО – уравнение маскирования ошибки.

Приложение А

Сторонняя информация и атаки на её основе

Данное приложение является результатом диссертационного исследования, появившимся в результате основательного изучения одного из практических приложений модели канала с алгебраическими манипуляциями. Относительная новизна данного направления и его интересные результаты способствовали составлению данного обзора атак по сторонним каналам. Представленные в нём материалы свидетельствуют о ярко выраженной необходимости защиты от такого рода атак, в том числе, от атаки по привнесённым помехам.

А.1 Введение

В современном мире роль вычислительных устройств трудно переоценить. Телефоны, электронные книги, смарт–карты, планшетные и персональные компьютеры — всё это становится частью жизни человека. С ростом производительности на цифровую технику возлагается всё больше функций: оплата покупок с помощью банковских смарт–карт, пересечение границ по электронным паспортам, управление финансами через Интернет–банкинг, определение местоположения с помощью систем навигации и многое другое. С увеличением функционала устройств растёт также и количество конфиденциальной информации, к которым они получают доступ. PIN– и CSC–коды смарт–карт, номера банковских счетов и пароли для управления ими, конфиденциальные данные, хранимые на компьютере или передаваемые с телефона — все эти данные должны быть тщательны защищены от несанкционированного доступа со стороны третьих лиц.

Криптография решает как раз эту задачу — она включает в себя различные меры для обеспечения сохранности конфиденциальных данных и аутентичности информации. Симметричные и открытые криптосистемы, системы электронной цифровой подписи, хэш–функции, управление ключами — все эти направления криптографии при грамотном использовании обеспечивают высокий уровень защиты информации.

Обычно, в криптографии предполагается, что реализация алгоритмов и протоколов осуществляется по принципу «чёрного ящика», когда криптографические вычисления не только скрыты от стороннего наблюдателя, но и не могут подвергаться воздействию извне. При таких допущениях уровень защищённости криптографической системы полностью определяется математическими свойствами используемых алгоритмов и размерами ключей [42]. В классическом криптоанализе считается, что злоумышленнику известен алгоритм шифрования, а также открытый и шифрованный тексты (рисунок А.1).

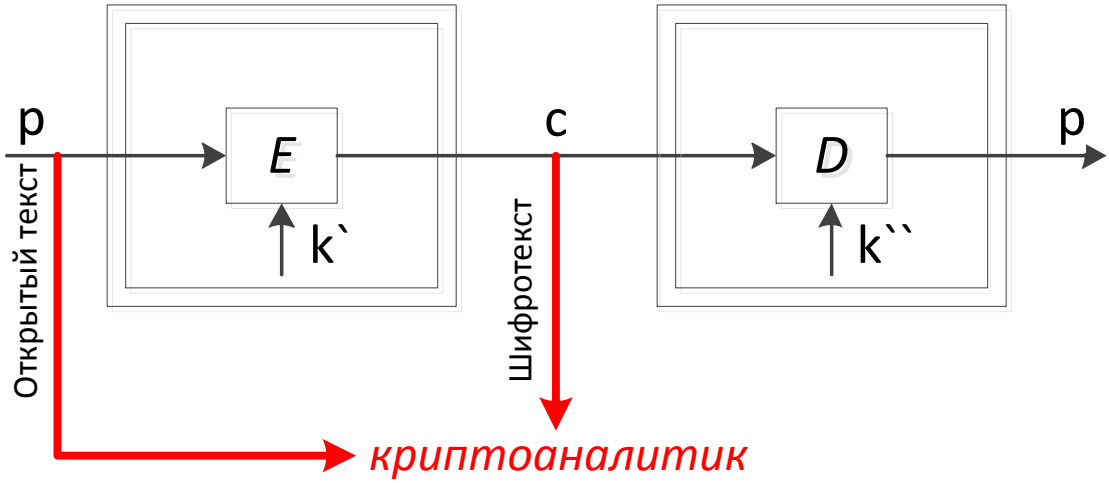


Рисунок А.1 – Классическое представление об информации, доступной криптоаналитику, где p — открытый текст, c — шифротекст, а k' и k'' — ключи шифрования и дешифрования

С другой стороны, любой криптографический алгоритм должен быть реализован либо как исполняемый процессором код, либо как соответствующая аппаратная схема. Из этого следует вывод, что процесс шифрования данных подвержен воздействию физических условий окружающей среды. Более того, самому процессу также могут быть характерны определённые физические параметры.

Атаки, использующие дополнительную информацию о реализации криптоалгоритма, позволяют значительно ослабить стойкость алгоритма, а в некоторых случаях — полностью устранить её. Эффективность этих атак гарантируется существованием корреляции между измерениями физических параметров устройства в определённые моменты и внутренним состоянием шифратора, которое, в свою очередь, связано с секретным ключом. В качестве примеров физических параметров устройства можно привести его энергопотребление, электромагнитное излучение (ЭМИ), производимые звуки и так далее (рисунок А.2). Такие атаки получили название атак по сторонним (побочным) каналам (Side–Channel Attacks, SCA).

Атаки по сторонним каналам — вид криптографических атак, использующих информацию, полученную по сторонним каналам [102]. Сторонней информацией называется любая информация, относящаяся к процессу шифрования (дешифрования) и не являющаяся открытым текстом или шифротекстом [102]. Как уже отмечалось, классический криптоанализ рассматривает криптоалгоритмы как чисто математические объекты, в то время как криптоанализ по сторонним каналам также принимает во внимание особенности их реализации. Поэтому SCA также называют атаками по реализации.

Очень удачным кажется следующее сравнение. Классический криптоанализ похож на попытку грабителей ворваться в запертый дом через надёжную входную дверь. Это потребует больших временных затрат. Сторонний криптоанализ можно сравнить с более разумными и эффективными действиями грабителей: проникнуть в дом можно через разбитое окно или же украсть ключ от двери у владельцев.

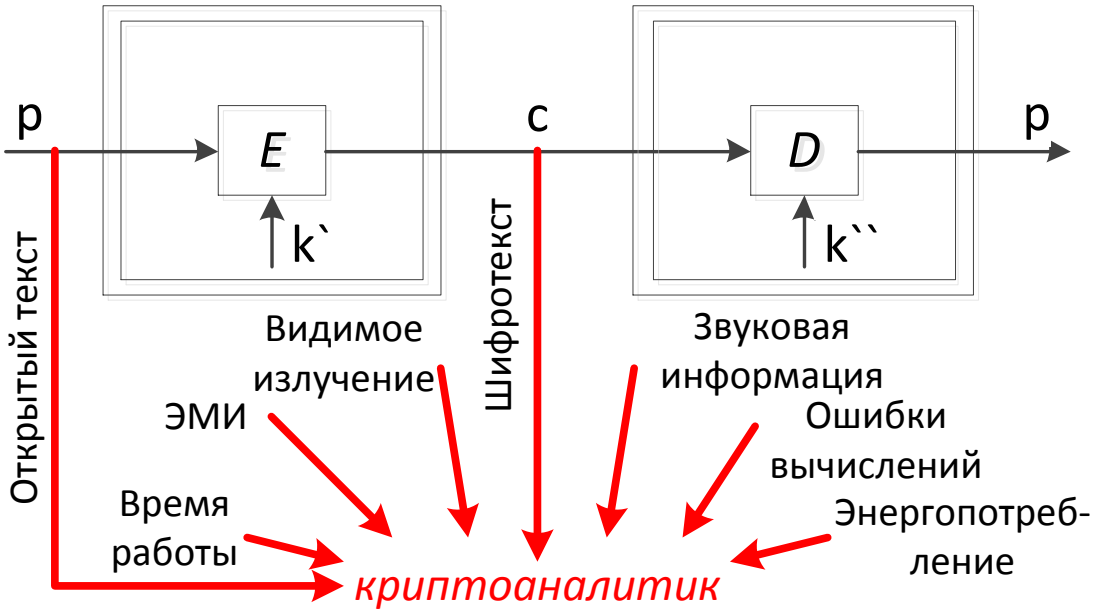


Рисунок А.2 – Примеры дополнительной информации, которая может быть доступна криптоаналитику

На практике SCA значительно эффективнее традиционных атак, основанных на математическом анализе шифра [102]. При этом атаки по сторонним каналам направлены на конкретные реализации криптографических алгоритмов, что значительно сужает область их применимости в обмен на эффективность взлома.

Интенсивное развитие атак по сторонним каналам спровоцировало значительное увеличение количества способов сбора и обработки сторонней информации. Например, противник может отслеживать энергопотребление атакуемого устройства во время осуществления операций с секретным ключом. Противник может также замерять время, затрачиваемое на выполнение криптографической операции, или анализировать поведение криптографического устройства при возникновении определённых ошибок. Часто на практике сбор сторонней информации осуществляется довольно легко, поэтому при проектировании и оценке защищённости систем необходимо учитывать угрозу от утечки сторонней информации.

Исходя из серьёзности атак по сторонним каналам остро встаёт вопрос о требованиях к реализации криптографических алгоритмов. Написание исполняемой программы или проектирование архитектуры криптомодулей должно быть продуманным и учитывать все существующие каналы утечки сторонней информации. Защищённые криптографические чипы находят применение в следующих сферах [103]:

– машиносчитываемые проездные документы (электронные паспорта, проездные документы);

– безопасность транспорта (антиугонная защита);

– поставка различных услуг (карты доступа, платёжные жетоны, RFID–метки, электронные ключи);

– сотовые телефоны (контроль батарей и аксессуаров, защищённые телефоны);

– изготовление различного оборудования (защита против клонирования, защита объектов интеллектуальной собственности);

– банковская индустрия (банковские карты, защищённые банковские операции);

– военные применения (защита данных, шифрование связи).

Первой официальной информацией, относящейся к атакам по сторонним каналам, является рассказ Питер Райт (Peter Wright) о своей работе в качестве учёного в Центре правительственной связи Великобритании в 1965 году [104]. В то время британская контрразведка MI5 безуспешно пыталась взломать шифр, которым пользовалось египетское посольство в Лондоне. Райт предложил установить микрофон около роторной шифровальной машины, используемой в посольстве, и подслушивать звуки, производимые машиной. Слушая щелчки роторов во время утренней настройки шифратора, MI5 удалось успешно определить позиции сердечников 2 или 3 роторов машины. Дополнительная информация снизила сложность взлома шифра, позволив британской контрразведке шпионить за коммуникациями египетского посольства годами.

В распоряжении современного криптоаналитика находится мощное оборудование: высокоточные цифровые мультиметры, источники лазерного и электромагнитного излучения, различные химические вещества, а также мощные вычислительные средства. Всё это может быть успешно применено для взлома криптографических алгоритмов, но всегда надо взвешивать требуемые затраты и результат взлома. Для того, чтобы взломать банковскую карту с $100 на счету едва ли стоит тратить $10000 на аппаратуру.

A.2 Описание устройства смарт-карт как жертв атак по сторонним каналам

Широкое распространение и простота внутренней реализации смарт–карт привело к тому, что они стали наиболее частыми жертвами злоумышленников. В данном разделе даётся краткое описание устройства смарт–карт.

По сути, смарт–карта это компьютер, встроенный в сейф. Он состоит из процессора (обычно, 8– или 32–битный), постоянного запоминающего устройства (ПЗУ, ROM), электрически-стираемого программируемого ПЗУ (ЭСППЗУ, EEPROM) и небольшого оперативного запоминающего устройства (ОЗУ, RAM), что позволяет смарт–карте выполнять вычисления. Основная задача смарт–карты заключается в выполнении криптографических операций, использующих секретный параметр (ключ), не раскрывая значение ключа окружающему миру. Задача атакующего противоположна — вычислить ключ.

Процессор встроен в чип и подключён к окружающему миру через 8 контактов, назначение, позиция, использование и другие параметры которых стандартизированы. В дополнение ко входным/выходным проводам нас будут интересовать следующие особенности чипа:

– Энергопотребление (power consumption): смарт–карты не имеют внутренней батареи. Ток, необходимый для её работы, предоставляется оборудованием, осуществляющим операции с картой. Это делает энергопотребление смарт–карты легко измеряемым со стороны.

– Тактирование (clocking): аналогично, смарт–карты не имеют внутреннего генератора тактовой частоты, тактовый сигнал должен поступать из читающего оборудования. Эта особенность реализации карт делает их уязвимыми к атакам по времени выполнения операций.

Зачастую смарт–карты снабжены защитным экраном (shield), чья задача состоит в сокрытии внутреннего поведения чипа, и сенсоров, которые должны уничтожить всю секретную информацию при вскрытии экрана и предотвратить работу карты.

