УДК 004.056:061.68;004.375:061.68
КП

№ госрегистрации 0113U000363

Инв.

Министерство образования и науки Украины
Харьковский национальный университет радиоэлектроники
(ХНУРЭ)
61166, г. Харьков, пр. Ленина, 14;
тел. (057) 702 13 97

УТВЕРЖДАЮ
Проректор по научной работе
д.ф-м. н., профессор
_____ Н.И. Слипченко

2014 г.

ОТЧЕТ
О НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ
«Анализ состояния, определение направлений развития, стандартизация, усовершенствование, разработка и внедрение криптографических систем, включая систему ЕЦП »

(заключительный)

Руководитель НИР
профессор кафедры
безопасности информационных технологий
д.т.н, профессор И.Д. Горбенко

(подпись)

(дата)

2014

Рукопись окончена 2014 г.
Результаты этой работы рассмотрены Ученым Советом ХНУРЭ,
протокол № ___ от «__»_____20__ г.

СПИСОК АВТОРОВ

Руководитель НИР, д.т.н., проф.	И. Горбенко (раздел 1-4)
Ответственный исполнитель, д.т.н., проф.	Г. Халимов (раздел 1-4)
Ведущий научный сотрудник, д.т.н., проф.	И. Лисицкая (раздел 3-4)
Ведущий научный сотрудник, д.т.н., проф.	В. Долгов (раздел 3-4)
Старший научный сотрудник, к.т.н.	Ю. Горбенко (раздел 3-4)
Старший научный сотрудник, к.т.н.	В. Руженцев (раздел 3-4)
Старший научный сотрудник, д.т.н., проф.	Е. Винокурова (раздел 1-2)
Старший научный сотрудник, д.т.н.	Р. Олейников (раздел 3-4)

Младший научный сотрудник, к.т.н.	Т. Гриненко (раздел 1-2)
Младший научный сотрудник, к.т.н.	А. Бойко (раздел 1-2)
Младший научный сотрудник, к.т.н.	О. Мельникова (раздел 1-2)
Младший научный сотрудник	А. Шумов (раздел 1-2)
Младший научный сотрудник	А.Тоцкий (раздел 1-2)
Младший научный сотрудник	Р. Мордвинов (раздел 1-2)
Младший научный сотрудник	Л. Макутонина (раздел 1-2)
Младший научный сотрудник	А. Казимиров (раздел 3-4)
Младший научный сотрудник, к.т.н.	В. Бобух (раздел 3-4)

Инженер 2 кат. С. Чичмарь (раздел 3-4)

Старший научный сотрудник, к.т.н., проф. А. Замула (раздел 6-7)

Студент К. Лисицкий (раздел 3-4)

Аспирант В. Черныш (раздел 6-7)

Аспирант Д. Семченко (раздел 6-7)

Н.с. Т. Мельник (раздел 1)

Н.с. Е. Колованова (раздел 3)

Н.с. Д. Иваненко (раздел 3)

Заведующий каф. ПМ, д.т.н., профессор А. Тевяшев (раздел 5)

Старший научный сотрудник	А.Долгоброд (раздел 5)
Ведущий научный сотрудник	А. Карпухин (раздел 5)
Н.с.	Е. Выходцев (раздел 5)
Младший научный сотрудник	Ю. Асаенко (раздел 5)
Ст.-исс., каф. ПМ	Т. Долотцева (раздел 5)
Профессор, д.т.н.	А. Кузнецов (раздел 3-4)
Доцент, к.т.н.	А. Северинов (раздел 1-2)
Докторант	К. Погребняк (раздел 1-2)

Доцент, к.т.н.	Д. Балагура (раздел 1-2)
Аспирант	И. Аулов (раздел 1-2)
Аспирант	О. Товма (раздел 1-2)
Аспирант	Д. Кайдалов (раздел 1-2)
Аспирант	Д. Повтарев (раздел 1-2)
Аспирант	Д. Маковейкий (раздел 1-2)
Докторант	А. Леншин (раздел 1-2)

РЕФЕРАТ

Отчет о НИР: стр., рис., табл., источников.

Предметом исследования являются криптографические системы и инфраструктуры электронных доверительных услуг открытых ключей, включая национальную систему ЭЦП и электронных доверительных услуг, методы, механизмы, криптографические протоколы и их стандартизация.

Объектом исследований являются процессы, касающиеся анализа состояния, определения направлений развития, стандартизации, создания, совершенствования, разработки и внедрения электронной системы трансграничных электронных доверительных услуг, включая систему ЭЦП на государственном и международном уровнях, в том числе Европейского Союза (ЕС) .

Цель научно-исследовательской работы. Целью НИР является анализ, теоретическое обоснование унификации, стандартизации и развития методов и механизмов криптографической защиты информации для предоставления электронных трансграничных доверительных услуг, совершенствование и / или разработка методов и криптографических примитивов блочного симметричного и асимметричного шифрования, ЭЦП и хеширования, а также средств их реализации и оценки свойств, с ориентацией на предоставление трансграничных электронных доверительных услуг.

Разработанные методы синтеза и анализа перспективных национальных стандартов функции хеширования, блочного симметричного шифра (преобразования) и направленного шифрования позволили создать национальные стандарты на функцию хеширования и блочное симметричное преобразование, соответствующие международным требованиям. Криптографические и табличные преобразования, применяемые в шифре, отвечают самым современным требованиям к уровню криптографической стойкости и быстродействия.

Блочный симметричный шифр ГОСТ 7624: 2014 и функция хеширования ДСТУ 7564: 2014 являются гибкими, поддерживают размер блока и длину ключа от 128 до 512 бит, что является уникальным в мире. Сейчас такие параметры еще не поддерживается ни одним из существующих международных, региональных и национальных криптографических примитивов. По сравнению с известного международного стандарта AES (ISO / IEC 18033-3: 2010), Национальный стандарт Украины ДСТУ 7624: 2014 обеспечивает высокий уровень криптографической стойкости (с возможностью применения блока и ключа шифрования вплоть до 512 бит) и аналогичную или более высокое быстродействие на современных и перспективных программных и программно-аппаратных платформах.

ШИФРОВАНИЕ, ХЭШ-ФУНКЦИЯ, КРИПТОПРИМИТИВ, АЛГОРИТМ, КРИПТОАНАЛИЗ.

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ, УСЛОВНЫХ ОБОЗНАЧЕНИЙ, СИМВОЛОВ, ЕДИНИЦ И ТЕРМИНОВ

БСШ	–	блочный симметричный шифр
ДК	–	дифференциальный криптоанализ
КЗИ	–	криптографическая защита информации
ЛК	–	линейный криптоанализ
NESSIE	–	New European Schemes for Signature, Integrity and Encryption, международный конкурс криптографических примитивов для применения в Европейском Союзе
SHA		алгоритм криптографического хэширования
NIST		Институт Стандартов и Технологий США
FPGA		полупроводниковое устройство, конфигурируется производителем или разработчиком после изготовления
ASIC		интегральная схема для специфического применения
OpenSSL		криптографический пакет с открытым исходным кодом для работы с SSL
SSL		криптографический протокол, который обеспечивает безопасность связи через Интернет
HMAC		хэш-код аутентификации сообщений
ASIC		интегральная схема, специализированная для решения конкретной задачи
FPGA		полупроводниковое устройство

ВВЕДЕНИЕ

Целью НИР является анализ, теоретическое обоснование унификации, стандартизации и развития методов и механизмов криптографической защиты информации, определение условий их практической реализации с учетом признанных международных и национальных требований относительно гарантий безопасности, усовершенствование и разработка методов, криптографических примитивов блочного симметричного шифрования и хэширования, а также методов оценки криптографической стойкости, сложности криптографических преобразований.

Объектом исследований являются криптографические методы и механизмы защиты данных в перспективных стандартах КЗИ.

Предметом исследований являются методы и механизмы симметричных криптопреобразований и алгоритмы (функции) хэширования.

Методы исследований опираются на использование положений теории построения блочных симметричных шифров и функции хэширования, теории вероятности, методов комбинаторного и структурного анализа.

Актуальность. Теоретическое и практическое обоснование криптографических методов и механизмов симметричных криптопреобразований, функций хэширования, реализация криптографических преобразований и криптографических протоколов на основе применения аппаратно - программных средств КЗИ позволило достичь обеспечения уровней гарантий согласно принятой политике безопасности, совершенствование и разработки перспективных стандартов КЗИ.

Поставленная цель достигается решением следующих научно-технических задач:

– сравнительный анализ и совершенствование существующих методов и алгоритмов (функций) хэширования по критериям защищенности от

существующих угроз и повышения быстродействия, определение проблемных вопросов и прогнозирования путей совершенствования и развития;

– усовершенствование и программное тестирование по базовым статистическим тестам алгоритмов (функций) хэширования.

Хэширование является одним из основных элементов защиты информации. Хэш-функции применяются в криптографических преобразованиях требуемого уровня стойкости для решения задач информационной защиты данных. Хэширование используется для построения ассоциативных массивов, поиска дубликатов в сериях наборов данных, построения уникальных идентификаторов для наборов данных, контрольное суммирование с целью обнаружения случайных или намеренных ошибок при хранении, передаче данных, для хранения паролей в системах защиты, при выработке электронной подписи, кодов аутентификации сообщений и как составные элементы многих других протоколов.

Существует множество алгоритмов хэширования с различными свойствами: разрядность, вычислительная сложность, криптографическая стойкость. Выбор той или иной хэш-функции определяется спецификой решаемой задачи.

Криптографическая хэш-функция-удовлетворяющая ряду требований специфичных для криптографических приложений: необратимость, стойкость к коллизиям первого рода, стойкость к коллизиям второго рода. На основе функций хэширования строятся следующие криптографические примитивы: блочные шифры, поточные шифры, генераторы псевдослучайных чисел, алгоритмы электронной цифровой подписи.

В связи с актуальностью и необходимостью построения хэш-функций с удовлетворением современных требований в 2007-2012г.г. Национальный Институт Стандартов и Технологий США провел конкурс на построение нового алгоритма хэширования SHA-3. Необходимость разработки нового

алгоритма вызвана появлением в научной литературе сведений о потенциальных атаках на SHA-1. Национальный Институт Стандартов и Технологий США в 2007 году объявил конкурс на новый безопасный стандарт хэширования SHA-3. Команды криптографов со всех стран предлагали свои варианты и совместно анализировали дизайн разработок, выявляя уязвимости.

На данном конкурсе были представлены 64 кандидата, из которых только 5 (BLAKE, Grøstl, JH, KECCAK, Skein) оказались наиболее перспективными. Выбранные пять финалистов включают в себя новые идеи проектирования, которые возникли в последние годы, такие как конструкции HAIFA и Sponge.

В данном отчете приведены результаты классификации алгоритмов хэширования, оценки важнейших характеристик, а также способы построения функций хэширования с наименьшими затратами на реализацию. Результаты работы могут быть использованы для дальнейших исследований.

В разделе 1 рассмотрены основные свойства и критерии отбора предъявляемые криптографическим хэш-функциям. Проект SHA-3 –конкурс по выбору нового стандарта хэширования в США, который проводился Институтом Стандартов и Технологий США (NIST – National Institute of Standards and Technologies). Данный проект подразумевает выбор наилучшего алгоритма хэширования по критериям криптостойкости и быстродействия.

В разделе 2 рассмотрены наиболее перспективные алгоритмы хэширования. Алгоритм BLAKE использует конструкцию HAIFA, построен по усиленной схеме Меркля-Дамгаарда. Алгоритм Grøstl построен по схеме Меркля-Дамгаарда с финальным преобразованием. Алгоритм JH основан на схеме Меркля-Дамгаарда. Keccak построен по принципу криптографической Sponge. Алгоритм Skein основан на блочном шифре Threefish.