A.3 Классификация сторонних каналов

Атаки по сторонним каналам обычно классифицируются по 3 характеристикам [42]:

– контроль над вычислительным процессом;

– метод доступа к модулю;

– метод, применяемый в процессе анализа.

A.3.1 Контроль над вычислительным процессом

Относительно контроля над вычислительным процессом со стороны злоумышленника, SCA могут быть разделены на 2 категории: пассивные (passive) атаки и активные (active) атаки. К пассивным атакам относятся те виды атак, которые не влияют на протекание вычислительного процесса в атакуемом модуле; атакующий получает стороннюю информацию, но модуль при этом работает в стандартном режиме.

Активные атаки, в противоположность пассивным, подразумевают некоторое воздействие на вычислительный процесс. Система, подверженная активной атаке, может быть способна или неспособна обнаруживать такое воздействие, однако сторонний наблюдатель заметит различие в работе системы.

A.3.2 Метод доступа к модулю

При анализе защищённости криптографического модуля важно иметь представление об атакуемой поверхности — наборе физических, электрических и логических интерфейсов, которые могут быть использованы злоумышленником. На основе этого признака SCA классифицируются следующим образом:

– агрессивные (invasive) атаки;

– неагрессивные (non-invasive) атаки;

– полуагрессивные (semi-invasive) атаки.

Агрессивные атаки

Агрессивные атаки подразумевают распаковку чипа для получения прямого доступа к внутренним компонентам криптографических модулей или устройств. Типичный пример такой атаки заключается в создании дыры в пассивирующем слое и корпусе криптографического модуля и размещении щупа на шине данных для обзора передаваемой информации.

Механизмы, защищающие блоки от несанкционированного доступа, зачастую встроены в аппаратные схемы для эффективной защиты против агрессивных атак. Для примера, некоторые криптографические модули с высоким уровнем защиты обнуляют значения всей их внутренней памяти при обнаружении нарушения целостности модуля.

Этот тип атак характеризуется индивидуальным подходом к каждому атакуемому модулю (например, необходимо уничтожить часть защитного слоя с каждого чипа) и высокой стоимостью используемого оборудования. С другой стороны, эти атаки являются наиболее эффективными за счёт прямого доступа к внутренним элементам криптографического блока.

Неагрессивные атаки

Неагрессивные атаки заключаются в наблюдении за работой криптосхемы или манипулировании её работы. Эти атаки используют только внешнюю информацию, которая зачастую доступна в небольшом количестве. Стандартным примером такого типа атак являются атаки по времени выполнения вычислений: измерение времени, требуемого устройству для работы, и корреляция этого параметра с типом выполняемой операции для определения значения секретного ключа.

Основная характеристика неагрессивных атак это то, что они полностью необнаруживаемы. Для примера, смарт-карта никаким образом не может определить, что время её работы измеряется в данный момент.

Неагрессивные атаки относительно легко осуществимы на практике, не требуют особой обработки каждого чипа и выгоднее с экономической точки зрения. Таким образом, неагрессивные атаки представляют большую угрозу для некоторых типов устройств, например, смарт-карт.

Полуагрессивные атаки

Этот тип атак использует доступ к устройству, но не подразумевает повреждения чипа или создания электрических контактов кроме оригинальных. Примером такой атаки является атака с привнесением помех, когда атакующий использует лазерный луч для ионизации устройства с целью изменения данных в его памяти.

Полуагрессивные атаки являются связующим звеном между агрессивными и неагрессивными, объединяя в себе легкость повторения атаки с относительной дешевизной.

A.3.3 Метод анализа

В зависимости от метода, применяемого для анализа полученных данных, атаки по сторонним каналам могут быть разделены на простые (недифференциальные, simple, non-differential) атаки по сторонним каналам (SSCA) и дифференциальные (разностные, differential) атаки по сторонним каналам (DSCA).

В простых атаках используется сторонняя информация, зависящая в основном от выполняемых операций. Обычно, в SSCA–анализе используется одиночный след[1)] (trace), из которого может быть вычислен секретный ключ. Очевидно, что сторонняя информация, относящаяся к атакуемой команде (сигнал) должна значительно превышать информацию, относящуюся к побочным операциям (шум). Простые атаки по сторонним каналам используют зависимость между исполняемыми командами и сторонней информацией.

В случае, когда использование простых атак по сторонним каналам невозможно из–за высокого уровня шума измеряемого следа, применяют более сложный и требующий большего количества следов тип атак — дифференциальные атаки. Дифференциальные атаки по сторонним каналам используют корреляцию между обрабатываемыми данными и соответствующую по времени утечку информации из устройства. В связи с невысоким значением корреляции для более эффективного её использования необходимо применять статистические методы. При DSCA атакующий использует гипотетическую модель устройства, подвергающего атаке. Эта модель служит для предсказания значения утечки сторонней информации, и её качество зависит от возможностей атакующего.

A.4 Основные типы атак по сторонним каналам

В данном разделе будут приведены основные типы атак по сторонним каналам, некоторые простые примеры, а также текущее положение дел для наиболее исследованных атак.

A.4.1 Атака зондированием

Атака зондированием (probing attack) — агрессивная пассивная простая атака, заключающаяся в обеспечении прямого доступа к данным. Атака зондированием состоит из нескольких этапов. Для получения прямого доступа к криптографическому блоку устройство вскрывается. Для этого зачастую используют кислоты: HNO_3 и ацетон при температуре 60℃ (ручная обработка) или же соединение HNO_3 и H_2SO_4 (автоматическая) [105]. При этом защитный пассивирующий слой может быть удалён как с передней, так и с задней части чипа, полностью или частично. Примеры декапсулированных чипов представлены на рисунке А.3.

На сегодняшний день осуществление атаки становится всё более сложной задачей из–за уменьшающихся размеров чипов и применения детекторов несанкционированного доступа. Они

[1)] След — запись данных с историей событий, происходивших в системе (комп.)

могут представлять собой дополнительные металлические слои, которые формируют сенсорную сеть над непосредственной схемой. Все участки этой сети постоянно проверяются на прерывания и короткие замыкания, в случае чего устройство прекращает работу и уничтожает секретную информацию. Более того, часто также применяется мониторинг тактовой частоты (для устройств с внешним тактовым генератором). Это объясняется тем, что для упрощения наблюдения криптоаналитик обычно замедляет тактовую частоту работы устройства. Дополнительные сенсоры (ультрафиолет, свет, температура) также могут быть использованы.

Однако и для таких мер защиты существуют способы противодействия со стороны атакующего. Например, существует способ нейтрализации сенсорной сети с помощью установки, генерирующей фокусированный ионный пучок (focused ion beam workstation) [106].

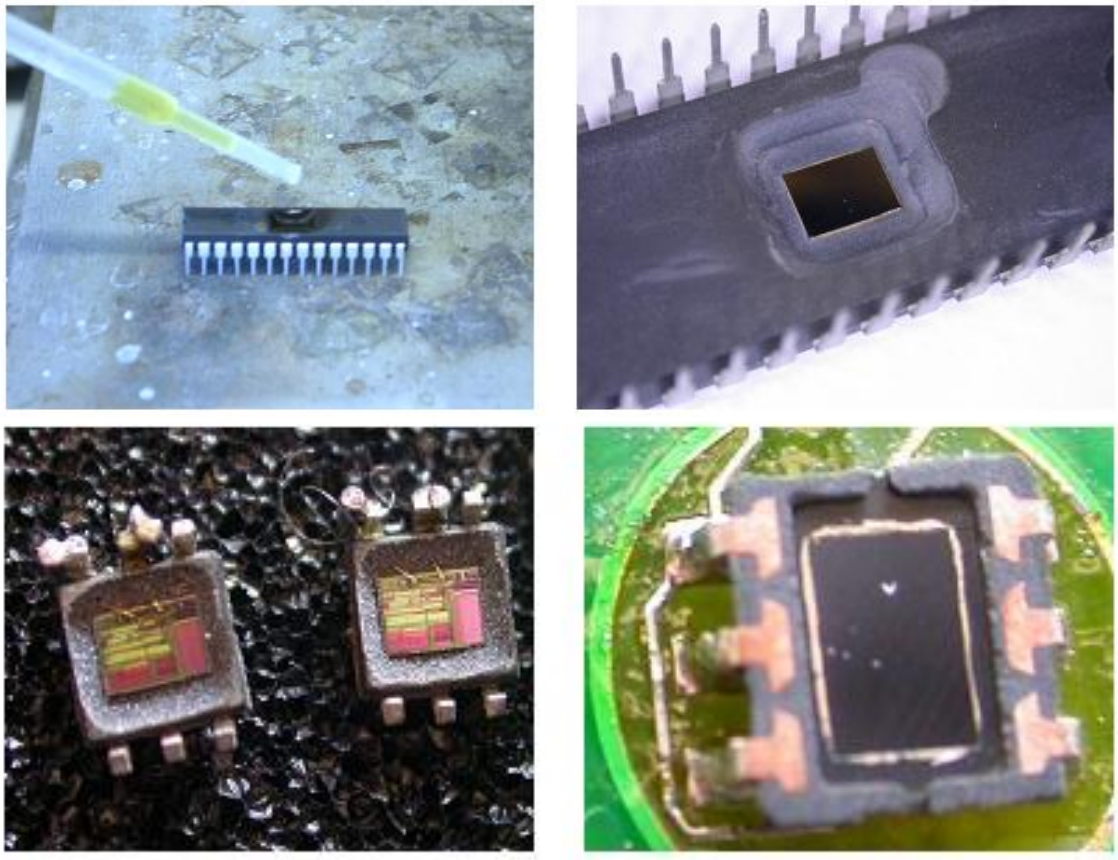


Рисунок А.3 – Примеры декапсуляции чипов из [106]

Следующим этапом атаки является исследование печатной платы с помощью микроскопа и обнаружение шин данных, вычислительных блоков и модулей памяти. Для этого используются микроскопы. Изображения печатных плат, полученные с помощью микроскопов, представлены на рисунке А.4.

При текущем состоянии технологий оптические микроскопы, в силу ограничений, накладываемых оптикой и длиной волны, заменяются на электронные микроскопы.

Заключительным этапом атаки является размещение щупов (зондов) на проводниках, по которым идут сигналы данных, и/или исследование ячеек памяти с помощью микроскопа. Также,

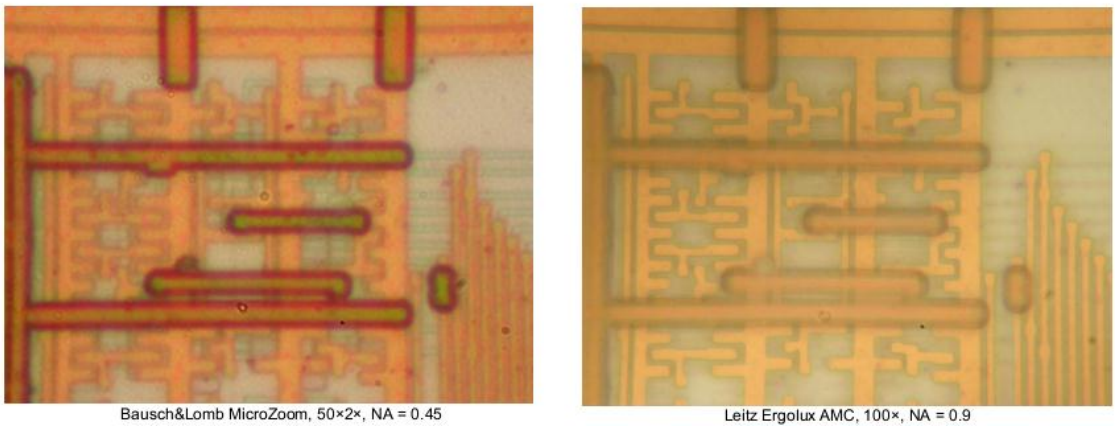


Рисунок А.4 – Изображения чипа, получаемые с помощью оптического (слева) и электронного микроскопов (справа) из [106]

через подключенные щупы в устройство могут передаваться тестовые сигналы с последующим анализом выхода. Примеры щупов представлены на рисунке А.5.

На сегодняшний день процесс проведения атаки упрощается с помощью использования зондирующей установки, включающей в себя микроскопы и микроманипуляторы для установки щупов на поверхности чипа. Такие установки используются в полупроводниковой промышленности для тестирования образцов продукции; цена на вторичном рынке составляет порядка $10000 (рисунок А.6).

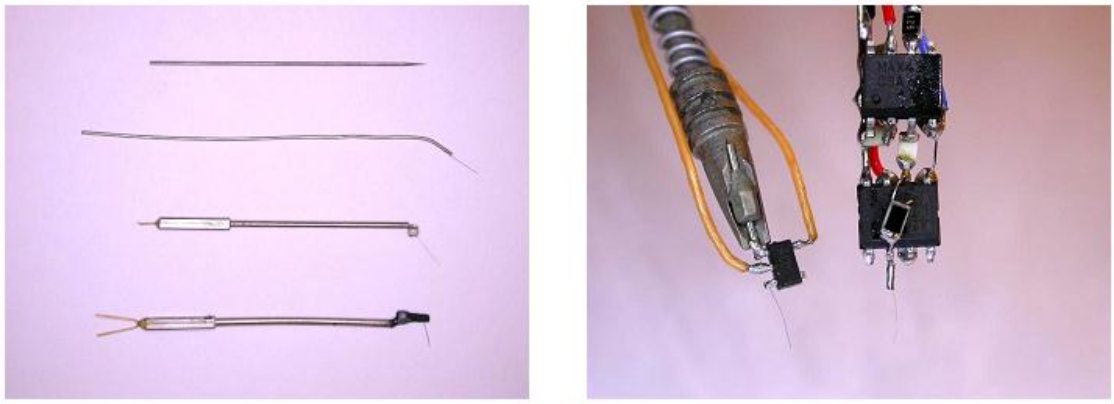


Рисунок А.5 – Щупы для подключения к чипу [106]

Методы защиты: как уже было написано выше, основными методами защиты криптографических устройств от атак зондированием являются экранирование и использование сенсорных сетей и датчиков несанкционированного доступа. Недостатком этих мер является существенное увеличение стоимости и размеров устройства.

А.4.2 Атака по времени

Атака по времени (timing attack) — пассивная неагрессивная атака, основанная на анализе времени выполнения операций шифрования. Атака по времени является самой первой из атак

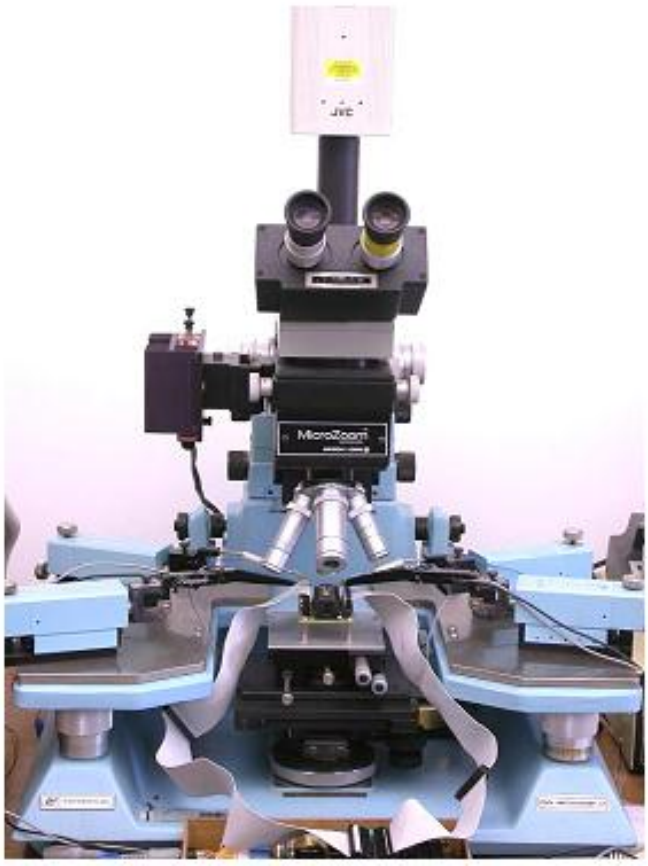


Рисунок А.6 – Пример зондирующей установки из [106]

по сторонним каналам, появившейся в гражданской криптографии. Была предложена Полом Кохером (Paul Kocher) в 1996 году [107]. Атака по времени основана на измерении времени, необходимого криптографическому модулю для выполнения операции шифрования (дешифрования). Это информация может вести к раскрытию информации о секретном ключе.

Вычисления в реализациях криптографических алгоритмов часто выполняются за различные интервалы времени. Причинами тому могут быть оптимизация производительности, пропускающая лишние операции, ветвление и условные выражения, чтение данных из кэша в оперативной памяти, операции, выполняемые за недетерминированное время (например, умножение или деление) и так далее. Зачастую, время выполнения операций зависит как от входных параметров, так и от секретного ключа. Если при этом из этого разброса времени можно получить некоторую информацию о скрытых параметрах и если иметь достаточно знаний о конкретной реализации, то применение статистических методов анализа способно полностью восстановить эти скрытые параметры.

В литературе приняты следующие основные предположения относительно атак по времени [42]:

– время выполнения криптографических операций некоторым образом зависит от ключа;

– достаточно большое количество операций шифрования может быть выполнено на одном ключе;

– время может быть замерено с определённым уровнем ошибки. Чем меньше ошибка, тем меньше опытов требуется.

Необходимо отметить, что атаки по времени были крайне успешно применены для взлома сетевых криптографических протоколов, например, OpenSSL и SSH [108, 109].

Ниже приведён пример алгоритма, который может быть успешно взломан с помощью атаки по времени. Быстрое возведение в степень используется в алгоритмах Диффи–Хеллмана и RSA. Тщательно замеряя время выполнения операции экспоненцирования $y = a^x$, атакующий может найти вес секретного ключа $x = (x_{n-1}, x_{n-2}, ..., x_0)$ (если бит x_i равен 1, то выполняется дополнительная инструкция) и даже точное его значение.

Вход: a, x

1: $R \leftarrow 1$
2: **для** $i \leftarrow n-1$ **до** 0 **выполнить**
3: $\quad R \leftarrow R^2$
4: $\quad$ **если** $x_i = 1$ **тогда**
5: $\quad\quad R \leftarrow R \cdot a$
6: $\quad$ **конец если**
7: **конец для**
8: $y \leftarrow R$
9: **вернуть** y

Выход: Вычисленное значение $y = a^x$

Алгоритм А.1 – Вычисление $y = a^x$ по схеме Горнера

Методы защиты: основным методом защиты от утечки информации по сторонним каналам является выполнения условия независимости вычислений от данных. Если вычисления явно не зависят от входных данных или секретного ключа, то криптоаналитик также не сможет их получить из информации по побочному каналу. Добиться этого можно несколькими способами:

– маскирование (masking) — способ при котором к входным данным применяется некоторая маска, далее производятся вычисления, после чего осуществляется обратная коррекция маски. Таким образом при атаке по сторонним каналам криптоаналитик получит некоторое промежуточное значение, не раскрывающее входных данных;

– произведение вычислений вслепую (blinding) — подход в криптографии, при котором устройство предоставляет функцию шифрования, не зная при этом реальных входных и выходных данных. Впервые подобный подход был применён к алгоритму RSA и основан на свойстве гомоморфности функции шифрования.

Выполнение условия независимости вычислений от данных является универсальным методом защиты от утечки информации по большинству сторонних каналов. Однако приведённые выше

методы не всегда могут быть применены к конкретным алгоритмам и устройствам. В этом случае применяют методы, предназначенные для защиты от конкретных типов атак.

Основным методом защиты аппаратных схем от атак по времени является введение шума (noise injection). Шум может быть добавлен как программно (введение случайных вычислений и фиктивных задержек), так и аппаратно (установка различных генераторов шума).

Другая идея для защиты от атак по времени заключается в том, что вычисления, соответствующие всем веткам условного оператора, должны занимать одинаковое количество времени (branch equalization). Чтобы криптоаналитик не смог провести атаку по времени исполнения, все этапы шифрования в устройстве должны выполняться за одинаковое время. Добиться этого можно следующими способами:

– добавление фиксированной задержки. Если известна конечная аппаратная платформа, то можно рассчитать время выполнения каждой операции и уравнять их, добавив фиксированные задержки;

– выполнение одновременно нескольких возможных операций. Если в какой-то момент исполнения алгоритма должно выполниться либо умножение, либо возведение в квадрат, то должны выполниться обе операции, а ненужный результат отброшен.

Очевидным недостатком таких решений является замедление работы устройства. Также такие меры не помогают от динамических задержек, таких как промах в кэш (подробнее в разделе А.4.8).

Другим способом защиты программных реализаций криптографических алгоритмов является устранение условных переходов в алгоритме, однако это представляется достаточно нетривиальной задачей. Эффективность предложенных средств такова: накладывание шума ослабляет силу атаки по времени, добавляя около 2-10% вычислительных расходов, но не устраняет ее, тогда как удаление ветвлений часто устраняет атаку, но за весьма значительную цену [42].

На данный момент большинство ошибок, провоцировавших утечку сторонней информации, были устранены производителями [42, 102]. Использование внутренних источников тактового сигнала и фазовой автоподстройки частоты (phase-locked loop, PLL), а также случайных задержек и фиктивных операций делает современные криптомодули мало уязвимыми к атакам по времени.

А.4.3 Атака по энергопотреблению

Атака по энергопотреблению (power analysis attack) — пассивная неагрессивная атака, основанная на анализе мощности, потребляемой устройством в процессе шифрования. Была предложена Полом Кохером в 1999 году [110]. С помощью простого и/или дифференциального анализа энергопотребления злоумышленник способен получить представление о процессах, протекающих в устройстве. Объединение полученной сторонней информации с другими методами криптоанализа позволяет значительно снизить сложность вычисления секретного ключа.

Как известно, интегральные схемы состоят из транзисторов, работающих как управляемые напряжением переключатели. В зависимости от напряжения, поданного на затвор транзистора, через него может течь или не течь ток. Этот ток переносит заряд дальше — к затворам других транзисторов, к проводникам и другим видам нагрузок цепи. Движение электрических зарядов поглощает энергию и выделяет электромагнитное излучение; оба эти параметра легко измеримы извне.

Чтобы измерить потребляемую схемой мощность необходимо последовательно с цепью питания или заземления подключить резистор малого сопротивления (например, 50 Ом). Падение напряжения, деленное на сопротивление, даст силу тока. Современные лаборатории располагают оборудованием, способным производить цифровые измерения напряжения на исключительно высоких частотах (более 1 ГГц) и с превосходной точностью (ошибка менее 1%). Устройства, способные вести измерения с частотой от 200 МГц и передавать данные компьютеру, стоят менее $400 [110].

Атаки по мощности являются наиболее исследуемыми из всех атак по сторонним каналам. На данный момент опубликовано более 200 работ на эту тему. Результаты исследований показывают, что атаки по энергопотреблению являются очень мощным инструментом для взлома криптографических схем. Варианты проведения атак по энергопотреблению были продемонстрированы для большинства шифров, как симметричных, так и с открытым ключом, например [111–113]. Согласно экспериментальным данным из [114], простая атака по энергопотреблению для смарт-карт обычно занимает несколько секунд, в то время как проведение разностной атаки может занимать несколько часов.

Простая атака по энергопотреблению

Простая атака по энергопотреблению (simple power analysis attack, SPA attack) основана на анализе следов потребляемой устройством мощности при выполнении процесса дешифрования. Для успешного проведения атаки злоумышленник, обладая точными знаниями о внутренней реализации криптоалгоритма, использует непосредственные результаты измерений энергопотребления для получения сведений как о работе самого устройства, так и о ключе.

Пример следа энергопотребления устройства, выполняющего шифрование с помощью алгоритма DES, представлен на рисунке А.7. На графике можно различить этап начальной перестановки, 16 раундов и заключительную перестановку.

Так как SPA позволяет восстановить последовательность исполненных инструкций, она может быть использована для взлома криптографических реализаций, в которых эта последовательность зависит от обрабатываемых данных.