Проводиться анализ универсальных алгоритмов хэширования Keccak, Skein. Выделяются отличительные характеристики обуславливаемые

оригинальностью конструкции и высокой степенью адаптации к различным платформам. Рассматриваются перспективы реализации на различных типах процессоров.

Реализованы четыре базовые статистические теста для 5 финальных алгоритмов SHA -3, а так же алгоритмов MD 5 и SHA-1 и SHA-2. Эти тесты основаны на различных статистических свойствах, присущих только случайным последовательностям. Представлены таблицы сравнения.

Приведены принципиальные схемы ядер алгоритмов хэширования. Описаны основные схемы аппаратной реализации. Проведена оценка вычислительных затрат данных хэш-алгоритмов с приведением описания методики оценки

Проводится оценка затрат на реализацию алгоритмов хэширования, портированных на 8-мибитные процессорах смарт-карт. Смарт-карты содержат микропроцессор и операционную систему, контролирующую устройство и доступ к объектам в его памяти. Кроме того, они обладают возможностью проводить криптографические вычисления. Приведена систематизация хэш-алгоритмов относительно приведенных критериев.

1 РАЗРАБОТКА И ТЕОРЕТИЧЕСКОЕ ОБОСНОВАНИЕ ДОКАЗОВО СТОЙКОЙ АУТЕНТИФИКАЦИИ НА ОСНОВЕ УНИВЕРСАЛЬНОГО ХЕШИРОВАНИЯ

Результаты международных проектов NESSIE (2000÷2003гг.) [1-20] и NIST SHA-3 Competition (2007÷2012гг.) [21-27] определяют требования к построению коллизионно-стойких функций хеширования и ключевых функций хеширования для вычисления кодов аутентификации сообщений, аутентификации и установления ключей в криптографических протоколах, прежде всего в протоколах электронной цифровой подписи [66]. Критериями оценивания кандидатов на стандарт SHA-3 являются стойкость к атакам на хеш функцию, сложность и скорость вычисления, характеристики и реализация алгоритма. Гарантированная стойкость к атакам реализуется в теории доказуемо стойкой аутентификации. Актуальность диссертационной работы обуславливается необходимостью разработки национального стандарта хеширования. Достижение этой цели возможно на основе методов доказуемо стойкой аутентификации удовлетворяющих требованиям по защищенности и скорости вычисления.

Задачей раздела является анализ методов построения доказуемо стойкой аутентификации в теории универсального хеширования. Универсальное хеширование предложено Картером и Вегманом [29,30] и определяется семействами хеш функций с заданными комбинаторными свойствами. Основные результаты исследований представлены в работах Стинсона, Биербрауэра и др. [32-34,36]. В 1.1 представлен анализ методов MAC кодирования, сравнительные результаты по защищенности и скорости наилучших и перспективных алгоритмов по исследованиям в работах [20-23,37,40-46]. В 1.2 - оценки секретности кодов аутентификации сообщений по работе [47]. Коллизионные свойства MAC кодов универсального хеширования и строго универсального хеширования рассмотрены в работах

[29,30,48-54] и представлены в 1.3 и 1.4. В 1.4, 1.5 изложены методы построения строго универсальных хеш функций и граничные оценки универсального хеширования по результатам работ [54-56]. В 1.6 сформулирована научная проблема диссертационных исследований и частные научные задачи.

1.1 Анализ методов построения MAC кодов

Определение 1.1 [37]. MAC код есть функция отображения $h: K \times D \to R$, где пространство ключей $K=\{0,1\}^k$, пространство сообщений $D=\{0,1\}^*$ и пространство MAC значений $R=\{0,1\}^n$ для $k, n \geq 1$. Для заданных значений ключа $k \in K$ и сообщения $X \in D$, функция производит MAC вычисление $Y \in R$.

MAC коды это итеративные конструкции, которые используют функцию сжатия f, предварительное разбиение данных X на подблоки X_i и связку по обратному входу промежуточных результатов вычислений хеш значений [68]. Секретный ключ может использоваться в векторе инициализации, в функции сжатия, и выходном преобразовании. Основными характеристиками MAC алгоритмов в соответствии с рекомендациями проекта NESSIE являются уровень защищенности MAC кодов от общих атак и быстродействие MAC алгоритмов [20].

Можно выделить три основных подхода к построению кодов аутентификации сообщений [20].

1. коды аутентификации сообщений, построенные с применением блочных шифров;

2. коды аутентификации сообщений, построенные на основе безключевых хэш-функций;

3. коды аутентификации сообщений, построенные с использованием семейства универсальных хэш-функций.

Основные результаты по MAC алгоритмам.

MAC коды на основе блочного шифра определяются стандартом ISO/IEC 9797-1 и используют шифрование в режиме CBC сцепления шифр текстов [63]. Безопасность CBC MAC конструкций основывается на

криптографической стойкости блочного шифра и оценки секретности получены для статистической модели блочного шифра как псевдослучайной функции [38].

Коды аутентификации сообщений на основе безключевых хеш-функций (HMAC) определяются стандартом ISO/IEC 9797-2 и включают значение секретного ключа в вычисление хеш результата [10,64]. Безопасность HMAC конструкции основывается на предположении, что основная хеш-функция является коллизионно-стойкой при секретном начальном значении, ключевая функция сжатия с помощью начального значения является защищенным MAC алгоритмом (для сообщений размером в один блок) и функция сжатия есть слабая псевдослучайная функция [39,65]. Коды подлинности сообщений на основе безключевых хеш-функций обычно более быстрые, чем коды подлинности сообщений, на основе блочного шифра.

Коды подлинности сообщений на основе универсального хеширования используют ключевое хеширование с помощью семейства хеш-функции. Безопасность универсальной аутентификации определяется коллизионной стойкостью MAC значений, которые вычисляются на семействе хеш-функции. Комбинаторные свойства универсального хеш семейства позволяют получить точные оценки секретности для MAC кодирования. Отличием универсальной аутентификации от CBC MAC и HMAC аутентификации является доказуемая безопасность.

Основными практическими алгоритмами кодов подлинности сообщений являются:

- **Two-Track-MAC**: K.U.Leuven, Бельгия и debis AG, Германия;
- **UMAC**: разработка корпорации Intel, США, Университета из штата Невада в Рено, США, Научно-исследовательской лаборатории IBM, США, Technion, Израиля и Университета из Калифорнии в Девисе, США;
- **CBC-MAC (ISO/IEC 9797-1);**
- **HMAC (ISO/IEC 9797-2).**

Практическими алгоритмами хеширования в HMAC (ISO/IEC 9797-2) алгоритме являются: ГОСТ 34.311-95, HAVAL, SHA-1, RIPEMD -160, MD5, ГОСТ 28147-89 режим 4, Whirpool [65], SHA-2 . В конструкции CBC-MAC (ISO/IEC 9797-1) блочные шифры Rijndael, DES. В таблице 1.1 представлены основные результаты по параметрам и оценке быстродействия основных алгоритмов аутентификации [20,65]. Скорость вычислений определяется количеством циклов процессора, затрачиваемых на один байт обрабатываемого сообщения.

Таблица 1.1

Быстродействие MAC алгоритмов

Алгоритм	Длина MAC кода (бит)	Длина ключа (бит)	Тип ПЭВМ				
			Pentium 2	PIII/Linux	Pentium 4	Xeon	AMD
TTmac	160	160	21	21	40	37	21
UMAC-16	64	128	6.1	6.0	6.2	6.1	6.2
UMAC-32	64	128	2.5	2.9	6.7	6.6	1.9
HMAC-Whirlpool	512	512	86	72	98	103	100
HMAC-MD4	128	512	4.7	4.7	6.4	6.4	4.7
HMAC-MD5	128	512	7.2	7.3	9.4	9.4	7.4
HMAC-RIPE-MD	160	512	23	18	27	26	21
HMAC-SHA-0	160	512	16	15	23	23	13
HMAC-SHA-1	160	512	16	15	25	24	12
HMAC-SHA-2	256 384 512	512	40 84 84	39 84 84	40 124 124	39 132 132	33 72 72

HMAC-Tiger	192	512	24	21	28	26	20
CBCMAC-Rijndael	128	128	24	26	26	27	31
CBCMAC-DES	64	56	62	61	72	69	54
CBCMAC-Shacal	512	160	31	31	67	74	29

TTMAC (Two-Track-MAC) алгоритм основан на хеш-функции RIPEMD-160 с небольшими модификациями [40,41]. Алгоритм работает на блоках 512 бит разделенных на слова по 32 бит, использует секретный ключ 160 бит и производит выход до 160 бит. Большой размер внутреннего состояния (320 бит) в Two-Track-MAC дает алгоритму высокий уровень защиты от атак, основанных на внутренних коллизиях.

Сложность основных атак на этот примитив следующая:

- приблизительно 2^{159} вычислений MAC кода и 160/m известных пар текст - MAC необходимы для исчерпывающего поиска ключа, где m - длина MAC результата (значения для m поддерживается алгоритмом между 32 и 160 битами).
- угадывание значения MAC кода имеет вероятность успеха 2^{-m}.
- атаки, основанные на внутренних коллизиях, требуют приблизительно 2^{160} известных пар текст – MAC код и приблизительно 2^{320-m} выбранных текстов.

Алгоритм имеет самый высокий уровень защиты для MAC примитивов, определенные преимущества в быстродействии, особенно в случае коротких сообщений, и оптимальную длину ключевых данных. Вместе с тем, TTMAC имеет низкую скорость, что делает проблематичным применение для приложений, где требуется хешировать данные больших объёмов.

UMAC алгоритм является программно ориентированным, на основе применения композиционной схемы с многократным универсальным хешированием и криптографическим вычислением тега аутентификации.

UMAC алгоритм известный по первоначальным модификациям UMAC(1999) и UMAC(2000) [42-44].

Алгоритм UMAC и спецификации были разработаны, так чтобы обеспечить параллельные вычисления в SIMD архитектуре. SIMD архитектура обеспечивается регистрами, которые, в некоторых инструкциях, могут обращаться с малоразмерными словами как векторами. Одна из самых быстрых реализаций UMAC использует MMX инструкции Pentium, которые обращаются с 64 битовым регистром как с четырехмерным вектором 16 битовых слов.

В зависимости от установки начальных параметров алгоритма UMAC изучены пять схем: UMAC-STD-30, UMAC-STD-60, UMAC-MMX-15, UMAC-MMX-30, и UMAC-MMX-60.

Результаты испытаний схем UMAC (1999) с оценкой скорости вычислений представлены в таблице 1.2 [43].

Таблица 1.2

Пиковые скорости вычисления UMAC, измеренные в Гбит/сек (циклы/байт).

Алгоритм	Pentium II	PowerPC	Альфа
UMAC-STD-60	1.49 (1.93)	1.81 (1.58)	1.03 (2.78)
UMAC-STD-30	2.79 (1.03)	2.28 (1.26)	1.79 (1.60)
UMAC-MMX-60	2.94 (0.98)	4.21 (0.66)	0.287 (10.0)
UMAC-MMX-30	5.66 (0.51)	7.20 (0.39)	0.571 (5.02)
UMAC-MMX-15	8.47 (0.33)	10.5 (0.27)	0.981 (2.85)
CBC-MAC-RC6	0.162 (17.7)	0.210 (13.7)	0.068 (42.5)
HMAC-SHA1	0.227 (12.6)	0.228 (12.6)	0.117 (24.5)

Максимальный коэффициент сжатия достигается на сообщениях 4 Кбт. UMAC выполняется лучше всего на длинных сообщениях, потому что хэш-функция является более эффективной из-за сокращения количества вычислений приходящихся на псевдослучайную функцию PRF. В качестве

PRF функции применяется одна из криптографических хеш-функции в режиме CBC-MAC или HMAC. Некоторый выигрыш в скорости появляется уже при длинах сообщения в пару сотен байт.