Примеры операций, подвергающихся атаке [116]:

– Расписание ключей (key schedule) DES: вычисление подключа в каждом раунде DES использует циклический сдвиг 28-битного регистра ключа. Условный переход зачастую используется для переноса сдвинутого бита в другой конец регистра (если этот бит равен 0,

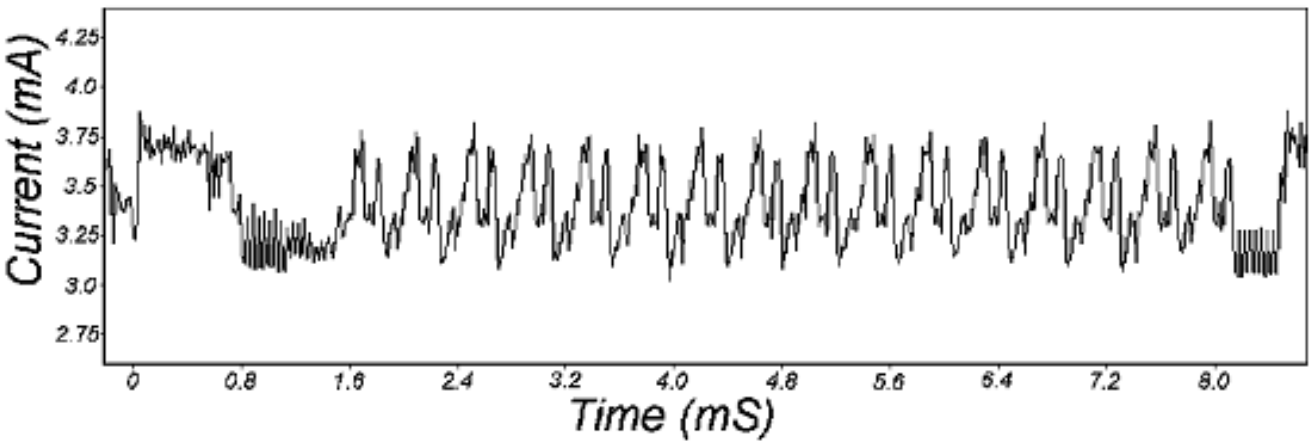


Рисунок А.7 – График энергопотребления смарт-карты, реализующей DES [115]

то его не переносят). Результирующий след потребляемой энергии для битов равных 0 и 1 будет принимать различный вид, если это значение бита влияет на выполняемые операции. Пример различий в энергопотреблении устройства при вычислении ключа раунда представлен на рисунке А.8.

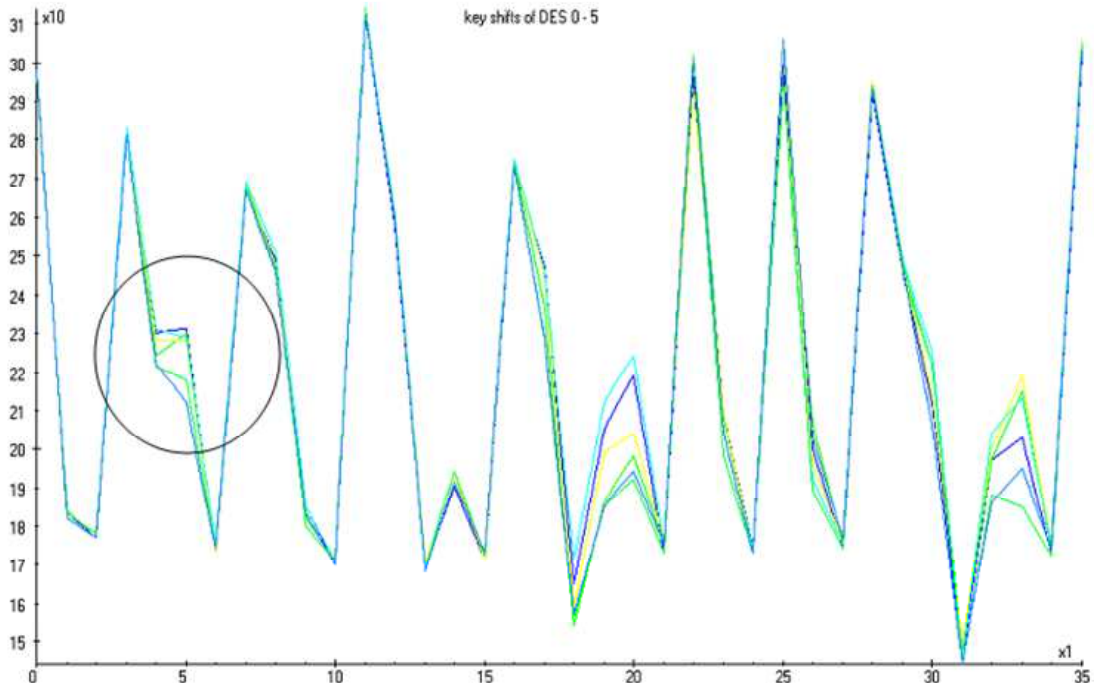


Рисунок А.8 – Различие в энергопотреблении устройства при вычислении подключа раунда алгоритма DES в зависимости от значения старшего бита ключевого регистра [117]

– Перестановки DES: реализации алгоритма DES используют большое количество перестановок. Условные переходы в программном коде могут приводить к значительным различиям потребления энергии для 0 и 1 битов.

– Сравнения: операции сравнения строк или значений памяти обычно используют условные переходы в случае, когда обнаружено несовпадение. Эти переходы служат причиной значительных различий во времени и энергопотреблении устройства.

– Умножители: блоки умножения по модулю провоцируют огромные утечки сторонней информации. Величина утечки зависит от устройства умножителя, но зачастую имеет очень сильную корреляцию со значениями операндов и их весами Хэмминга.

– Экспоненциаторы: простая реализация операции экспоненцирования побитово сканирует значение экспоненты, выполняя операцию возведения в квадрат на каждой итерации с дополнительным умножением для каждого единичного бита экспоненты. Экспонента может быть скомпрометирована в случае, если операции возведения в квадрат и умножения потребляют различное количество энергии, занимают различное время выполнения или разделены различными инструкциями.

Дифференциальная атака по энергопотреблению

Дифференциальная атака по энергопотреблению (differential power analysis attack, DPA attack) — одно из наиболее эффективных средств для осуществления атак по сторонним каналам. В отличие от простых атак по энергопотреблению, которые требуют детального знания реализации алгоритма, дифференциальные атаки используют статистические методы в процессе анализа, не требуя при этом подробных данных о конкретной реализации. Это делает DPA атаку крайне эффективной, тем более что для ее проведения требуются крайне незначительные ресурсы [42].

Помимо достаточно значимых изменений энергопотребления, связанных с последовательностью выполнения инструкций, существуют так же эффекты, вызванные обрабатываемыми данными. Эти изменения достаточно малы и иногда «затемняются» ошибками измерения и другим шумом. В таких случаях часто все же возможно взломать систему, используя статистические функции, привязанные к алгоритму шифрования.

Стандартная DPA–атака заключается в накоплении злоумышленником множества пар $\{C_i, T_i\}$, где T_i — след, соответствующий шифротексту C_i. Также, злоумышленник определяет функцию выбора D, которая в зависимости от шифротекста и предположения о значении части ключа выдаёт бинарный результат. Идея заключается в том, что если предположение о ключе верно, то результат функции D отображает особенности, проявляемые в процессе вычислений. В противном случае, D будет случайной функцией на всём множестве шифротекстов.

Злоумышленник делает предположение K_g о ключе и использует это предположение и функцию выбора для разделения множества следов на 2 подмножества: для одного $D(C_i, K_g) = 0$, а для другого $D(C_i, K_g) = 1$. Далее он исследует различие усредненных по подмножествам следов. Если K_g был ошибочным, то эти два подмножества некоррелированы, и разность усреднённых следов становится плоской с ростом размера выборки (рисунок А.9). В случае, если K_g был верным, то разность стремится к корреляции D и энергопотребления, которая будет пред-

ставлена пиком. Пример графика разности при верном предположении о значении части ключа приведён на рисунке А.10.

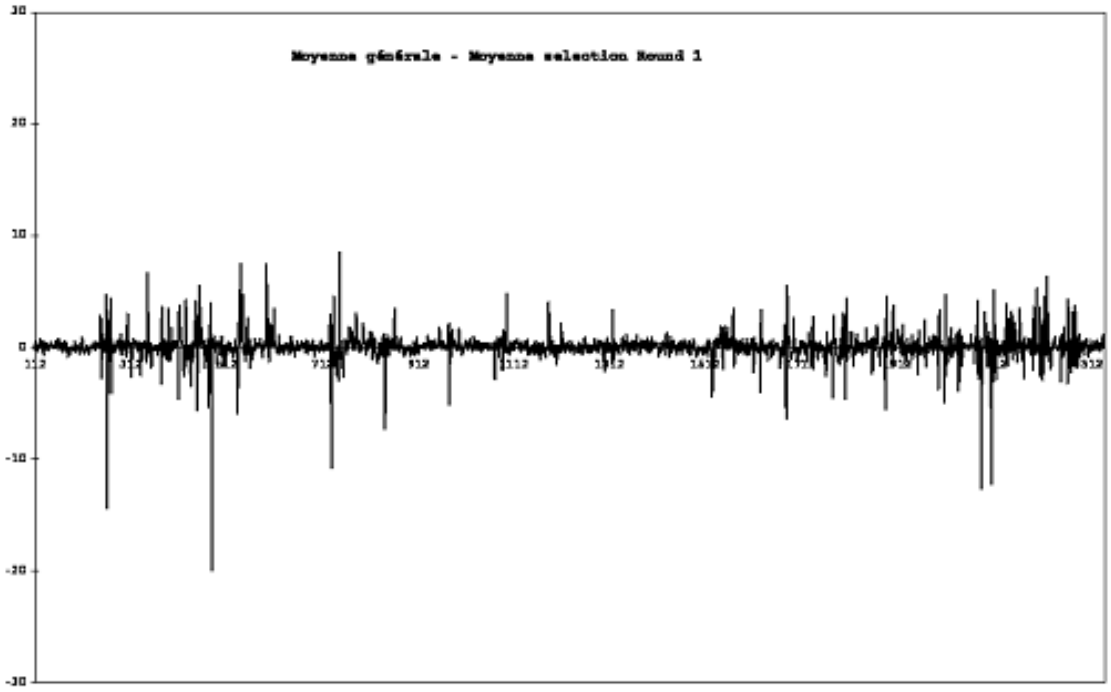


Рисунок А.9 – Разность между усреднёнными следами энергопотребления при ошибочном предположении о значении битов ключа [102]

Атаки по энергопотреблению были легко применены против RSA. В качестве простого примера уязвимости можно привести тот факт, что если потребление энергии блоками умножения и возведения в квадрат различается настолько, что их можно отличать друг от друга, то значение секретной экспоненты может быть легко получено из следов энергопотребления этих блоков (возможно, усреднённых по некоторому количеству операций для уменьшения шума). Дальнейшие результаты по этой теме могут быть найдены в [118] и [119].

Аккар (Akkar) и другие вернулись к проблеме анализа мощности в [120]. Они попытались определить относительную значимость утечки информации об энергопотреблении в зависимости от выполняемых инструкций, обрабатываемых данных и т.д. и в результате предложили довольно простую модель. Основываясь на некоторых практических результатах, они показали, что критерий разделения Кохера (разделение следов в зависимости от значения конкретного бита) не является оптимальным, и предложили другой критерий, который максимизирует разность энергопотребления. Это позволяет в 5 раз сократить размеры необходимых статистических данных, но требует при этом более детального знания о реализации алгоритма.

Методы защиты

Основным подходом для защиты от атак по энергопотреблению является балансировка энергопотребления. По возможности, при проведении операций должны быть задействованы все ап-

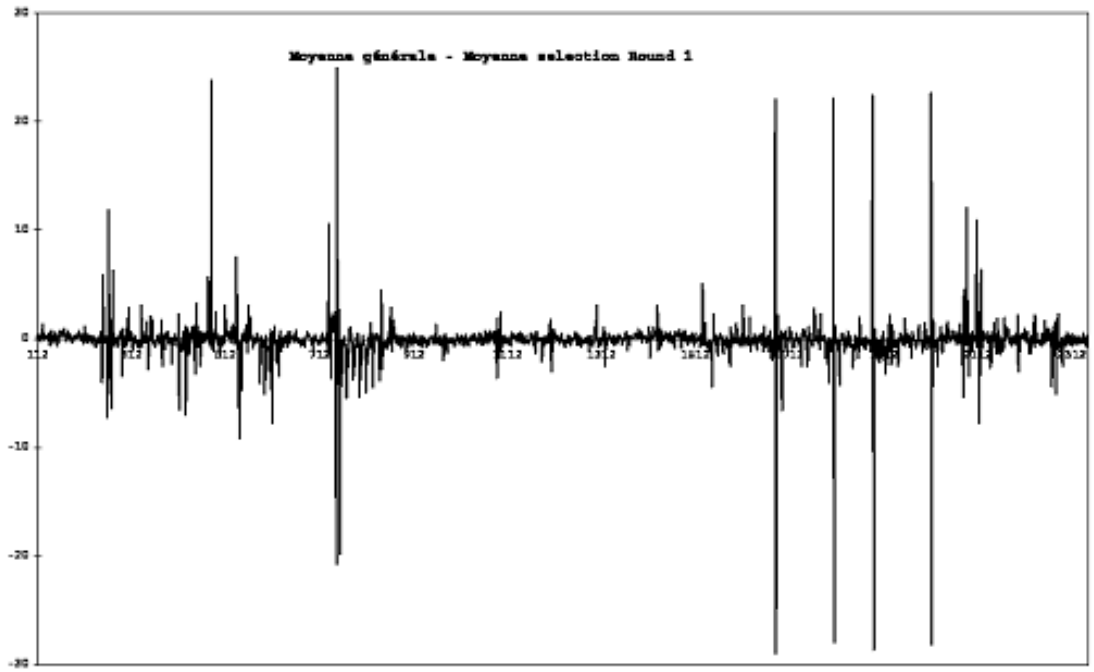


Рисунок А.10 – Разность между усреднёнными следами энергопотребления при правильном предположении о значении 6 битов ключа [102]

паратные части устройства (например регистры или вентили), на неиспользуемых частях следует проводить ложные вычисления. Таким образом можно добиться постоянства энергопотребления устройством.