Повышение скорости вычислений особенно для сообщений малой длины было достигнуто в UMAC (2000). В UMAC (2000) предложены две схемы: UMAC32 (без SIMD параллелизма) и UMAC16 (с SIMD параллелизмом), что позволило достигнуть производительности в три раза быстрее, чем первоначальных версиях UMAC-STD и UMAC-MMX. В 2005 году была предложена модифицированная версия UMAC для 64 битных архитектур - UHASH8 [46], обеспечивающая достижение пиковой производительности 0,5 циклов процессора на байт на Athlon 64. Сравнительные результаты производительности представлены в таблице 1.3 [43,45,46].

Таблица 1.3

Производительность алгоритмов UMAC для различных длин данных (циклы/байт).

Алгоритм	43 Бт.	256 Бт.	1500 Бт.	256 КБт.
UMAC32	16.3	3.8	2.1	1.9
UMAC-STD	52.9	12.3	3.8	1.9
UMAC16	14.0	2.7	1.2	1.0
UMAC-MMX	35.9	4.5	1.7	1.0
UHASH8	4.8	1.5	1.2	1.0

UMAC использует три семейства хеш-функций: NH, RPHash и IPHASH. В реализации UMAC32 каскад NH хеширует блоки по 2КБт, производя хеш-результат в 32 бита, что соответствует коэффициенту сжатия 512. Вероятность коллизии не превышает 2^{-15}. Результат передается к RP уровню хеширования, который вычисляет выходную строку длины 128 бит. RP семейство использует три простых поля с 32 битным, с 64 битным и с 128 битным простыми модулями, соответственно. Длина сообщения ограничена

максимальным значением 2^{64} бит, и доказано, что этот уровень добавляет около 2^{-19} к вероятности коллизии. Если аутентифицируемое сообщение короткое, RP слой пропускается для оптимизации скорости вычисления. IP уровень сворачивает входную последовательность с 128 битами к выходному – с 16 битами, поддерживая вероятность коллизии на уровне 2^{-15}. Конструкция с тремя уровнями повторяется неоднократно, с независимыми ключами, увеличивая длину аутентификационного тега и уменьшая шанс подделки MAC кода. Заданное по умолчанию число - четыре раза, и конкатенация 16 битовых слов вычисляет MAC код с 64 битами с вероятностью подделки 2^{-60}. Главное отличие в UMAC32 состоит в том, что используются слова с 32 битами и вычисления повторяются по схеме с тремя слоями только дважды (значение по умолчанию).

Отметим ряд недостатков в данном алгоритме. NH хеширование на начальном шаге UMAC алгоритма предполагает генерирование 1024 4-х байтовых подключей. В принципе это является типичным для многих современных шифров и для длительных сеансов не оказывает существенного влияния на производительность алгоритма. Однако учитывая кратковременность многочисленных сеансов в современных компьютерных сетях, это является проблемой, так как приводит к снижению производительности алгоритма. Алгоритм UHASH8 снижает зависимость от ключевых данных до 160 байт на один сеанс передачи данных [46].

Второй уровень RPHash хеширования использует для вычислений в простом поле с 32 битным, с 64 битным и с 128 битным простыми модулями. Это определяется необходимостью перехода к хешированию данных всё возрастающей длины. Согласно UMAC алгоритма 32-х разрядные вычисления поддерживают хеширования 2^{17} битовых данных, что является недостаточным для передачи файловых и мультимедийных приложений.

Основные результаты конкурса Национального института стандартизации США (NIST) “NIST SHA-3 Competition”.

Алгоритм победитель конкурса должен поддерживать размер выходного блока 224, 256, 384 и 512 бит, минимальные отличия в реализации, реализацию на множестве операционных платформ, включая 64 битовые процессоры и 8 битовые процессоры, и смарт карты [21,23]. Критериями оценивания кандидатов на стандарт SHA-3 являются:

- стойкость к атакам;
- сложность и скорость вычисления;
- характеристики и реализация алгоритма.

Основные требования представленные Национальным институтом стандартизации США к алгоритмам конкурсантам предусматривают построение классов хеш функций гарантированной стойкости, стойких к потенциальным атакам на SHA-2, и сохранение или увеличение скорости хеширования по сравнению с SHA-2. Из 56 алгоритмов заявленных на начало конкурса в финальную стадию прошли 5 алгоритмов: BLAKE (Швейцария - Jean-Philippe Aumasson), Grøstl (Дания, Австрия - Lars Ramkilde Knudsen), JH (Сінгапур - Hongjun Wu), Keccak (Бельгия - Joan Daemen), Skein (США - Брюс Шнайер). Данный выбор алгоритмов определяется различием их архитектурных, структурных моделей, методов вычислений, при условии обеспечения высокой надежности и защищенности. Особенности реализации алгоритмов представлены в таблице 1.4.

Таблица 1.4

Конструктивные особенности алгоритмов хеширования конкурса SHA-3

Алгоритм	Описание алгоритма	Особенности реализации
BLAKE	Алгоритм сжатия использует ключевую подстановку конструкции Davies-Meyer. Ключевая подстановка основана на внутренней организации поточного шифра ChaCha. Свойство нелинейности определяется наложением модульного дополнения и операции XOR.	Применение ключевой подстановки
Grost l	Ширококанальная конструкция хеш алгоритма Merkle-Damgard. Функция сжатия использует два S-бокса AES в качестве блока подстановки.	Применение в функции сжатия S-боксов AES

	Нелинейность алгоритма определяется S-боксом AES.	
JH	Используется конструкция близкая по структуре к “sponge” для встраивания хеш алгоритма в блок подстановки. Блок подстановки есть комбинация двох 4 битовых S-боксов з рядом линейных операций смешивания и разрядных подстановок. Нелинейность определяется S-боксами.	Применение в функции сжатия S-боксов
Keccak	Используется конструкция “sponge” и блок подстановки. Подстановка может быть реализована на 5 битовых S-блоках или на комбинации линейной и нелинейной операции смешивания.	Применение подстановки “sponge”
Skein	Алгоритм хеширования использует функцию сжатия Matyas-Meyer-Oseas с блочным шифром “Threefish”. Блочный шифр “Threefish” использует большое число простых раундов в которых применяется три 64 битовых дополнения и поразрядная операция XOR. Нелинейность хеш алгоритма определяется наложением модульного дополнения и операцией неэквивалентности.	Применение БСШ “Threefish” в режиме CBC

Основные типы вычислений алгоритмов кандидатов на стандарт SHA-3 представлены в таблице 1.5.

Таблица 1.5

Типы вычислений алгоритмов кандидатов на стандарт SHA-3

Алгоритм	S-box	Умножения в поле (GF MUL)	Умножения (MUL)	Много-операндные сложения (mADD)	Сложения /вычитания (ADD/SUB)	Бинарные логические функции (Boolean)
BLAKE				mADD3	ADD	XOR
Groestl	AES 8x8	x02..x07				XOR
JH	Serpent 4x4	x2, x5				XOR
Keccak						NOT:AN

						D: XOR
Skein					ADD	XOR

mADDn – определяет многооперандное сложение, где *n* – количество операндов.

Быстродействие алгоритмов кандидатов на стандарт SHA-3 представлено в таблице 1.6. Результаты оценки скорости вычислений получены на платформе процессора Intel® Core™2 Quad Processor Q6600 (8M Cache, 2.40 GHz, 1066 MHz FSB) [23].

Таблица 1.6

Оценки скорости хеширования алгоритмов кандидатов на стандарт SHA-3

Алгоритм	Выход 256 бит			Выход 512 бит		
	Мбайт/с	Циклов /байт	Норм. скорость	Мбайт/с	Циклов /байт	Норм. скорость
BLAKE	199.58	12.03	2.273	44.30	54.18	0.767
Groestl	29.31	81.88	0.334	21.17	113.37	0.367
JH	19.85	120.91	0.226	19.97	120.18	0.346
Keccak	41.05	58.47	0.467	22.60	106.19	0.391
Skein	63.91	37.55	0.728	61.85	38.80	1.071

Нормализованная скорость - скорость относительно алгоритма SHA-2.

Представленные в проекте оценки стойкости хеш функций определены на основе методов дифференциального, линейного криптоанализа, проверки наличия криптографически слабых вычислений и не являются доказуемо стойкими.

Выводы.

1. Результаты международных проектов NESSIE и NIST SHA-3 Competition подтверждают актуальность решения задачи разработки хеш функций гарантированной стойкости для построения национального стандарта хеширования.

2. Практические алгоритмы вычислений должны поддерживать эффективные вычисления на различных типах платформ процессоров 8, 32 и 64 разрядности, а перспективе и 128 разрядов, возможность реализации алгоритмов с параллельными вычислениями и наборами инструкций с высоким быстродействием, структурно простыми и прозрачными для анализа криптографической стойкости.

3. Гарантированная стойкость к атакам реализуется в теории доказуемо стойкой аутентификации. Доказуемо стойкая аутентификация строится на основе универсального хеширования. Практическим алгоритмом является UMAC алгоритм. Алгоритмы универсального хеширования определяются строгими теоретическими оценками безопасности и имеют самую высокую скорость вычислений.

1.2 Оценки секретности кодов аутентификации сообщений

Коды аутентификации сообщений в представлении Картера-Вегмана определяются семейством хеш-функций [29,30].

Определение 1.2 [29]. $(N;n,m)$ хеш семейство есть множество из N функций H такое, что

$$h: A \to B, \tag{1.1}$$

где $h \in H$, $|A| = n$ и $|B| = m$, $n \geq m$.

Массив значений MAC кодов состоит из N строк, n столбцов, элементы принимают одно из m значений. Каждая функция $f \in H$ определяется значением используемого ключа, связывается со строкой и определяет правило отображения элементов множества X (номеров столбцов массива) в элементы Y (собственные значения элементов массива).

Замечание 1.1. Представление MAC кодов в виде массива значений позволяет рассматривать статистические распределения аутентификаторов на пространстве $A \times B$, что в свою очередь связано с коллизионными характеристиками.

Определение MAC кода с учетом секретности имеет следующий вид.

Определение 1.3 [37]. MAC код $f: A \to B$ является $(t;\varepsilon;q)$ - секретным, если, при случайно взятом ключе k, противник не может подделывать новое сообщение за время t с вероятностью больше чем ε, если ему представлены значения q MAC кодов других сообщений по его выбору.

Постановка задачи. Пусть состояние источника сообщений определяется значением x. Секретный ключ является одной из функций $f \in H$. В передаваемое сообщение включается аутентификатор $y = f(x)$. Нарушитель может наблюдать передаваемое сообщение x и его аутентификатор $y = f(x)$. Определить оценки секретности кодов аутентификации сообщений.

Решение задачи представлено в [47].

Для подмены сообщений, нарушитель должен сформировать x' и соответствующий аутентификатор $y' = f(x')$. Это может быть сделано двумя способами: путем имитации и подмены [67].

При имитации нарушитель производит аутентификатор $y = f(x)$, основываясь на априорных вероятностях распределений MAC значений и ключевых данных. Вероятность имитации будет определяться максимальной вероятностью того, что произведенный аутентификатор $y = f(x)$ является истинным

$$P_{им} = P(y = f(x) \text{-истинно}),\ (x,y) \in A \times B,\ f \in H. \qquad (1.2)$$

Утверждение 1.1 [47]. При равновероятном выборе ключа вероятность имитации, которую обозначим через вероятность имитации по ключу, определяется выражением

$$P_{им.кл.} = \left| f \in H : y = f(x) \right| / |H|,\ (x,y) \in A \times B, \qquad (1.3)$$

где $\left| f \in H : y = f(x) \right|$ - число хеш- функций *f*, порождающих для сообщения *x* значение MAC кода *y*.

Действительно, при равновероятном выборе ключа, что эквивалентно выбору $f \in H$, необходимо учитывать распределение MAC значений *y* для

данного сообщения x по ключевому пространству. Размер ключевого пространства равняется $|H|$. Имитация наступит, если для $f \in H : y = f(x)$. Мощность множества благоприятных исходов равна $\left| f \in H : y = f(x) \right|$ и отношение к $|H|$ даёт искомый результат.

Замечание 1.2.

1. Нижняя граница вероятности имитации по ключу равна

$$P_{им.кл.} \geq 1/|H| \tag{1.4}$$

2. Если все записи в столбцах массива MAC кодов встречаются одинаковое число раз, получим

$$P_{им.кл.} \geq 1/|B|$$

3. Верхняя граница вероятности имитации MAC кода по ключу определяется максимальным значением $P_{им.кл.}$ по всему пространству сообщений и значение вероятности ограничивается следующим соотношением

$$P_{им.кл.} \leq \max_{x \in A} \left| f \in H : y = f(x) \right| / |H|, \ (x, y) \in A \times B. \tag{1.5}$$

Если не учитывать распределение MAC значений y для данного сообщения x по ключевому пространству, тогда вероятность имитации обозначим как вероятность имитации по MAC значению $P_{им.MAC.}$. Имитация через навязывание MAC значения определяется тем, что выбирается одно значение MAC из множества предполагаемых кодов.

Утверждение 1.2 [47]. Вероятность успеха имитации через навязывание MAC кода определяется выражением

$$P_{им.MAC.} = 1/\left| y \in B : y = f(x) \right|, \ (x, y) \in A \times B, f \in H, \tag{1.6}$$

где $\left| y \in B : y = f(x) \right|$ есть мощность множества возможных MAC значений для сообщения x.

Действительно, благоприятный исход имитации возможен только для одного значения $y \in B$. Для каждого сообщения x существует пространство MAC кодов мощности $\left| \ y \in B : y = f(x) \ \right|$, что приводит к выражению (1.6).

Замечание 1.3.

1. Если MAC значения для сообщения *x* принимают полное множество значений $|B|$, имеем $P_{им.MAC.} = 1/|B|$.

2. Для вероятности имитации по MAC коду справедлива граница снизу

$$P_{им.MAC.} \geq 1/|B|. \qquad (1.7)$$

3. Если известно статистическое распределение MAC значений для сообщения, тогда оценка для вероятности имитации по MAC значению сводится к оценке вероятности имитации по ключу.

4. Верхняя граница для вероятности имитации по MAC значению определяется максимальным значением $P_{им.MAC.}$ по всему пространству сообщений

$$P_{им.MAC.} \leq \max_{x \in A} 1/\left| \ y \in B : y = f(x) \ \right| , \ y \in B, \ f \in H . \qquad (1.8)$$

Атака подмены основана на том, что нарушитель наблюдает *(x,y)* и изменяет его на *(x',y'),* где *x≠x'*. Вероятность подмены определяется условной вероятностью

$$P_{под} = P(f(x') = y' - истинно \,|\, f(x) = y), \ (x,y),(x',y') \in A \times B, \ x \neq x', \ f \in H . \qquad (1.9)$$

Утверждение 1.3 [47]. Пусть нарушитель наблюдает *(x,y)* и изменяет его на *(x',y'), x≠x'*. Выражение для вероятности подмены имеет следующий вид

$$P_{под} = \left| \ f \in H : y = f(x), y' = f(x') \ \right| / \left| \ f \in H : y = f(x) \ \right|, \ (x,y),(x',y') \in A \times B, \ x \neq x', \ f \in H , \qquad (1.10)$$

где $\left| \ f \in H : y = f(x) \ \right|$- мощность множества ключей, для которых $y = f(x)$, $\left| \ f \in H : y = f(x), y' = f(x') \ \right|$ - мощность множества ключей для которых $y = f(x)$ и $y' = f(x')$.

Действительно, наблюдая (x, y), нарушитель может определить пространство ключей $f \in H$ для которых $y = f(x)$ мощности $\left| \ f \in H : y = f(x) \ \right|$. Благоприятный исход подмены состоит в том, чтобы угадать ключ на котором $y' = f(x')$. Пространство благоприятных ключей равно $\left| \ f \in H : y = f(x), y' = f(x') \ \right|$, что приводит к искомому результату.

Замечание 1.4.

1. Точные оценки вероятности подмены возможны в случае известных статистик совместных распределений МАС кодов по ключам для исходных и навязываемых сообщений.

2. Если статистика совместных распределений МАС кодов по ключам для исходных и навязываемых сообщений неизвестна, вместо оценки вероятности подмены рассматривают вероятность коллизии для МАС кода, которая определяется соотношением

$$P_{кол} = \max_{x,x' \in A} \left| \ f \in H : y = f(x), y = f(x') \ \right| / |H|, \ (x, y), (x', y') \in A \times B, \ x \neq x', \ f \in H . \quad (1.11)$$

3. Возможны два случая, когда подмена сообщения x на x', при $x \neq x'$ осуществляется с одним и тем же МАС кодом $y = y'$ (подмена первого рода - $P_{под1}$) и с разными МАС $y \neq y'$ (подмена второго рода- $P_{под2}$).

Основные результаты по оценкам секретности кодов аутентификации сообщений представлены в табл. 1.7, 1.8.

Таблица 1.7

Атаки на аутентификацию	Оценки секретности кодов аутентификации сообщений
атака имитации	$P_{им}$=$P($ $y = f(x)$ -истинно$)$, $(x,y) \in A \times B$, $f \in H$. $P_{им.кл.} = \left| \ f \in H : y = f(x) \ \right| / |H|$

	$P_{им.MAC.} = 1/\left\| \; y \in B : y = f(x) \; \right\|$
атака подмены	$P_{под} = P(f(x') = y' - истинно \,|\, f(x) = y)$, $(x,y), (x',y') \in A \times B$, $x \neq x'$, $f \in H$. $P_{под} = \left\| \; f \in H : y = f(x), y' = f(x') \; \right\| / \left\| \; f \in H : y = f(x) \; \right\|$

Таблица 1.8

Атаки имитации	Оценки вероятности атаки имитации	
	Граница снизу	Граница сверху
атака имитации по ключу	$P_{им.кл.} \geq 1/\|H\|, \|H\| < \|B\|$ $P_{им.кл.} \geq 1/\|B\|, \|H\| \geq \|B\|$	$P_{им.кл.} \leq \max\limits_{x \in A} \left\| \; f \in H : y = f(x) \; \right\| / \|H\|$
атака имитации по MAC коду	$P_{им.MAC.} \geq 1/\|B\|$	$P_{им.MAC.} \leq \max\limits_{x \in A} \; 1/\left\| \; y \in B : y = f(x) \; \right\|$

Выводы.

1. Для точного вычисления имитационной и коллизионной стойкости MAC кодов необходимо использовать статистику совместных распределений MAC кодов по ключам для исходных и навязываемых сообщений. Для практических MAC кодов знание такой статистики выглядит проблематичным из-за чрезвычайно больших размеров массива аутентификаторов.

2. Нижние границы для вероятностей имитации и подмены определяются мощностью пространства ключей и MAC кодов, не учитывают статистические свойства массивов аутентификаторов. Требования к вероятности подмены и имитации определяют минимальные требования к размеру ключевого пространства и пространства MAC значений.

3. Верхние границы для вероятностей имитации и подмены связаны с комбинаторными свойствами MAC массивов и определяют значения вероятности коллизий на пространстве $A \times B$ для наихудшего случая выбора ключей и сообщений.

1.3 Коллизионные свойства MAC кодов универсального хеширования

MAC коды универсального хеширования определяются массивами с известными статистическими и комбинаторными свойствами, что позволяет, как правило, получить точные коллизионные границы. Основные положения универсального хеширование приведены в работах [29,30,48,49], уточнения и дополнения в [50,51].

Определение 1.4 [29]. $(N;n,m)$ хеш семейство является $\varepsilon-$ универсальным, если для любых двух различных элементов $x_1, x_2 \in A$, существует самое большее εN функций $h \in H$ таких, что $h(x_1) = h(x_2)$. Аббревиатура $\varepsilon - U$ используется для обозначения $\varepsilon-$ универсальных хеш функций.

Утверждение 1.4 [50]. Пусть h выбирается случайно из заданного $\varepsilon - U(N;n,m)$ хеш семейства, тогда вероятность коллизии хеш значений для двух разных входных сообщений $x_1, x_2 \in A$ не превышает ε

$$P_{кол} = \Pr_{h \in H}\left[h(x_1) = h(x_2)\right] \leq \varepsilon.$$

Действительно, по соотношению (1.11) мощность множества пространства ключей равно $|H| = N$ и мощность пространства благоприятных ключей самое большее $\left| f \in H : y = f(x), y' = f(x') \right| = \varepsilon N$, что дает оценку сверху для $P_{кол} \leq \varepsilon$. ◊

Замечание 1.5.

1. Первоначальное определение универсальных хеш функций Картера и Вегмана было предложено для $\varepsilon = 1/m$ [29].

2. Вероятность коллизии для универсальных хеш функций Картера и Вегмана является наименьшей и определяется мощностью пространства хеш значений $P_{кол} = 1/|B|$.

Определение 1.5 [29]. H является $\varepsilon-$ почти универсальным семейством хеш- функций $\varepsilon - AU(N;n,m)$, если $\Pr_{h \in H}\left[h(x_1) = h(x_2)\right] \leq \varepsilon$ для $x_1, x_2 \in A$, $x_1 \neq x_2$, $1/m < \varepsilon \leq 1$.

Замечание 1.6.

1. Для почти универсальных семейств несколько ослабляются требования к вероятности коллизии.

2. Свойство универсальности (почти универсальности) не связанно с распределением MAC значений по ключевому пространству и, следовательно, не определяет вероятностные характеристики имитационной атаки.

3. Универсальное хеширование определяет доказуемо стойкую аутентификацию со счетчиком в представлении Картера – Вегмана [29]

Известны три метода построения универсальных хеш функций. Алгоритмы вычисления хеш функций и свойства хеш классов представлены в табл. 1.9.

Таблица 1.9

Алгоритмы построения универсальных хеш классов и их свойства

Название метода	Алгоритм вычисления	Свойство хеш класса
Метод скалярного произведения [30]	$y=\sum_{i=1}^{k} x_i m_i\ ,\ y,x_i,m_i \in F_q$	$1/q-U(q^k,q^k,q)$
Метод полиномиального хеширования [30]	$y=\sum_{i=1}^{k} m_i x^i\ ,\ y,x,m_i \in F_q$	$k/q-U(q,q^k,q)$
Метод на основе алгебраического кодирования [31,32]	$y=c_i(x)\ ,\ c(x)\in(n,k,d)_q$	$(1-d/n)-U(n,q^k,q)$

Дальнейшим развитием универсальных схем являются строго универсальные.