Методы защиты аппаратных схем от атак по энергопотреблению также включают использование внутренних источников энергии, исполнение инструкций в случайном порядке, случайное переименование регистров, использование 2 конденсаторов, один из которых заряжается от внешнего источника, пока второй питает устройство, и многое другое.

Интересным методом защиты криптографических устройств от атак по энергопотреблению является разделение обрабатываемых данных на 2 части таким образом, чтобы можно было легко осуществить обратную процедуру объединения (например, ключ K можно разделить на части K_1 и K_2 с помощью операции XOR: $K = K_1$ XOR K_2). Обе части ключа обрабатываются отдельно, из-за чего пропадает корреляция между битами исходного ключа K (они больше не используются в чистом виде) и энергопотреблением. Следовательно, метод DPA не может быть применён. Однако для борьбы с таким методом защиты был разработан DPA второго порядка. Далее он был обобщён на случай любого порядка.

На данный момент атаки по энергопотреблению находятся в фокусе сегодняшних исследований сторонних каналов, однако проведение таких атак становится всё более сложной задачей. С увеличением частоты работы устройств увеличивается и уровень шума измерений, эти факторы требуют использования более быстрого и точного оборудования. Кроме того, уменьшению отношения сигнал/шум способствует снижение энергоснабжения с 5 В до 1 В. Выделение изменения 1 бита усложняется за счёт перехода к данным большей разрядности (переход от 8–битных дан-

ных к 32– и 64–битным данным). Более сложные электрические цепи приводят к увеличению уровня помех от других элементов, возникновению перекрёстных помех (crosstalks). Помимо технологических сложностей, возникающих у криптоаналитиков, усилиями учёных и инженеров на данный момент разработаны эффективные меры противодействия атакам по энергопотреблению для большинства алгоритмов.

А.4.4 Атака по электромагнитному излучению

Атака по электромагнитному излучению (electromagnetic analysis, EM attack) — тип пассивной неагрессивной атаки по сторонним каналам, заключающийся в анализе электромагнитного излучения криптографического модуля.

Простым примером атаки по электромагнитному излучению является так называемый перехват ван Эйка. В 1985 году вышла статья Вима ван Эйка (Wim van Eyck) «Электромагнитное излучение от мониторов: опасность подслушивания» [121]. В этой статье подробно описывалось, как с помощью оборудования суммарной стоимостью не больше $100 можно осуществить перехват информации, представленной на удаленном электронно-лучевом мониторе компьютера. Из очевидных достоинств такого перехвата информации можно выделить то, что засечь его невозможно (он не оставляет следов). Из недостатков же можно выделить необходимость физической близости к объекту подслушивания и использование дополнительного оборудования. В 2004 году Маркусу Куну (Markus Kuhn) удалось осуществить перехват ван Эйка для жидкокристаллических мониторов [122].

Будучи электрическими устройствами, компоненты компьютера излучают электромагнитное излучение (ЭМИ) во время выполнения операций. Расположив рядом с компонентами устройства проводник, злоумышленник способен измерить величину индукционного тока, возникающего в нём, и вычислить параметры ЭМ поля. Пример электромагнитного излучения устройства из [117] представлен на рисунке А.11. Из рисунка легко заметить, что пики ЭМИ совпадают с переходными процессами следа энергопотребления. Это говорит о схожей информативности данных каналов утечки сторонней информации. Анализ ЭМИ устройства может дать атакующему дополнительную информацию о выполняющихся вычислениях и используемых данных, что может упростить вычисление секретного параметра. Как и атаки по анализу энергопотребления, атаки по ЭМИ могут быть также разделены на две категории: простые (simple electomagnetic attack, SEMA) и дифференциальные (simple electomagnetic attack, DEMA).

Атаки по времени, энергопотреблению и ЭМИ можно рассматриваться как атаки с увеличивающимся количеством размерностей. Так, атака по времени измеряет длительности исполнения операций для каждого опыта (1 величина). Атака по энергопотреблению для анализа использует пары из длительности операции и соответствующего её энергопотребления (2 величины). Атака по ЭМИ за счёт изменения положения тестового проводника позволяет составлять трёхмерную карту ЭМИ устройства, которое к тому же будет изменяться во времени (4 величины). Это позволяет, например, раздельно рассматривать вклад различных элементов схемы в общее ЭМИ.

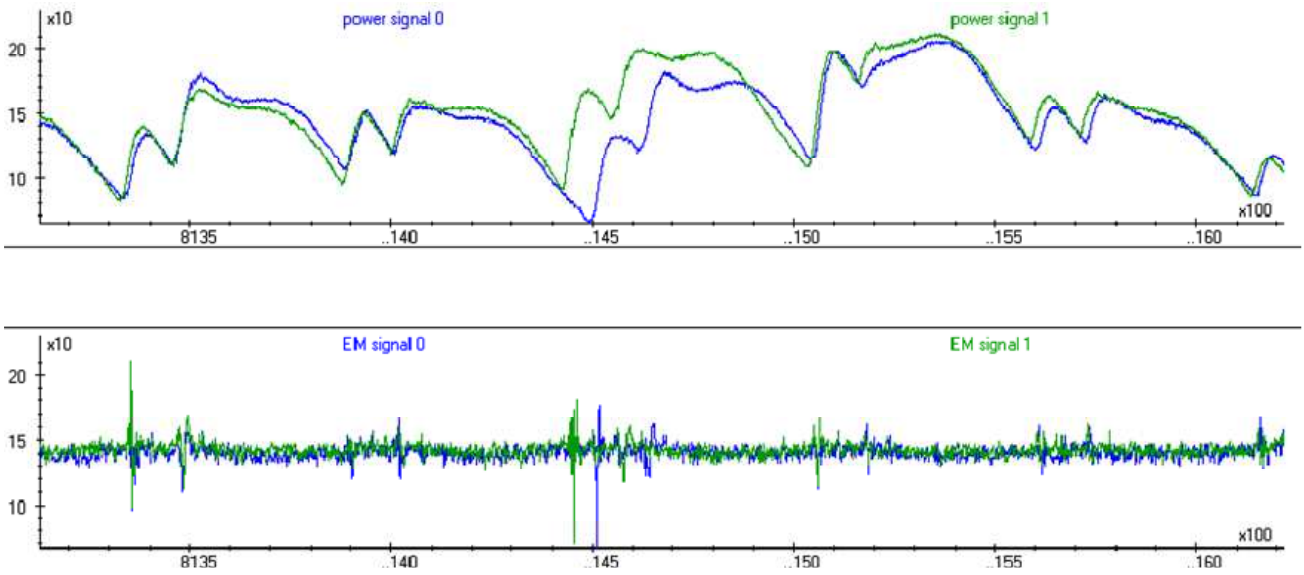


Рисунок А.11 – Сопоставление энергопотребления вычислительного устройства его электромагнитному излучению [117]

ЭМИ может быть поделено на 2 типа [123]:

– прямое излучения (direct emanations) — излучения, возникающие из-за интенсивного движения электрического тока внутри схемы;

– случайные излучения (unintentional emanations) — излучения, возникающие в результате эффекта взаимодействия частей схемы.

Последние исследования показывают, что случайные излучения, которыми активно пренебрегали раньше, могут служить источником утечки значительно большего количества информации, нежели прямые. Более того, их радиус распространения может составлять порядка 4,5 м. Также, существуют ситуации, при которых атаки по ЭМИ могут быть использованы для взлома методов защиты от атак по энергопотреблению.

Несмотря на то, что атака по ЭМИ является неагрессивной (так как она состоит в измерении ЭМ поля вблизи устройства), вскрытие устройства делает её более эффективной и устраняет возмущения, связанные с пассивирующим слоем.

Возможность использования электромагнитных излучений уже относительно давно известна в военных кругах. Например, недавно рассекреченный документ TEMPEST, изданный Национальным Агентством Безопасности (NSA), исследует различные компрометирующие излучения, включая ЭМИ, излучения проводов и распространение акустического сигнала.

Методы защиты: мерам борьбы с электромагнитным излучением посвящено большое количество литературы и нормативных документов. По принципу защиты их можно разделить на две группы: уменьшение мощности сигнала и уменьшение информативности сигнала. Методы, направленные на уменьшение мощности сигнала, заключаются в перепроектировании устройства с целью уменьшения случайного излучения, использовании экранирования и физически

защищённых зон. Методы для уменьшения информативности сигнала основаны на привнесении случайности в процессы вычислений, а также на частом обновлении ключа.

Проведение данной атаки в современных условиях развития технологий значительно усложняется. Сложности, возникающие при проведении атак по ЭМИ, аналогичны сложностям атак по энергопотреблению:

– высокие требования к частоте работы измерительного оборудования;

– снижение уровня напряжения интегральных схем приводит к уменьшению отношения сигнал/шум измеряемых данных;

– переход к данным большей размерности усложняет анализ измерений;

– усложнение схем приводит к уменьшению точности измерений;

– эффективные меры защиты для большого количества алгоритмов.

А.4.5 Атака по привнесённым помехам

Атака по привнесённым помехам (fault-injection attack, FIA) — активная атака, связанная с анализом работы устройства в непредусмотренных условиях.

В русскоязычных источниках обычно называется атакой по ошибкам вычислений.

С момента своего появления в 1997 году [59] FIA были успешно применены почти ко всем криптографическим алгоритмам. Атаки по помехам предоставляют криптоаналитику большой выбор возможных сценариев атаки системы. Методы использования помех могут сильно отличаться для разных алгоритмов. Осуществимость атаки зависит от конкретных возможностей атакующего и типа помех, которые он может индуцировать в устройстве. В общем, модель помехи должна определять как минимум следующие параметры помехи:

– точность, с которой атакующий может индуцировать помехи в определённой области устройства (пространственная разрешающая способность);

– размерность данных, подверженных помехе (например, один бит или один байт);

– скорость внедрения и постоянство существования помехи: помеха кратковременна, долговременная или постоянна (временная разрешающая способность);

– тип помехи: инверсия одного бита; одностороннее изменение значения одного бита (например, $0 \to 1$, $1 \to 1$); изменение байта данных на некоторую случайную величину; и так далее.

В общем, атаки по привнесённым помехам состоят из двух этапов. Первый заключается в наведении помех в определённый момент работы устройства. Наиболее исследованные криптографическим сообществом механизмы внедрения помех включают в себя отклонение в электропитании [50, 124–126], возбуждение силиконового слоя чипа с помощью света или лазера [49, 51, 124, 127–129] и создание индукционного тока на поверхности чипа с помощью магнитного поля [129, 130].

Вторым этапом атаки является анализ результатов воздействия помех на работу устройства с целью вычисления секретного параметра. Одним из наиболее эффективных методов анализа работы устройства в условиях помех является дифференциальный анализ ошибок (differential fault analysis, DFA) [47]. DFA был хорошо изучен с теоретической точки зрения и кажется применимым почти ко всем симметричным и асимметричным криптосистемам.

Интересным примером успешного проведения атаки с привнесением помехи является эксперимент, описанный С. П. Скоробогатовым (Sergey Skorobogatov) и Р. Дж. Андерсоном (Ross J. Anderson) [57]. Принцип индуцирования помех заключался в чувствительности полупроводниковых транзисторов к ионизирующему излучению. Такое излучение может быть вызвано ядерным взрывом, радиоактивными изотопами, рентгеновским излучением или космическими лучами. В промышленности для моделирования воздействия ионизирующего излучения на полупроводники применяются лазеры.