Определение 1.6 [48]. $(N;n,m)$ хеш семейство является $\varepsilon-$ строго универсальным $\varepsilon-SU(N;n,m)$, если для каждого $x\in A$ и $y\in B$ число функций $h\in H$, таких, что $h(x)=y$ равно N/m, а для любых двух различных элементов $x_1,x_2\in A$, и не обязательно различных $y_1,y_2\in B$ число функций $h\in H$ таких, что $h(x_1)=y_1,\ h(x_2)=y_2$ не превышает

$\nu \le \varepsilon \cdot N / m$. Аббревиатура $\varepsilon - SU$ используется для обозначения $\varepsilon -$ строго универсальных хеш-функций.

Замечание 1.7.

1. Строгая универсальность определена для $\varepsilon = 1/m$.

2. При смягчении требования $\varepsilon > 1/m$ класс функций определяется как почти строго универсальный $\varepsilon - ASU$.

3. Строго (почти строго) универсальное хеширование определяет безусловную аутентификацию и было представлено Стинсоном [48,49].

Утверждение 1.5. Пусть $(N;n,m)$ семейство хеш функций является $\varepsilon -$ строго универсальным $(\varepsilon - SU(N;n,m))$. Тогда $N \ge m^2$, $P_{им.} = 1/m$ и $P_{под} = 1/m$.

Доказательство. По определению строгой универсальности число функций $h \in H$ таких, что $h(x_1) = y_1$, $h(x_2) = y_2$ не превышает $\varepsilon \cdot N / m$. Возьмём нижнюю границу $\nu = 1$ и так как $\varepsilon = 1/m$ имеем $N \ge m^2$. Вероятность имитационной атаки по ключу определяется соотношением (1.3). Прямое вычисление дает $P_{им.кл.} = (N/m)/N = 1/m$, что соответствует нижней границе для вероятности имитации по MAC коду, следовательно $P_{им.} = 1/m$. Вероятность подмены определяется условной вероятностью (1.9). Так как число h для которых $h(x) = y$ равно N/m, а число h для которых $h(x) = y$, $h(x') = y'$ равно $\nu \le \varepsilon \cdot N/m = N/m^2$ и подставляя оценки N/m и N/m^2 в выражение для $P_{под}$ получим $P_{под} = 1/m$. ◊

Коллизионные свойства почти строго универсальных MAC кодов представлены следующими утверждениями.

Утверждение 1.6. Пусть $(N;n,m)$ семейство хеш функций является $\varepsilon -$ строго универсальным $(\varepsilon - SU(N;n,m))$. Тогда $N \ge m^2$, $P_{им.} = 1/m$ и $P_{под} = 1/m$.

Утверждение 1.7. Пусть $\varepsilon - ASU(N;n,m)$ семейство почти строго универсальных хеш функций. При равновероятном выборе хеш функции

вероятность успеха имитационной атаки равна $P_{им}=1/m$ и вероятность подмены $P_{под}\le\varepsilon$

Доказательство аналогично предыдущему.

Характеристики универсальных классов хеш функций приведены в табл. 1.10 и 1.21.

Таблица 1.10

Определения и свойства универсального хеширования.

Класс хеш функций	Свойство хеш класса	Вероятность коллизии
ε–U($N;n,m$)	$\left\| h\in H : h(x)=y, \forall x \right\| = N/m$	$P_{кол}=\Pr_{h\in H}\ h(x_1)=h(x_2)\ =1/m$
ε–AU($N;n,m$)	$\left\| h\in H : h(x)=y, \forall x \right\| = \varepsilon N$	$P_{кол}=\Pr_{h\in H}\ h(x_1)=h(x_2)\ \le\varepsilon$
ε–SU($N;n,m$)	$\left\| h\in H : h(x)=y, \forall x \right\| = N/m$ $\left\| h\in H : h(x_1)=y_1, h(x_2)=y_2, \forall x_1,x_2 \right\| = N/m^2$	$P_{кол}=\Pr_{h\in H}\ h(x_1)=h(x_2)\ =1/m$
ε–ASU($N;n,m$)	$\left\| h\in H : h(x)=y, \forall x \right\| = N/m$ $\left\| h\in H : h(x_1)=y_1, h(x_2)=y_2, \forall x_1,x_2 \right\| = \varepsilon N/m$	$P_{кол}=\Pr_{h\in H}\ h(x_1)=h(x_2)\ \le\varepsilon$

Таблица 1.11

Оценки секретности кодов аутентификации сообщений для универсальных классов хеш функций.

Виды аутентификации	**Универсальные классы хеш функций**	$P_{им.}$	$P_{под}$	N
Доказуемо стойкая аутентификация со счетчиком	**ε–U($N;n,m$)**		**$1/m$**	$N\ge n$
	ε–AU($N;n,m$)		ε	$N\ge m$
Безусловная аутентификация	**ε–SU($N;n,m$)**	**$1/m$**	**$1/m$**	$N\ge m^2$
	ε–ASU($N;n,m$)	**$1/m$**	ε	$N\ge m^2$

Выводы.

1. Универсальное хеширование определяет требование - размер ключевого пространства должен быть не меньше пространства сообщения. Почти универсальное хеширования ослабляет это условие, размер ключа может быть больше или равен размеру хеш кода.

2. Строго универсальный хеш класс характеризуется предельным наименьшим значением вероятности имитации и подмены, но размер ключевого пространства не меньше чем в квадрат раз превышает размер хеш кода.

3. Коллизионные оценки почти строго универсального хеширования связываются с распределениями хешей для пар сообщений по ключевому пространству, что определяет безусловную аутентификацию на массивах аутентификаторов связности два.

1.4 Оценки параметров строго универсального хеширования

Для построения строго универсальных хеш функций используются методы на основе ортогональных массивов, независимых массивов и слабосмещенных массивов.

1.4.1 Строго универсальное хеширование на основе ортогональных массивов

Определение 1.7 [52]. Пусть X, Y являются множествами из k и υ элементов, соответственно, и H есть множество функций осуществляющих отображение $f: X \to Y$. *Ортогональным массивом* $OA_{\lambda}(t,k,\upsilon)$ называется массив элементов $y_i \in Y$, со столбцами соответствующими элементам множества X и строками, определяемыми элементами множества m в котором для любой выборки из t элементов $y_1, y_2, ..., y_t$ из Y существует только λ функций $f \in m$ для которых справедливо $f(x_i) = y_i$, $i = 1,2,...,t$.

Основная конструкция OA массивов определена теоремой 1.1.

Теорема 1.1 [53]. Пусть q простое число, m,n,t - целые числа, $n \geq m$, $2 \leq t \leq q^n$. Зафиксируем сьюрективное F_q - линейное отображение $\phi: F_q^n \to F_q^m$. Для каждого t набора $z, a_1, a_2, ..., a_{t-1}$, где $z \in F_{q^m}$, $a_j \in F_{q^n}$, $i = 1,2,...,t-1$, определим отображение $f = f(z, a_1, a_2, ..., a_{t-1}): F_q^n \to F_q^m$, вида

$$f(x) = \phi\left(\sum_{j=1}^{t-1} a_j x^j\right) + z. \qquad (1.12)$$

Тогда массив, составленный из отображений вида (1.12) является ортогональным с параметрами $OA_{q^{(t-1)(n-m)}}\ t, q^n, q^m$.

Следствие 1.1. Пусть q простое число, $m = n$, $t = 2$. Тогда $OA_{\lambda=1}\ 2, q^m, q^m$ называется простым, каждая строка повторяется только (точно) один раз и определяется линейным отображением $\phi : F_{q^m} \to F_{q^m}$ с функцией $f(x) = \phi(ax) + z$, где $a, z \in F_{q^m}$.

Метод ортогональных массивов можно применить для построения строго универсальных хеш функций. Основной результат представлен в теореме 1.2.

Теорема 1.2 [50]. Пусть q - простое число, a, b, k целые числа, $a > b$. Тогда существует $\frac{k}{q^b} - SU(q^{a+b}, q^{ka}, q^b)$ семейство хеш-функций.

Доказательство. Фиксируем сюрективное F_q - линейное отображение

$$\phi : F_{q^a} \to F_{q^b}.$$

X является множеством полиномов $p(X)$, определенным над F_{q^a} степени $\le k$, без постоянного члена.

Элементы матрицы отображения $X \to Y$ на пересечении (u, υ) строки, $u \in F_{q^a}$, $\upsilon \in F_{q^b}$, и $p(X)$ столбца определим как

$$\phi(p(u)) + \upsilon = y.$$

Очевидно, что число элементов $y \in F_{q^b}$ в столбце равно $N = q^{a+b}$, и они повторяются точно $\frac{N}{m} = q^a$ раз. Зафиксируем $p_1(X)$, $p_2(X)$ и $y_1, y_2 \in F_{q^b}$, тогда

$$\phi(p_1(u)) + \upsilon = y_1;$$

$$\phi(p_2(u)) + \upsilon = y_2.$$

В силу линейности отображения ϕ справедливо

$$\phi((p_1 - p_2)(u)) = y_1 - y_2.$$

Каждое решение u этого уравнения даёт уникальную пару (u,υ). Число решений (u,υ) будет $\le kq^{a-b}$ т. к. мощность обратного отображения $M=\phi^{-1}(y_1-y_2)$ $F_{q^b}\to F_{q^a}$ равна $|M|=q^{a-b}$, а $p_1(X)-p_2(X)$ есть полином степени $\le k$. Тогда $\varepsilon=kq^{a-b}/q^a=k/q^b$. ◊

Замечание 1.8.

1. Если $k=1$ имеем строго универсальный класс хеш-функций $\frac{1}{q^b}-SU(q^{a+b},q^a,q^b)$. Размер ключевых данных N определяется произведением пространства аутентификаторов и пространства сообщений, что уточняет ранее полученную границу из утверждения 1.5.

2. Для почти строго универсального хеширования снижаются требования к размеру ключевых данных, которое ограничивается размерами поля вычислений F_{q^a} и F_{q^b}.

Пример 1.1 [50]. Пусть $q=2$, $a=4$, $b=2$. Построить строго универсальный класс хеш-функций.

Построим простой ортогональный массив $OA_{q^{a-b}}(2,q^a,q^b)$ с помощью линейного отображения $\phi:F_2^4\to F_2^2$ с функцией $f(x)=\phi(ax)+z$. Ортогональный массив будет иметь вид матрицы, в которой строки определяются функциями f_i с параметрами $a_i\in F_{2^4}$, $z_i\in F_{2^2}$, столбцы – значениями $x_i\in F_{2^4}$, а элементы – значениями $y_i\in F_{2^2}$.

Для $z=0$ имеем таблицу 1.12. Если дополнить эту матрицу еще тремя с $z_1=1$, $z_2=\beta$, $z_3=\beta^2$, тогда получим $OA_4(2,16,4)$. Действительно, анализ приведенной матрицы показывает, что существует самое большее $\lambda=4$ функций, для которых справедливо $f(x_1)=y_1$ и $f(x_2)=y_2$. Данный ортогональный массив является семейством строго универсальных хеш-функций. По определению 1.6 имеем следующие параметры. Общее число функций $N=64$. Число записей со значением у в каждом столбце матрицы

отображения $X \to Y$ встречается $\frac{N}{2^m} = 16$ раз. Число функций $f \in H$ таких, что $f(x_1) = y_1$, $f(x_2) = y_2$ не превышает $\nu \le 4$, так как $\lambda = 4$. Вероятность коллизии ε будет равна $\varepsilon \cdot \frac{N}{2^b} = \lambda$, $\varepsilon = \frac{1}{4}$ и имеем $\frac{1}{4} - SU(64,16,4)$ семейство хеш-функций.

Таблица 1.12

Значения функции отображения $f(x) = \phi(ax) + z$, $z = 0$.

$f_i(a_i x_j)$	x_j															
	0	1	α^1	α^2	α^3	α^4	α^5	α^6	α^7	α^8	α^9	α^{10}	α^{11}	α^{12}	α^{13}	α^{14}
$f_0(0x_j)$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
$f_1(\alpha^0 x_j)$	0	0	0	1	β	0	1	β^2	β	1	β	1	β^2	β^2	β^2	β
$f_2(\alpha^1 x_j)$	0	0	1	β	0	1	β^2	β	1	β	1	β^2	β^2	β^2	β	0
$f_3(\alpha^2 x_j)$	0	1	β	0	1	β^2	β	1	β	1	β^2	β^2	β^2	β	0	0
$f_4(\alpha^3 x_j)$	0	β	0	1	β^2	β	1	β	1	β^2	β^2	β^2	β	0	0	1
$f_5(\alpha^4 x_j)$	0	0	1	β^2	β	1	β	1	β^2	β^2	β^2	β	0	0	1	β
$f_6(\alpha^5 x_j)$	0	1	β^2	β	1	β	1	β^2	β^2	β^2	β	0	0	1	β	0
$f_7(\alpha^6 x_j)$	0	β^2	β	1	β	1	β^2	β^2	β^2	β	0	0	1	β	0	1
$f_8(\alpha^7 x_j)$	0	β	1	β	1	β^2	β^2	β^2	β	0	0	1	β	0	1	β^2
$f_9(\alpha^8 x_j)$	0	1	β	1	β^2	β^2	β^2	β	0	0	1	β	0	1	β^2	β
$f_{10}(\alpha^9 x_j)$	0	β	1	β^2	β^2	β^2	β	0	0	1	β	0	1	β^2	β	1
$f_{11}(\alpha^{10} x_j)$	0	1	β^2	β^2	β^2	β	0	0	1	β	0	1	β^2	β	1	β
$f_{12}(\alpha^{11} x_j)$	0	β^2	β^2	β^2	β	0	0	1	β	0	1	β^2	β	1	β	1
$f_{13}(\alpha^{12} x_j)$	0	β^2	β^2	β	0	0	1	β	0	1	β^2	β	1	β	1	β^2
$f_{14}(\alpha^{13} x_j)$	0	β^2	β	0	0	1	β	0	1	β^2	β	1	β	1	β^2	β^2
$f_{15}(\alpha^{14} x_j)$	0	β	0	0	1	β	0	1	β^2	β	1	β	1	β^2	β^2	β^2

Утверждение 1.8. Линейное отображение $\phi : F_{q_1} \to F_{q_2}$ с функцией $f(x) = \phi(ax) + z$, где q_1 и q_2 - простые числа, $q_1 > q_2$, $a \in F_{q_1}$, $z \in F_{q_2}$ приводит к почти строго универсальному хешированию $\frac{2}{q_2} - ASU\ \ q_1 q_2, q_1, q_2\ \ .$

Действительно, пусть q_1 и q_2 - простые числа, $q_1 > q_2$, $t = 2$. Тогда $OA_{\lambda=\lceil q_1/q_2 \rceil}\ 2, q_1, q_2$ - массив, каждая строка которого повторяется самое большее $\lambda = \lceil q_1 / q_2 \rceil$ раза, где $\lceil q_1 / q_2 \rceil$ определяет округление к большему целому. Вероятность коллизии ε по определению строгой универсальности будет равна $\varepsilon \cdot \frac{N}{q_2} = \lambda$, $N = q_1 q_2$, $\varepsilon = \lambda / q_1 = \lceil q_1 / q_2 \rceil / q_1 \le 2 / q_2$ и получим почти строго универсальное хеширование $\frac{2}{q_2} - ASU\ \ q_1 q_2, q_1, q_2$. ◊

Замечание 1.9.

1. Теорема 1.2 определяет строго универсальное хеширование $\frac{1}{q^m} - SU\ \ q^{n+m}, q^n, q^m$ над расширенным полем (см. утверждение 3 [50]).

2. Линейное отображение $\phi : F_q^n \to F_q^m$ определяет умножение элементов в F_{q^n}, проектирование m координат $F_{q^n} \to F_{q^m}$ и сложение в F_{q^m} (см. пример 1.1).

3. Линейное отображение $\phi : F_{q_1} \to F_{q_2}$ где q_1 и q_2 - простые числа, определяет отображение простого конечного поля на простое поле меньшей размерности.

1.4.2 Строго универсальное хеширование на основе почти независимых массивов

Обобщением ортогональных массивов являются почти независимые массивы (almost independent arrays). Теория почти независимых массивов снимает ограничение на равновероятное распределение наборов хешей по столбцам массива. Почти независимые массивы были рассмотрены Стинсоном [48,49] и в рамках этой теории были определены многократные или t связные коды аутентификации.

Определение 1.8 [54]. Пусть $0 \le \varepsilon \le 1$. Массив $n, k\ _p$ является t - связным, ε - зависимым (ε - dependent), если для любого набора U из $s \le t$

столбцов и каждого вектора $a \in F_p^s$ частота $\nu_U\ a$ появления в столбцах значения a удовлетворяет условию $\left|\frac{\nu_U\ a}{n} - \frac{1}{p^s}\right| \le \varepsilon$.

Замечание 1.8. Если массив $n,k\ _p$ является t - связным, независимым (0- зависимым) тогда по определению имеем $\nu_U\ a\ / n = 1/\ p^t$. В этом случае $n,k\ _p$ является ортогональным массивом силы t и образует t - строго универсальное семейство хэш-функций **[**54,55].

Утверждение 1.9 [55]. Пусть $n,k\ _p$-массив, содержащий n строк, k столбцов и записи из набора p элементов. Для $\forall a \in F_p$ частота $\nu_a\ u$ появления значения a в столбцах массива $u = u_1,u_2,...,u_n \in F_p^n$ удовлетворяет условию $\left|\nu_a\ u\ / n - 1/\ p\right| \le \varepsilon_1$ и для любых пар столбцов u, u' частота $\nu_{a,a'}\ u,u'$ появления в столбцах значений a и a' удовлетворяет условию $\left|\nu_{a,a'}\ u,u'\ / n - 1/\ p^2\right| \le \varepsilon_2$. Тогда $n,k\ _p$-массив есть семейство $\varepsilon - ASU\ \ n,k,p$ хеш функций и $\varepsilon = \ p^{-2} + \varepsilon_2\ \big/\ p^{-1} - \varepsilon_1$.

Доказательство. Параметр ε определяется условной вероятностью появления любых записей a,a' для различных столбцов u, u' при равновероятном выборе i строки $\varepsilon = \Pr(u'_i = a' / u_i = a)$. По формуле полной вероятности имеем

$$\Pr(u'_i = a' / u_i = a) = \Pr(u_i = a, u'_i = a') / \Pr(u_i = a).$$

Вероятность появления в произвольно выбранном столбце значения a определяется, как $\Pr(u_i = a) = \nu_a\ u\ / n$ и с условием ограничения $\left|\nu_a\ u\ / n - 1/\ p\right| \le \varepsilon_1$ удовлетворяет неравенству

$$p^{-1} - \varepsilon_1 \le \Pr(u_i = a) \le p^{-1} + \varepsilon_1.$$

Аналогично для вероятности $\Pr(u_i = a, u'_i = a') = \nu_{a,a'}\ u,u'\ / n$ имеем

$$p^{-2} - \varepsilon_2 \le \Pr(u_i = a, u'_i = a') \le p^{-2} + \varepsilon_2.$$

Максимальное значение условной вероятности $\Pr(u'_i = a' / u_i = a) = p^{-2} + \varepsilon_2 \;/\; p^{-1} - \varepsilon_1$ получим, подставляя выражение для полной вероятности $\Pr(u_i = a) = p^{-1} - \varepsilon_1$ и $\Pr(u_i = a, u'_i = a') = p^{-2} + \varepsilon_2$. ◊

Замечание 1.9.

1. Параметр ε-зависимость характеризует отклонение от равномерного распределения совместных вероятностей появления кодовых комбинаций в t произвольных столбцах случайно выбранной строки n,k_p массива. В теории безусловной аутентификации Стинсона $t = 2$ и рассматривается ASU_2 аутентификация.

2. Как следует из утверждения 1.9, значение параметра зависимости ε определяет вероятность коллизии MAC кодов и в общем случае, как показано в [56], коллизионные свойства t - кратных кодов аутентификации. Практическое построение почти независимых массивов является проблематичным, так как нужны методы, которые позволяют формировать массивы хешей с заданными распределениями по столбцам. В этом отношении для построения строго универсальных хеш функций более продуктивным является применение слабо смещенных массивов.

1.4.3 Строго универсальное хеширование на основе слабосмещенных массивов

Слабо смещённые массивы впервые были введены в работах [57,58] для массивов дискретных значений большой размерности с распределением незначительно отличающимся от равномерного. Слабо смещённые массивы определяют свойства распределений хешей в столбцах массива [55].

Определение 1.9. Пусть p - простое число, $u = u_1, u_2, ..., u_n \in F_p^n$. Для $\forall i \in F_p$, $\nu_i\ u$ есть частота появления элемента i в последовательности u $\nu_i\ u = \frac{n}{p} + \delta_i\ u$, где $\delta_i\ u$ - есть отклонение частоты $\nu_i\ u$ от среднего

значения и $\sum_{i\in F_p}\delta_i\ u\ =0$. Пусть ξ комплексный корень p- степени из единицы, тогда смещение вектора u определяется как

$$bias\ u\ =\frac{1}{n}\left|\sum_{i\in F_p}\delta_i\ u\ \xi^i\right|=\frac{1}{n}\left|\sum_{i\in F_p}\nu_i\ u\ \xi^i\right|.$$

Смещение $bias\ u$ имеет следующие свойства.

Утверждение 1.10 [55]. Для произвольного вектора u $0\le bias\ u\ \le 1$ и $bias\ u\ =1$ только тогда, когда $u=const$.

Действительно, по определению нормы $bias\ u$ не может быть меньше нуля. Так как $\sum_{i\in F_p}\xi^i=0$, имеем

$$\frac{1}{n}\left|\sum_{i\in F_p}\nu_i\ u\ \xi^i\right|=\frac{1}{n}\left|\sum_{i\in F_p}\left(\frac{n}{p}+\delta_i\ u\ \right)\xi^i\right|=\frac{1}{n}\left|\sum_{i\in F_p}\delta_i\ u\ \xi^i\right|.$$

Для произвольного вектора u, в силу того, что $\left|\delta_i\ u\ \right|\le\frac{n}{p}$, получим

$$bias\ u\ =\frac{1}{n}\left|\sum_{i\in F_p}\delta_i\ u\ \xi^i\right|\le\frac{1}{n}\sum_{i\in F_p}\left|\nu_i\ u\ \xi^i\right|\le\frac{1}{n}\sum_{i\in F_p}\left|\delta_i\ u\ \right|\left|\xi^i\right|\le 1.$$

Пусть $u=const$, тогда для некоторого $i\in F_p$, имеем $\nu_i\ u\ =n$, а для всех остальных $\nu_j\ u\ =0$ $j\ne i, j\in F_p$. Отсюда следует, что $bias\ u\ =\frac{1}{n}\left|\sum_{i\in F_p}\nu_i\ u\ \xi^i\right|=\frac{1}{n}\left|n\xi^i\right|=1.$ ◊

Определение 1.10. Пусть $n,k\ _p$-массив, содержащий n строк, k столбцов и записи из набора p элементов и $0\le\varepsilon\le 1$. Массив $n,k\ _p$ является ε -смещённым (ε-biased), если любая нетривиальная линейная комбинация столбцов имеет смещение $bias\le\varepsilon$.