В своём эксперименте Скоробогатов и Андерсен в качестве источника излучения использовали фотовспышку, купленную в секонд-хэнд магазине за $30, и лазерную указку, купленную за $8. В качестве атакуемого устройства была выбрана ОЗУ стандартного микроконтроллера PIC16F84. С поверхности чипа был удалён защитный слой, после чего ОЗУ было подвержено многократному излучению. В результате такого опыта было показано, что даже используя такое недорогое и доступное оборудование, можно изменять значения конкретных ячеек памяти на требуемое. Также, с помощью метода обратного проектирования[2)] была составлена таблица соответствия адресов ячеек памяти их фактическому расположению на чипе.

Необходимо отметить, что этот микроконтроллер не являлся защищённым, поэтому из факта успешного осуществления атаки не следовало, что подобная атака может быть эффективной против специализированных устройств. Однако сам факт простоты манипулирования содержанием памяти на транзисторах уже говорит о большом потенциале такой атаки.

Развитие этого эксперимента представлено в работе [58]. В статье сотрудницы компании Samsung описывается эксперимент по воздействию лазера на работу смарт-карт, являющихся защищёнными устройствами. Облучению подвергались различные элементы смарт-карт:

– логическая часть, включающая процессор, сопроцессоры, криптографические модули, логические схемы и так далее;

[2)]Обратное проектирование (reverse engineering) - процесс систематического разбора программы (восстановления её исходного текста и структуры) или микросхемы для изучения алгоритмов её работы с целью имитации или повторения некоторых или всех её функций в другой форме или на более высоком уровне абстракции. Широко используется в современной индустрии - от чистого копирования до скрытого.

– постоянное запоминающее устройство (ПЗУ, read only memory, ROM);

– запоминающее устройство прямого доступа (ЗУПД, random access memory, RAM);

– энергонезависимая память (ЭНП, non-volatile memory, NVM).

Набор сбоев в работе устройств, полученный в результате эксперимента, представлен в таблице А.1. Как видно из таблицы, при воздействии на логические элементы устройства было спровоцировано всё множество перечисленных сбоев. Для всех типов памяти были осуществлены сбои чтения и записи. Кроме того, при воздействии лазером на типы памяти, предназначенные для хранения исполняемого кода, были также спровоцированы нарушения в последовательности выполняемых операций.

Таблица А.1 – Типы сбоев в работе смарт-карт в зависимости от объекта воздействия лазером

Тип ошибки	Логика	ПЗУ	ЗУПД	ЭНП
Ошибочные криптографические вычисления	✓			
Ошибки записи/чтения из памяти	✓	✓	✓	✓
Ошибочный адрес записи/чтения	✓			
Пропуск/повреждение инструкции	✓	✓		✓
Ошибочные вычисления процессора	✓			
Повреждение регистра	✓			

Примеры атак с простым и дифференциальным анализом помех были представлены в разделе 1.2.3. Ниже приведём описание более сложных атак на шифры DES, RSA и AES.

В работе [47] приводится тип атаки на DES с недифференциальным анализом помех (non-differential fault analysis). Формально, подобный метод анализа является простым. Описываемая атака подразумевает физическое повреждение регистров, ячеек памяти или проводов (например, с помощью лазера), в результате чего соответствующие повреждённым блокам значения рассматриваются как некоторые фиксированные константы (например, ноль), и более не могут влиять на вычислительный процесс. Данная атака является атакой на основе шифротекста. Для её осуществления необходимо наличие набора шифротектов, вычисленных устройством с заданными искажениями (описаны ниже), или же наличие такого устройства с искомым ключом у криптоаналитика. Приведём описание данной атаки.

Рассматривается аппаратная реализация алгоритма DES, в которой один раунд, реализованный на чипе согласно рисунку А.12, используется 16 раз. Через L и R обозначены левый и правый регистры, а F есть функция шифрования. Злоумышленник внедряет помеху в наименее значимый бит (least significant bit, LSB) левого регистра L_i, $1 \leq i \leq 16$. Внедрение искажения осуществляется уничтожением регистра или отсоединением провода, входящего или выходящего из этого регистра; без ограничения общности предполагается, что значение этого бита

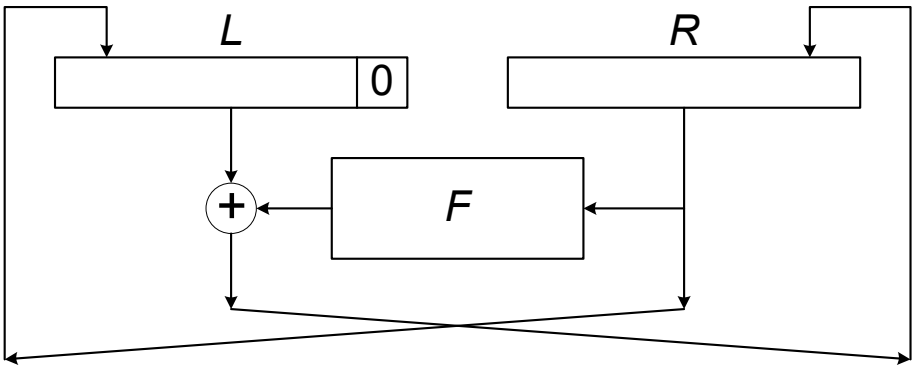


Рисунок А.12 – Общая схема аппаратной реализации раунда DES

фиксируется равным нулю. Рисунок А.13 демонстрирует значения, посчитанные на последнем этапе алгоритма. Для любого шифротекста наименее значимый бит левой половины (L_{16}) равен

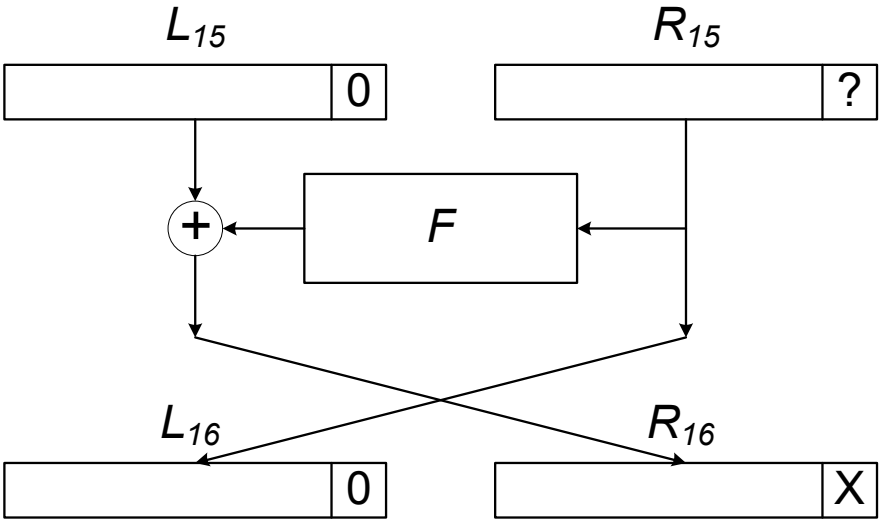


Рисунок А.13 – Заключительный раунд DES

нулю, также нулю равен наименее значимый бит L_{15}. LSB выходного значения функции шифрования F последнего раунда равен LSB правой части шифротекста. Этот бит является выходом блока подстановки S7, используемого в блоке, реализующем функцию шифрования. Вход блока S7 образован побитовым сложением по модулю два шести известных битов шифротекста (они содержатся в L_{16}) и шести битов ключа. Далее, криптоаналитик перебирает все 64 возможные варианта значений шести битов ключа, отбрасывая все значения, которые не приводят к LSB правой половины шифротекста. Каждый шифротекст позволяет отбросить из рассмотрения примерно половину оставшихся ключей. Таким образом, имея шесть шифротекстов (об открытых текстах которых криптоаналитик не имеет никакой информации), может быть выявлено значение шести битов секретного ключа. Оставшиеся биты подключа последнего раунда могут быть найдены, например, за счёт внедрения дополнительных помех.

Кроме этого, в этой работе приводятся и другие атаки с простым анализом помех, основанные на различных вариантах реализации алгоритма DES.

Приведём также примеры дифференциального анализа ошибок шифра RSA, реализованного без использования китайской теоремы об остатках [131]. Обозначим двоичное представление секретной экспоненты d как

$$d = (d(t-1)|d(t-2)|\ldots|d(i)|\ldots|d(0)),$$

где $d(i)$ есть значение i-ого бита d и принимает значения 0 или 1;

t есть размер экспоненты в битах;

через «|» обозначена операция конкатенации.

Пусть e есть открытая экспонента, а $C = M^e \bmod n$ — шифротекст. Обозначим через $C(0) = C$, $C(1) = C^2 \bmod n$, $C(2) = C^{2^2} \bmod n, \ldots, C(t-1) = C^{2^{t-1}} \bmod n$. Имея C и d, соответствующий открытый текст может быть представлен как

$$M = C(t-1)^{d(t-1)} \cdot C(t-2)^{d(t-2)} \cdot \ldots \cdot C(i)^{d(i)} \cdot \ldots \cdot C(0)^{d(0)} \pmod{n}.$$

Первый вариант атаки. Криптоаналитик произвольно выбирает открытый текст M и вычисляет шифротекст C. Далее, злоумышленник оказывает нестандартное физическое воздействие на чип, провоцируя искажение секретной экспоненты, при этом запуская на устройстве дешифрацию C. Предположим, что один бит экспоненты d изменил значение на противоположное, номер искажённого бита неизвестен. Обозначая искажённый бит через $d(i)'$, $i = 0, \ldots, t-1$, получаем, что выходом устройства при условии наличия искажения станет

$$M' = C(t-1)^{d(t-1)} \cdot C(t-2)^{d(t-2)} \cdot \ldots \cdot C(i)^{d(i)'} \cdot \ldots \cdot C(0)^{d(0)} \pmod{n}.$$

Обладая M и M', криптоаналитик может вычислить

$$M'/M = C(i)^{d(i)'}/C(i)^{d(i)} \pmod{n}.$$

Если $M'/M = C(i) \bmod n$, тогда $d(i) = 0$, в противно случае $M'/M = 1/C(i) \bmod n$ и $d(i) = 1$. Злоумышленник способен заранее посчитать значения $C(i)$ и $1/C(i)$ для $i = 0, ..., t-1$ и, сравнивая их с $M'/M \bmod n$, выявить значения отдельных битов d. Данная процедура повторяется до тех пор, пока криптоаналитик не получит достаточное количество информации о значении секретной экспоненты.

Второй вариант атаки. Для простоты предположим, что при дешифрации атакуемое устройство сначала вычисляет значения $C(i)$, а затем использует их при вычислении открытого текста

$$M = C(t-1)^{d(t-1)} \cdot C(t-2)^{d(t-2)} \cdot \ldots \cdot C(i)^{d(i)} \cdot \ldots \cdot C(0)^{d(0)} \pmod{n}.$$

Допустим, злоумышленник исказил один бит в значении $C(i)$, $i = 0, \ldots, t-1$. Обозначим искажённое значение через $C(i)'$. Следовательно, результатом дешифрации станет следующий текст

$$M' = C(t-1)^{d(t-1)} \cdot C(t-2)^{d(t-2)} \cdot \ldots \cdot C(i)'^{d(i)} \cdot \ldots \cdot C(0)^{d(0)} \pmod{n}.$$

Атакующий вычисляет

$$M'/M = C(i)'^{d(i)}/C(i)^{d(i)} \pmod{n} = C(i)'/C(i) \pmod{n}$$

(необходимо отметить, что $d(i)$ в этом соотношении должен быть равен 1, в противном случае $M'/M = 1$). Как и в предыдущей атаке, злоумышленник может заранее посчитать все возможные отношения $C(i)'/C(i) \bmod n$. Найдя соответствие между посчитанным значением и величиной $M'/M \bmod n$, криптоаналитик определяет номер позиции i, на которой произошло искажение, и выявляет, что $d(i) = 1$. Данная процедура повторяется до полного выявления значения секретной экспоненты.

Приведём также один из сценариев атаки на AES-128 (AES с ключом размера 128 бит) с последующим дифференциальным анализом помех из [61] с сохранением обозначений. Для начала опишем сам алгоритм шифрования. AES это симметричный алгоритм блочного шифрования с длиной блока равным 128 битам, поддерживающий ключи размерами 128, 192 и 256 бит. Шифр является итеративным и состоит в выполнении одинаковых последовательностей действий — раундов — над текущим состоянием шифра (State). Стоит отметить, что последний раунд отличается от остальных отсутствием одной операции, а перед первым раундом выполняется дополнительная операция. Количество раундов обозначается N_r и зависит от длины ключа ($N_r = 10$ для 128 бит, $N_r = 12$ для 192 бит и $N_r = 14$ для 256 бит). AES выполняет преобразование состояния, обозначаемого $S \in M_4(GF(2^8))$ (то есть S есть квадратная матрица размера 4x4 с коэффициентами из конечного поля $GF(2^8)$), в некоторое другое состояние в $M_4(GF(2^8))$. Ключ K расширяется в $N_r + 1$ подключей, обозначаемых $K_i \in M_4(GF(2^8))$, $i = 0, 1, \dots N_r$. Раунд шифрования AES состоит из четырёх основных операций:

– подстановка байтов (SubBytes);

– сдвиг строк (ShiftRows);

– смешивание столбца (MixColumn);

– прибавление ключа раунда (AddRoundKey).