Замечание 1.10.

1. Смещение массива является свойством F_p - линейного кода, построенного с помощью столбцов порождающей матрицы.

2. Для двоичных массивов параметр ε смещения прямо связывается с вероятностями появления 0 и 1 в столбцах массива.

3. Для строго универсального класса, массив хеш значений определяется $n,k_{\ p}$ массивом со смещением равным нулю [55].

Утверждение 1.11 [55]. Пусть n,k двоичный ε - смещённый массив, содержащий n строк, k столбцов, тогда вес Хемминга ω любой нетривиальной линейной комбинации столбцов удовлетворяет неравенству

$$\frac{1-\varepsilon}{2} \le \frac{\omega}{n} \le \frac{1+\varepsilon}{2}.$$

Пусть ω_i вес Хемминга нетривиальной линейной комбинации u_i столбцов двоичной матрицы n,k . Тогда $\nu_1\ u\ = \omega_i$, $\nu_0\ u\ = n - \omega_i$ и по определению 1.7 получим $\frac{1}{n}\left|\sum_{i\in F_2} \nu_i\ u\ \xi^i\right| = \frac{1}{n}\left|\nu_0\xi^0 + \nu_1\xi^1\right| = \frac{1}{n}\left|n - 2\omega_i\right| \le \varepsilon$ или $\frac{1-\varepsilon}{2} \le \frac{\omega_i}{n} \le \frac{1+\varepsilon}{2}$. ◊

В общем случае, когда $p \ne 2$ прямого соответствия между смещением и вероятностью появления символов в столбцах массива нет.

Практическим методам построения слабосмещенных массивов является метод сумм экспонент Вейля- Карлитца- Ушиямы (ВКУ).

Определение 1.11 [59]. Метод сумм экспонент ВКУ определяет массив $p^f, f\ *\ n - n/p\ _{\ p}$ со смещением $bias \le\ n-1\ p^{-f/2}$, с записями вида $Tr\ a_j\alpha^i$, где a_j - базис поля $F_{p^f} \mid F_p$, $i \le n$ и i не кратно p, $Tr : F_{p^f} \to F_p$ - след элемента $a_j\alpha^i$.

Пример 1.2 [55]. Построить массив ВКУ $p^f, f\ *\ n - n/p\ _{\ p}$ со смещением $bias \le\ n-1\ p^{-f/2}$ при $p = 2, f = 4, n = 1$. Базисные элементы поля имеют вид $a_j : 1, \alpha, \alpha^2, \alpha^3$. Так как $n = 1$, следует взять только одну экспоненту $\phi : X$. Строки массива индексируются элементами $\alpha \in F_{2^4}$, столбцы –

функциями: $X, \alpha X, \alpha^2 X, \alpha^3 X$, а записи - $Tr\ \beta = \beta + \beta^2 + \beta^4 + \beta^8$. Получим $2^4, 4$ массив со смещением $bias = 1 - 1\ 2^{-2} = 0$.

Пусть $p = 3, f = 2, n = 2.$ Тогда $a_j : 1, \alpha$, $\phi : X, X^2$ и $Tr\ \beta = \beta + \beta^3$. Строки массива индексируются элементами $\alpha \in F_{3^2}$ (порождающий многочлен поля $z^2 + z + 2$), столбцы – функциями: $X, \alpha X, \alpha X^2, X^2 = \alpha^4 X + 1$ (mod $X^2 + X + 2$). Массив $3^2, 4_{\ 3}$ имеет вид представленный в табл. 1.13.

Таблица 1.13

Слабосмещенный массив ВКУ $3^2, 4_{\ 3}$

α^i	X	αX	X^2	αX^2
0	0	0	0	0
α^0	α^4	α^4	α^4	α^4
α^1	α^4	0	0	α^4
α^2	0	α^4	α^0	α^0
α^3	α^4	α^0	0	α^0
α^4	α^0	α^0	α^4	α^4
α^5	α^0	0	0	α^4
α^6	0	α^0	α^0	α^0
α^7	α^0	α^4	0	α^0

Зададим произвольную линейную комбинацию столбцов $Y = \sum_{j=1}^{4} \gamma^j Y_j$, $\gamma_j \in F_3, j = 1,4$ например, $Y = Y_1 + \alpha^4 Y_2 + \alpha^4 Y_4$. Получим результирующий вектор $Y_p = 0, \alpha^0, 0, 0, \alpha^0, \alpha^4, \alpha^0, \alpha^0$. Значения частот элементов $0, \alpha^0, \alpha^4$ равны: $\nu_0 = 4, \delta_0 = +1; \nu_{\alpha^0} = 4, \delta_{\alpha^0} = +1; \nu_{\alpha^4} = 1, \delta_{\alpha^4} = -2$, а смещение

$$bias\ \ v_Y = \frac{1}{9}\left|1 * e^{j\frac{2\pi}{3}*0} + 1 * e^{j\frac{2\pi}{3}*1} + (-2) * e^{j\frac{2\pi}{3}*2}\right| = \frac{1}{9}\left|1 - \frac{1}{2} + \frac{\sqrt{3}}{2} j + 1 + \sqrt{3} j\right| = \frac{1}{3}.$$

Для всех нетривиальных линейных комбинаций столбцов значение $bias \le \frac{1}{3}$ и $bias \le p^{-1}$.

Замечание 1.11.

1. Пусть $f=2$, $n=1$, тогда имеем $p^2,2_p$. Строки массива индексируются элементами $\alpha \in F_{p^2}$, столбцы – функциями: $X, \alpha X$, записи - $Tr\ \beta = \beta + \beta^p$. Значение смещения столбца $bias \le (n-1)\ p^{-f/2}$ будет равно 0. Можно показать, что если $f * n - n/p$ чуть меньше 2, верхняя граница смещения массива $p^2,2_p$ $bias \le p^{-1}$.

2. Линейная комбинация столбцов массива $p^2,2_p$ $Y = \sum_{j=1}^{2} \gamma^j Y_j$, $\gamma_j \in F_p$ имеет смещение $bias = 0$ и значение $Y + \eta$ в строке индексированной α, η, $\alpha \in F_{p^2}$, $\eta \in F_p$ определяет строго универсальный класс $\frac{1}{p} - SU(p^3, p^2, p)$. Это совпадает с результатами теоремы 1.2.

3. Пусть $f=2$, $n=2$, тогда имеем $p^2,4_p$. Строки массива индексируются элементами $\alpha \in F_{p^2}$, столбцы – функциями: $X, \alpha X, X^2, \alpha X^2$, записи - $Tr\ \beta = \beta + \beta^p$. Если $f * n - n/p$ строго равняется 4, значение смещения будет точно равно $bias = p^{-1}$. Можно показать, что если $f * n - n/p$ чуть меньше 4, верхняя граница смещения массива $p^2,4_p$ $bias \le 2/p$. Линейная комбинация столбцов массива $p^2,4_p$ $Y = \sum_{j=1}^{4} \gamma^j Y_j$, $\gamma_j \in F_p$ имеет смещение $bias \le 1/p$ и значение $Y + \eta$ в строке индексированной α, η, $\alpha \in F_{p^2}$, $\eta \in F_p$ определяет почти строго универсальный класс $\frac{1}{p} - ASU(p^3, p^4, p)$.

4. В случае $f=1,\ n=1$, имеем простой ортогональный массив $q,1_q$ с линейным отображением $\phi: F_q \to F_q$ и функцией $f(x) = \phi(ax) + z$, где $a, z \in F_q$.

1.5 Граничные оценки композиционной схемы универсального хеширования

Определение 1.12 [54]. Массив аутентификаторов $n,k_{\ p}$ является ε почти строго универсальным ASU_2, если каждый столбец имеет смещение 0 и для двух записей e,e' одной строки в любых столбцах c,c' условная вероятность $\Pr(c_i = e \mid c_i' = e') \le \varepsilon$ и равномерном распределении номера строка i.

Замечание 1.12.

1. Определение ASU_2 аутентификации вводится в теории безусловной аутентификации, свойства следуют из утверждения 1.7.

2. Строка массива $n,k_{\ p}$ определяется значением ключа, столбец - сообщением источника и значение записи является аутентификационным тегом.

Утверждение 1.12 [55]. Пусть смещение массива $n,k_{\ p}$ равно 0. Тогда $\varepsilon - ASU(n, p^k, p)$ определяет почти строго универсальный хеш класс аутентификаторов.

Число записей в любой нетривиальной линейной комбинации столбцов $n,k_{\ p}$ массива для каждого хеш значения строго равно $\frac{n}{p}$, по определению 1.6 для строго универсального хеш класса. Тогда $\nu_j\ u\ = \frac{n}{p},\ j \in F_p$ и отсюда имеем $bias\ u\ = \frac{1}{n}\left|\sum_{i \in F_p} \nu_i\ u\ \xi^i\right| = \frac{1}{n}\left|\sum_{i \in F_p} \frac{n}{p}\xi^i\right| = \frac{1}{p}\left|\sum_{i \in F_p} \xi^i\right| = 0.$ ◊

Соотношение между смещением и зависимостью установлено в [56].

Теорема 1.3 [56]. Если массив является t - связным и ε – смещенным, он является также и t - связным и ε' - зависимым, причём, $\varepsilon' < \varepsilon$.

Фундаментальное значение этой теоремы заключается в том, что она определяет возможность применения слабо смещённых массивов в схемах аутентификации.

Теорема 1.4 [54]. Пусть C линейный $[n,k,d]_{q=p^m}$ код и B внутренний $(n_0,m)_p$ массив со смещением ε_0. Тогда существует $(nn_0,km)_p$ массив со смещением $\varepsilon = 1-\delta+\delta\varepsilon_0 \le 1-\delta+\varepsilon_0$, где $\delta = \frac{d}{n}$.

Замечание 1.13.

1. Теорема 1.4 определяет двух каскадную композиционную схему построения слабосмещенных массивов. Первый включает порождающую матрицу линейного кода, второй – слабосмещенный массив.

2. Композиционная схема слабосмещенного массива является обобщением конструкции Стинсона [6] для строго универсального хеширования с алгебраическим кодированием.

Граничные оценки параметров для композиционных схем со слабосмещенными массивами представлены теоремой 1.5.

Теорема 1.5 [54]. Пусть C линейный код определенный над $F_{q=p^m}$ и $(p^m,m)_p$ -несмещенный массив (включает все m наборы). Тогда справедливы следующее оценки:

a) $f_p(b,e) = 2(b+e-\log_p m)$ для кодов РС;

b) $f_p(b,e) \le \frac{5}{3}(b+e)$, $b \ge 2e$ для кодов Эрмита;

c) $f_p(b,e) \le \frac{3}{2}(b+e)+2$, $b > 3e$ для кодов Сузуки.

Замечание 1.14.

1. Результат для кодов РС является новым.