Операция подстановки байтов заключается в осуществлении преобразования s над элементами состояния S. Обозначим через $S_{i,Su}$ состояние после i-ой подстановки:

$$M_4(GF(2^8)) \to M_4(GF(2^8)),$$

$$S = \begin{pmatrix} S[1] & S[5] & S[9] & S[13] \\ S[2] & S[6] & S[10] & S[14] \\ S[3] & S[7] & S[11] & S[15] \\ S[4] & S[8] & S[12] & S[16] \end{pmatrix} \mapsto S_{i,Su} = \begin{pmatrix} s(S[1]) & s(S[5]) & s(S[9]) & s(S[13]) \\ s(S[2]) & s(S[6]) & s(S[10]) & s(S[14]) \\ s(S[3]) & s(S[7]) & s(S[11]) & s(S[15]) \\ s(S[4]) & s(S[8]) & s(S[12]) & s(S[16]) \end{pmatrix},$$

где s есть нелинейное преобразование:

$$GF(2^8) \to GF(2^8),$$

$$x \mapsto s(x) = \begin{cases} a * x^{-1} + b, & \text{if } x \neq 0, \\ b, & \text{if } x = 0; \end{cases}$$

$a \in M_8(GF(2))$ есть матрица над GF(2);

$*$ есть операция умножения матриц над $GF(2)$;

x^{-1} есть вычисление обратного по умножению элемента в $GF(2^8)$;

константа b есть элемент поля $GF(2^8)$ с шестнадцатеричным значением 63_{16}.

$$a = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Операция сдвига строк представляет собой циклический сдвиг строк матрицы состояния на разные значения. Обозначим через $S_{i,Sh}$ состояние после i-ого сдвига строки:

$$M_4(GF(2^8)) \to M_4(GF(2^8)),$$

$$S = \begin{pmatrix} S[1] & S[5] & S[9] & S[13] \\ S[2] & S[6] & S[10] & S[14] \\ S[3] & S[7] & S[11] & S[15] \\ S[4] & S[8] & S[12] & S[16] \end{pmatrix} \mapsto S_{i,Sh} = \begin{pmatrix} S[1] & S[5] & S[9] & S[13] \\ S[6] & S[10] & S[14] & S[2] \\ S[11] & S[15] & S[3] & S[7] \\ S[16] & S[4] & S[8] & S[12] \end{pmatrix}.$$

Операция смешивания столбца заключается в умножении матрицы состояния на фиксированную матрицу $A_0 \in M_4(GF(2^8))$. Обозначим через $S_{i,M}$ состояние после i-ого смешивания столбцов:

$$M_4(GF(2^8)) \to M_4(GF(2^8)),$$
$$S \mapsto S_{i,M} = A_0 * S,$$

где A_0 задаётся следующим набором чисел в шестнадцатеричном представлении:

$$A_0 = \begin{pmatrix} 02_{16} & 03_{16} & 01_{16} & 01_{16} \\ 01_{16} & 02_{16} & 03_{16} & 01_{16} \\ 01_{16} & 01_{16} & 02_{16} & 03_{16} \\ 03_{16} & 01_{16} & 01_{16} & 02_{16} \end{pmatrix};$$

через $*$ в данном случае обозначено перемножение матриц над $GF(2^8)$.

Операция прибавления ключа раунда заключается в сложении текущего состояния и ключа раунда. Состояние после прибавления ключа i-ого раунда будем обозначать как $S_{i,A}$:

$$M_4(GF(2^8)) \to M_4(GF(2^8)),$$
$$S \mapsto S_{i,A} = S + K_i.$$

Суть описываемой атаки заключается в изменении значения одного байта состояния после операции сдвига строк на $N_r - 1$ раунде, позиция искажённого байта считается известной (данное допущение используется для простоты объяснения принципа атаки и может отсутствовать). Новое значение искажённого байта считается неизвестным. Привнесённая помеха e оказывает влияние на четыре байта выходного состояния шифра. Для каждого искажённого байта текущего шифротекста вычисляется множество возможных значений помехи e. Над полученными для каждого искажённого байта множествами можно выполнить операцию пересечения, тем самым уменьшив итоговое количество потенциальных вариантов помехи e и, как следствие, количество требуемых искажённых шифротекстов для полного анализа. Заключительным этапом анализа является определение возможных значений четырёх байтов ключа десятого раунда для текущей помехи e. Повторяя это действие для набора шифротекстов, получаем значение четырёх байтов ключа последнего раунда.

Рассмотрим пример влияния помехи e на первый байт состояния шифра. Искажённое состояние будем обозначать через F. Привнесение помехи можно записать как

$$F_{N_r-1,Sh} = S_{N_r-1,Sh} + \begin{pmatrix} e & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

После операции смешивания столбца получаем

$$F_{N_r-1,M} = S_{N_r-1,M} + A_0 * \begin{pmatrix} e & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} = S_{N_r-1,M} + \begin{pmatrix} 2e & 0 & 0 & 0 \\ e & 0 & 0 & 0 \\ e & 0 & 0 & 0 \\ 3e & 0 & 0 & 0 \end{pmatrix}.$$

После прибавления ключа раунда искажённое состояние шифра может быт записано следующим образом

$$F_{N_r-1,A} = S_{N_r-1,A} + \begin{pmatrix} 2e & 0 & 0 & 0 \\ e & 0 & 0 & 0 \\ e & 0 & 0 & 0 \\ 3e & 0 & 0 & 0 \end{pmatrix}.$$

Далее начинается выполнение последнего раунда шифра, который отличается от предыдущих отсутствием операции смешивания столбца. Последний раунд начинается с операции подстановки байтов

$$F_{N_r,Su} = S_{N_r,Su} + \begin{pmatrix} e'_0 & 0 & 0 & 0 \\ e'_1 & 0 & 0 & 0 \\ e'_2 & 0 & 0 & 0 \\ e'_3 & 0 & 0 & 0 \end{pmatrix},$$

где через e'_0, e'_1, e'_2, e'_3 обозначены дифференциальные ошибки.

Последняя операция сдвига строк приводит к следующему виду состояния шифра:

$$F_{N_r,Sh} = S_{N_r,Sh} + \begin{pmatrix} e'_0 & 0 & 0 & 0 \\ 0 & 0 & 0 & e'_1 \\ 0 & 0 & e'_2 & 0 \\ 0 & e'_3 & 0 & 0 \end{pmatrix}.$$

Заключительной операцией процесса шифрования является прибавление ключа последнего раунда; состояние шифра может быть записано как

$$F_{N_r,A} = S_{N_r,A} + \begin{pmatrix} e'_0 & 0 & 0 & 0 \\ 0 & 0 & 0 & e'_1 \\ 0 & 0 & e'_2 & 0 \\ 0 & e'_3 & 0 & 0 \end{pmatrix}.$$

$F_{N_r,A}$ представляет собой искажённый выход шифратора (искажённый шифротекст). Сравнивая состояния $F_{N_r,A}$ и $S_{N_r,A}$, легко вычислить значения ошибок e'_0, e'_1, e'_2 и e'_3.

Единственной операцией, которая может выявить информацию о значении подключа K_{N_r}, является последняя операция подстановки байтов. Таким образом, необходимо решить систему уравнений относительно x_0, x_1, x_2, x_3 и e:

$$\begin{cases} s(x_0 + 2e) = s(x_0) + e'_0 \\ s(x_1 + e) = s(x_1) + e'_1 \\ s(x_2 + e) = s(x_2) + e'_2 \\ s(x_3 + 3e) = s(x_3) + e'_3 \end{cases}. \tag{A.1}$$

Для краткости опустим подробности решения данной системы уравнений, вся необходимая теория и примеры приведены в статье. Вычислив значения x_0, x_1, x_2, x_3 и e, легко получить множества возможных значений байтов подключа K_{N_r}, на которых произошли ошибки: $K_{N_r}[0]$, $K_{N_r}[7]$, $K_{N_r}[10]$, $K_{N_r}[13]$. Далее, повторяя процедуру внедрения помех и их анализа, криптоаналитик получает точные значения четырёх байтов ключа последнего раунда, выполняя операцию пересечения множеств возможных значений. Очевидно, что внедряя помехи на других позициях, криптоаналитик способен восстановить ключ последнего раунда шифрования целиком.

Приведём наглядный пример описанной атаки из [61]. Все значения ниже представлены в шестнадцатеричном формате.

Открытый текст: «32 43 F6 A8 88 5A 30 8D 31 31 98 A2 E0 37 07 34».

Ключ: «2B 7E 15 16 28 AE D2 A6 AB F7 15 88 09 CF 4F 3C».

Шифротекст: «39 25 84 1D 02 DC 09 FB DC 11 85 97 19 6A 0B 32».

Ниже приведены состояния шифра и значения подключей после операций шифрования, начиная со сдвига строк девятого раунда; в скобках указан номер раунда. Выделенные серым цветом ячейки таблиц содержат искажённые значения.

Сдвиг строк (9)

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

Внедрение помехи 1E

99	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

Смешивание столбцов (9)

7B	40	A3	4C
29	D4	70	9F
8A	E4	3A	42
CF	A5	A6	BC

$\oplus$

Ключ раунда (9)

AC	19	28	57
77	FA	D1	5C
66	DC	29	00
F3	21	41	6E

Прибавление подключа (9)

D7	59	8B	1B
5E	2E	A1	C3
EC	38	13	42
3C	84	E7	D2

Подстановка байтов (10)

0E	CB	3D	AF
58	31	32	2E
CE	07	7D	2C
EB	5F	94	B5

Сдвиг строк (10)

0E	CB	3D	AF
31	32	2E	58
7D	2C	CE	07
B5	EB	5F	94

$\oplus$

Ключ раунда (10)

D0	C9	E1	B6
14	EE	3F	63
F9	25	0C	0C
A8	89	C8	A6

Искажённый шифротекст

DE	02	DC	19
25	DC	11	3B
84	09	C2	0B
1D	62	97	32

Значения дифференциальных ошибок e'_0, e'_1, e'_2 и e'_3 вычисляются сложением шифротекста с искажённым шифротекстом:

Искажённый шифротекст

DE	02	DC	19
25	DC	11	3B
84	09	C2	0B
1D	62	97	32

$\oplus$

Шифротекст

39	02	DC	19
25	DC	11	6A
84	09	85	0B
1D	FB	97	32

$=$

Ошибки

E7	00	00	00
00	00	00	51
00	00	47	00
00	99	00	00

Получаем, что $e'_0 = E7$, $e'_1 = 51$, $e'_2 = 47$, $e'_3 = 99$. Далее, необходимо решить систему уравнений А.1, подставив вычисленные значения дифференциальных ошибок :

$$\begin{cases} s(x_0 + 2e) = s(x_0) + E7 \\ s(x_1 + e) = s(x_1) + 51 \\ s(x_2 + e) = s(x_2) + 47 \\ s(x_3 + 3e) = s(x_3) + 99 \end{cases}.$$

Множество возможных значений ошибки e, являющихся решением данной системы уравнений, обозначим через S_e.

$S_e = \{$01, 04, 13, **1E**, 21, 27, 33, 3B, 48, 4D, 50, 53, 55, 5D, 64, 65, 7E, 7F, 80, 83, 8D, 8F, 93, A7, A8, A9, AB, B3, B8, C9, F6$\}$.

Из этого можно получить, что первый байт ключа последнего раунда $K_{10}[0]$ может принимать следующие значения (настоящее значение $K_{10}[0]$ равно D0):

$K_{10}[0] \in$ {03, 06, 09, 0C, 10, 15, 1A, 1F, 21, 24, 2B, 2E, 32, 37, 38, 3D, 43, 46, 49, 4C, 50, 55, 5F, 61, 6B, 6E, 72, 77, 78, 7D, 83, 86, 89, 8C, 90, 95, 9A, 9F, A1, A4, AB, AE, B2, B7, B8, C3, C6, C9, CC, **D0**, D5, DA, DF, E1, E4, EB, EE, F2, F7, F8, FD}.
Внедрив пять ошибок {1E, E1, B3, 16, 9E}, криптоаналитик способен выявить корректные и единственные значения четырёх байтов ключа последнего раунда $K_{10}[0]$, $K_{10}[7]$, $K_{10}[10]$ и $K_{10}[13]$.

Представленные примеры дифференциального анализа помех применительно к симметричным шифрам (на примере DES и AES) и к шифрам с открытым ключом (на примере различных реализаций RSA) демонстрируют уязвимость современных методов защиты информации к атакам с привнесением помех. При этом, наличие большого количества вариантов атак на конкретный алгоритм (с внедрением помех в различные блоки устройства) приводит к необходимости комплексной защиты модуля. Таким образом, в области проектирования устройств, устойчивых к привнесённым помехам, остро встаёт вопрос обеспечения целостности обрабатываемой и хранимой информации, что будет способствовать повышению надёжности функционирования данных устройств.

Методы защиты: при допущении, что злоумышленник не способен провоцировать одинаковую ошибку дважды, естественным способом защиты представляется дублирование оборудования или повторение процедуры шифрования. Сравнение результатов покажет наличие ошибки.

Для защиты систем с открытым ключом существует метод, при котором перед отправкой электронной цифровой подписи предполагается её проверка с помощью открытого ключа.

Другими методами защиты устройств являются: использование контрольных сумм и методов помехоустойчивого кодирования, привнесение случайности в процесс исполнения операций и многое другое.

На данный момент атаки по помехам представляют серьёзную опасность для криптографических устройств. Несмотря на большое количество предлагаемых способов защиты, многие из них не могут обеспечить достаточный уровень защиты при сильной модели атаки (высокое пространственное и временное разрешение, провоцирование конкретных значений ошибок и так далее).

A.4.6 Атака по видимому излучению

Атака по видимому излучению (visible light attack) — пассивная атака, предложенная Маркусом Куном в 2002 году [132]. В своей работе он показал, что, используя высокоточный датчик интенсивности света, можно измерить изменения в интенсивности рассеянного от монитора света (например, отражённого от стены), и таким образом восстановить изображение на экране. Даже при условии, что устройство не находится в зоне прямой видимости атакующего, злоумышленник способен получать информацию от вычислительного устройства. Таким образом, даже защиты от утечки информации через ЭМИ может оказаться недостаточно. Как и в случае

атаки по ЭМИ, особенностью атаки по видимому излучению является то, что она не требует физического доступа к устройству и не может быть обнаружена.

Данный тип атак также можно применить к шифраторам, использующим светодиодные индикаторы, анализируя данные от которых, можно получить информацию об операциях в устройстве [133]. Также, в этой работе представлена систематика методов защиты от утечки информации через видимое излучение, которые позволяют блокировать соответствующие атаки. Основным методом защиты от данной атаки является ликвидация видимого излучения, например, с помощью помещения оборудования в непрозрачные корпуса.

На данный момент описанная атака не получила развития.

А.4.7 Акустическая атака

Акустическая атака (acoustic attack) — пассивная атака, направленная на получение информации из звуков, производимых устройством. Исторически данный тип атак связывается с прослушкой работы шифровальных устройства, принтеров и клавиатур (пример из А.1 с прослушкой роторной шифровальной машины), но в последние годы были найдены уязвимости, позволяющие использовать акустические атаки, направленные на внутренние компоненты электронных шифраторов. Так, Шамир (Shamir) и Трэмер (Tramer) в [134] продемонстрировали предварительное доказательство того, что существует корреляция между звуком процессора и производимыми им вычислениями.

Результат проведённого ими эксперимента показал, что звуковой сигнал, производимый процессором, не искажается шумом от вентиляторов системы охлаждения и сторонних объектов. Кроме того, было выявлено, что наибольшим количеством шума сопровождалось состояние покоя процессора, вызываемое командой HALT. Таким образом, упрощается определение моментов начала и конца работы процессора. Пример спектра звукового сигнала из [134] представлен на рисунке А.14.

На представленном рисунке изображён спектр звука, производимый процессором, при вычислении электронной цифровой подписи по алгоритму RSA. Основными этапами данного алгоритма являются выполнения вычислений по модулям p и q, которые в данном примере заключены между командами HALT. Таким образом, обладая знаниями о процессоре и алгоритме, криптоаналитик на основе звука, производимого процессором, может различать различные этапы алгоритма, а также выделять одинаковые этапы на основе идентичности звукового сигнала.

Данная атака пока не получила дальнейшего развития.

А.4.8 Атака по кэш

Атака по кэш (cache-based attack) — пассивная неагрессивная атака, основанная на анализе промахов кэш-памяти процессора. Современные компьютеры используют процессорный кэш (CPU cache), или просто кэш, между процессором и основной памятью для увеличения средней скорости работы. В случае, если процессору требуется информация, которая не хранится в кэше,

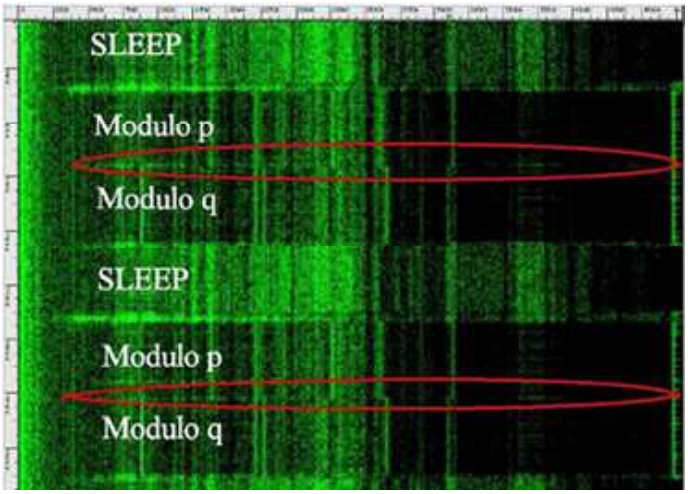


Рисунок А.14 – Пример спектра звукового сигнала, производимого процессором, из [134]

то есть произошёл промах кэша (cache miss), то загрузка данных из основной памяти в кэш будет сопровождаться задержкой.

Атаки по кэш делятся на 2 вида [135]:

– атаки на основе следа (trace driven attacks);

– атаки на основе времени (time driven attacks).

Атаки по кэш на основе следа основаны на способности злоумышленника получать и анализировать профиль обращения процессора к кэш при исполнении криптографического алгоритма [136]. Для успешного осуществления атаки злоумышленнику необходимо иметь след (попадание или промах кэш) для каждого обращения к памяти. Рассмотрим принцип этой атаки на примере блоков подстановки (S-box) алгоритма DES, реализованных через таблицы преобразований[3)] (look-up table). Пусть след обращения к кэш-памяти для 3 раундов алгоритма выглядит следующим образом:

$$MMMMMMMM \qquad HHMMMMMH \qquad HMHHMMHM,$$

где M обозначает промах кэш;

H — попадание.

Для представленного примера криптоаналитик может сделать предположение о том, что входные значения первого блока подстановки на первом и втором раундах совпадают. Таким

[3)]Таблица преобразования - структура данных, обычно массив или связанный массив, зачастую используется для замены прямых вычислений на простую операцию индексирования. По значению входного аргумента вычисляется номер ячейки таблицы, содержащий корректное выходное значение.

образом, модифицируя открытый текст и анализируя получаемые при его шифровании профили обращения к кэш, криптоаналитик может выявить значения секретных параметров алгоритма.

Атаки по кэш на основе времени являются частным случаем атак по времени [42, 102]. Описанные раннее атаки по времени использовали тот факт, что условные переходы в алгоритме шифрования могут приводить к различиям во времени выполнения операции. Недостаток кэш памяти, при котором процессору приходится обращаться к основной памяти, также способен провоцировать временные различия. Измерение этой задержки может позволить атакующему определить появление и частоту промахов кэша. Из этого возникает сторонний канал.

Оригинальная идея использования кэш памяти как стороннего канала, через который происходит утечка информации в процессе исполнения криптографического алгоритма, была предложена Келси (Kelsey) и другими в [137]. Атаки по кэш были успешно применены к программным реализациям шифров MISTY1, DES, Camelis и др. [138, 139] Идея атаки на кэш была расширена на случай блоковых шифров, использующих подстановки и перестановки, например, AES [140]. Этот подход может быть применён для взлома криптографических примитивов, которые реализуют функции с помощью таблиц преобразования. Атака по кэшу является достаточно сильным типом атак по сторонним каналам, так как не нуждается ни в знании специального открытого текста, ни шифротекста, и проводится простым мониторингом воздействия криптографического процесса на кэш.

Методы защиты: среди методов защиты от атак по кэш можно перечислить следующие: убрать кэш или доступ к кэш со стороны блоков подстановки (S-box), отключить очистку кэша, осуществлять временное смещение операций, использовать аппаратную архитектуру разделённого кэша. Кроме того, возможна полная или случайно–частичная загрузка таблиц подстановки в кэш-память.

На данный момент угроза от атаки по кэш не устранена. В [140] утверждается, что примитивы, реализованные без таблиц преобразования, такие как SHA и Serpent, невосприимчивы к описанному типу атак. В то же время нахождение эффективных решений, которые будут независимы от алгоритмов и архитектуры, до сих пор остаётся открытой проблемой.

A.4.9 Атака по частоте

Атака по частоте (frequency-based attack) — дифференциальная атака против мобильных вычислительных устройств таких как PDA, мобильные телефоны и пейджеры, предложенная Тию (C. C. Tiu) в [141].

Эта атака в некотором смысле аналогична дифференциальным атакам по энергопотреблению и ЭМИ, но производится в частотной области. Атака заключается в вычислении дифференциальной спектральной плотности энергии сигнала. Она эффективна даже при условии смещения следов результатов экспериментальных атак, в то время как описанные выше атаки по ЭМИ оказываются неэффективными (за счет сдвигов во временной области теряются корреляционные

свойства сигналов). Такая особенность позволяет атаке первого порядка по частоте преодолевать десинхронизирующие контрмеры, вводящие случайные задержки.

Атака по частоте к на данный момент не получила дальнейшего развития.

А.4.10 Атака по сканированию

Атака по сканированию (scan–based attack) — эффективная атака, использующая встроенные в аппаратные схемы сканирующие тесты. Тесты, основанные на сканировании, являются крайне эффективной методикой тестирования аппаратуры. Однако, в [142] было показано, что они также являются мощным инструментом для проведения атаки по сторонним каналам. Такие тесты были использованы для извлечения секретных ключей в аппаратных реализация алгоритма DES.

На данный момент в США предприняты попытки по стандартизации тестов для защищённых устройств. В документе [143] [4)] рекомендуется использование определённого вида встроенных самотестирующихся схем для гарантии защиты криптографических схем на физическом уровне. Однако высокое покрытие ошибок с помощью тестов, основанных на сканировании, является серьёзным поводом для проектирования защищённых сканирующих тестов для криптографических чипов.

А.5 Вывод

В данном приложении к диссертационной работе были представлены основные типы атак по сторонним каналам (остались без внимания, например, атаки по сообщениям об ошибках [144] и остаточная информация [145]). Однако даже краткий обзор этих атак даёт представление о той угрозе, которую они представляют для реализаций криптографических алгоритмов.

Как было сказано, для многих существующих типов атак уже предложены методы защиты, позволяющие ослабить либо полностью устранить угрозу их успешного проведения при условии относительно небольших аппаратных/временных расходов. Однако исследование сторонних каналов является быстро развивающимся направлением криптологии, результатом чего является непрерывное развитие и усовершенствование инструментов криптоанализа. Постоянное изучение существующих угроз и создание контрмер становится необходимым требованием для обеспечения защиты конфиденциальных данных.

Одной из наиболее эффективных атак является атака по привнесённым помехам. Сочетание атак по привнесённым помехам и аппаратов простого и дифференциального анализа помех были успешно применены для вычисления секретных параметров многих существующих криптографических алгоритмов, включая DES, AES и RSA. Атака не теряет свою эффективность даже при такой реализации криптоалгоритма, когда работа устройства не коррелирована с обрабатываемыми данными и выполняемыми операциями.

[4)] FIPS 140 - стандарт, используемый федеральными организациями США при спецификации криптографических систем для обеспечения защиты важных данных.

Наличие большого диапазона методов внедрения помех в работу криптографического модуля приводит к значительному усложнению требуемых контрмер. Из этого следует необходимость разработки универсальных методов защиты устройств от помех вне зависимости от их происхождения. Данная диссертационная работа посвящена исследованию и совершенствованию существующих методов защиты от этой атаки.