Доказательство. Пусть задан РС код над $F_{q=p^m}$ с параметрами $[p^m, \varepsilon p^m, p^m-\varepsilon p^m+1]_{q=p^m}$, $0<\varepsilon<1$ и пусть $\varepsilon = p^{-e}$. Массив $(p^m,m)_p$ является несмещенным и $\varepsilon_0 = 0$. По теореме 1.4 имеем массив $(p^{2m}, m\varepsilon p^m)_p = (p^{2m}, mp^{m-e})_p$ со смещением $\varepsilon = 1-\delta+\delta\varepsilon_0 = 1-(1-p^{-e})-p^{-m} < p^{-e}$ и получим $f_p(b,e) = 2(b+e-\log_p m) = 2m$, где $b = \log_p(m\cdot p^{m-e})$. ◊

2. Результаты для кодов Эрмита и Сузуки впервые представлены в [54].

3. Применение длинных алгебраических кодов дает лучшие результаты для параметров слабосмещенных массивов по сравнению с ВКУ методом.

Пример 1.3 [55]. Построить слабосмещенный массив с РС кодом в $F_{q=p^4}$. Определим РС код с параметрами $p^4, p^3, p^4 - p^3 + 1$ и внутренний массив $p^2, 4_p$ (замечание 1.11). Тогда по теореме 1.4 имеем массив $p^6, 4p^3{}_p$ со смещением $\varepsilon = 1 - \frac{p^4 - p^3 + 1}{p^4} + \frac{p^4 - p^3 + 1}{p^4} * \frac{1}{p} \le \frac{2}{p}$.

Теорема 1.6 [54]. Пусть n,k_p массив со смещением ε_0 и $t \le k$. Тогда существует $\varepsilon - ASU_2(p^t n, p^k, p^t)$ универсальное хеширование, где $\varepsilon = p^{-t} + \varepsilon_0$.

Замечание 1.15.

1. Универсальное хеширование по теореме 1.6 определяется через слабосмещённые массивы, является обобщением конструкций линейных кодов, ВКУ массивов, теоремы 1.2 и утверждения 1.12.

2. Отличие схемы ASU_2 по теореме 1.6 состоит в том, что используется специальное индексирование строк массива аутентификаторов и записей, что увеличивает пространство ключей и записей, и приводит к лучшим оценкам параметров аутентификации.

1.6 Теоретическое обоснование доказово стойкой аутентификации на основе универсального хеширования

Анализ методов универсального и строго универсального хеширования показывает, что решение задачи построения коллизионно–стойких функций хеширования и ключевых функций хеширования которые удовлетворяют международным требованиям гарантированной стойкости к атакам, сложности и скорости вычисления, характеристикам и реализациям алгоритма, возможно в теории доказуемо стойкой аутентификации.

Основные положения теории аутентификации были определены в [28]. Симмонс Г.Д. ввел понятие вероятности обмана и показал, что вероятность обмана имеет нижнюю границу $1/|B|$, где $|B|$ - мощность пространства кодов

аутентификации. Для доказуемо стойкой аутентификации ключевое пространство $|K|$ должно быть не меньше пространства сообщений $|D|$.

Научная задача построения доказуемо секретной аутентификации впервые сформулирована в работе [29]. Решение этой задачи было предложено в классе универсальных хеш функции, как аутентификации с максимальной теоретически достижимой секретностью. В методе универсального хеширования на основе скалярного произведения достигается $P_{кол} = 1/|B|$, при условии что $|K| = |D|$, а в методе полиномиального хеширования $P_{кол} \sim \log|D|$, $|K| = |B|$. Идеи универсальной аутентификации получили развитие в теории безусловной аутентификации с использованием строго универсального хеширования [48]. Теория построение массивов строго универсальных аутентификаторов определяется ортогональными массивами. Основной результат строго универсального хеширования состоит в том, что вероятность коллизии $P_{кол} = 1/|B|$ достигается при условии $|K| \geq |D| \cdot |B|$. Применение слабосмещенных массивов для построения почти строго универсальных хеш функций снимает ограничение на размер ключевого пространства $|K| \geq \cdot |B|^2$, но увеличивает при этом вероятность коллизии $P_{кол} > 1/|B|$.

Основное противоречие доказуемо стойкой аутентификации состоит в том, что для обеспечения гарантированной вероятности обмана на уровне нижней границы, размер ключа должен быть не меньше размера сообщения, а фиксирование размера ключа на нижней границе определяемой мощностью пространства хешей приводит к пропорциональному росту вероятности коллизии от длины данных.

В практическом отношении это означает, что для аутентификации с секретностью $P_{кол} = 1/|B|$ по закрытому каналу связи следует передавать ключевых данных больше, чем по открытому информационных данных.

Анализ методов универсального хеширования показывает, что основными путями разрешения этого противоречия являются универсальное хеширование на основе алгебраического кодирования и композиционное хеширование на

основе почти универсального хеширования и строго универсального хеширования.

Метод универсального хеширования на основе алгебраического кодирования предложен группой авторов в работе [31,32]. Основной результат состоит в том, что вероятность коллизии определяется параметрами алгебраического кода. Это позволило связать вероятность коллизии с длиной сообщения, ключа и полем вычисления хешей. Ключевое пространство определяется длиной кода. Выбор параметров алгебраического кода позволяет оптимизировать затраты на аутентификацию.

Безусловная аутентификация на основе композиционного хеширования предложена в [48,49,60]. Композиционное хеширование эффективно снижает затраты на размер ключевых данных, допуская снижение секретности. Применение алгебраических кодов в композиционных схемах и основные оценки получены в [32,34]. Например, применение кода Рида-Соломона с параметрами $2^{25},16,2^{25}-15_{q=2^{25}}$ позволяет построить строго универсальный класс хеш функций $2^{-20}-SU(2^{71},2^{400},2^{21})$ для источника данных в 400 бит, размера тега 21 бит и размера ключевых данных 71 бит. Применение кодов Эрмита длины $n=2^{36}$ над полем $F_{q=2^{36}}$ для аутентификации 2^{34} бит данных приводит к хеш классу $2^{-20}-SU(2^{111},2^{2^{34}},2^{21})$ при размере ключевых данных 111 бит. Если для кодирования 2^{57} бит источника использовать коды Сузуки длины $n=2^{78}$ над конечным полем $F_{q=2^{39}}$ получим $2^{-20}-SU(2^{138},2^{2^{57}},2^{21})$ хеширование на 138 битном ключе. Эти же результаты на ортогональных массивах достигаются на ключе в 2^{57} бит.

Как следует из анализа, лучшие результаты для строгой аутентификации достигаются на длинных алгебраических кодах. Коды Эрмита и Сузуки относятся к классу алгеброгеометрических кодов определённых над полем рациональных функций ассоциированным с алгебраической кривой Эрмита и группой Сузуки, что определяет актуальность построения теории универсального хеширования над полем рациональных функций алгебраических

кривых. Таким образом, **целью работы** является разработка теории универсального хеширования по алгебраическим кривым для построения доказуемо стойкой аутентификации.

Научная проблема состоит в разработке теории универсального хеширования по рациональным функциям алгебраических кривых для построения доказуемо стойкой аутентификации сообщений, с обеспечением гарантированной вероятности коллизии и минимизацией затрат на ключевое пространство, размер хеш кода и сложность вычислений.

Функция цели (Z) состоит в обеспечении гарантированной вероятности коллизии $P_{кол}$, минимизации затрат на ключевое пространство $|K|$, сложность вычислений $N_{опер}$ в условиях фиксированной длины сообщения $\log|M|$, поля вычисления F_q и множества алгебраических кривых $F_q(C)$:

$$Z = \min\ P_{кол}, |K|, N_{опер} \mid \log|M| = fix, F_q = fix, F_q(C) = \text{var}.$$

Для достижения поставленной цели необходимо решить следующие частные задачи:

1. разработать метод универсального хеширования по рациональным функциям алгебраических кривых, обосновать требования к выбору алгебраических кривых и их параметрам;

2. построить линейные векторные пространства максимальных кривых, выполнить теоретические обоснования функциональных полей, размерности и структуры подгруппы Вейерштрасса полюсов рациональных функций;

3. разработать метод подсчета числа точек кривых Ферма на основе вычисления мультипликативных подгрупп конечного поля, получить точные оценки числа точек кривых, определить наилучшие кривые по числу точек и отношению числа точек к роду для построения универсального хеширования по кривым Ферма в конечном поле;

4. разработать метод оценки числа точек кривых Ферма на основе вероятностного подхода, построить алгоритм вычисления числа точек, получить асимптотические оценки числа точек и их отношения к границе Хассе–Вейля;

5. разработать метод подсчета числа точек кривых Гурвица, определить кривые с большим числом точек и наилучшие кривые по отношению числа точек к её роду для построения универсального хеширования в простом, квадратичном и кубическом конечном поле;

6. разработать метод построения нетривиальных кривых Гурвица, определить условия существования нетривиальных кривых и обобщенных кривых Гурвица;

7. разработать метод построения максимальных кривых Гурвица, построить максимальные кривые Гурвица наибольшего рода;

8. разработать метод вычисления хеш функций, построить алгоритмы универсального хеширования по алгебраическим кривым, получить коллизионные оценки и оценки сложности универсального хеширования;

9. разработать метод каскадного универсального хеширования со связкой хеша с текстом, получить оценки параметров каскадного универсального хеширования по алгебраическим кривым;

10. разработать метод каскадного универсального хеширования на основе произведения функциональных полей алгебраических кривых, получить оценки параметров универсального хеширования и рекомендации по применению.

2 АНАЛИЗ И ОБОСНОВАНИЕ ТРЕБОВАНИЙ К АЛГОРИТМАМ ХЭШ-ФУНКЦИЙ

2.1 Требования, предъявляемые к криптографическим хэш-функциям

В настоящее время практически ни одно приложение криптографии не обходится без использования хэширования. Хэш-функции – это функции, предназначенные для «сжатия» произвольного сообщения или набора данных, записанных, как правило, в двоичном алфавите, в некоторую битовую комбинацию фиксированной длины, называемую сверткой. Хэш-функции имеют разнообразные применения при проведении статистических экспериментов, при тестировании логических устройств, при построении алгоритмов быстрого поиска и проверки целостности записей в базах данных. Основным требованием к хэш-функциям является равномерность распределения их значений при случайном выборе значений аргумента. Криптографической хэш-функцией называется всякая хэш-функция, являющаяся криптостойкой, то есть удовлетворяющая ряду требований специфичных для криптографических приложений.

В криптографии хэш-функции применяются для решения следующих задач: построения систем контроля целостности данных при их передаче или хранении, аутентификация источника данных.

Хэш-функцией называется всякая функция $h:X \to Y$, легко вычислимая и такая, что для любого сообщения M значение $h(M) = H$ *(свертка)* имеет фиксированную битовую длину. X — множество всех сообщений, Y — множество двоичных векторов фиксированной длины.

Как правило хэш-функции строят на основе так называемых одношаговых сжимающих функций $y = f(x_1, x_2)$ двух переменных, где x_1, x_2 и y — двоичные векторы длины m, n и n соответственно, причем n — длина свертки, а m — длина блока сообщения. Для получения значения $h(M)$ сообщение сначала разбивается на блоки длины m (при этом, если длина сообщения не кратна m

то последний блок неким специальным образом дополняется до полного), а затем к полученным блокам M_1, M_2,.., M_N применяют следующую последовательную процедуру вычисления свертки:

$$H_o = v, H_i = f(M_i, H_{i-1}),\ i = 1,.., N, h(M) = H_N\ , \qquad (2.1)$$

где v — некоторая константа, часто ее называют инициализирующим вектором. Она выбирается из различных соображений и может представлять собой секретную константу или набор случайных данных (выборку даты и времени, например).

При таком подходе свойства хэш-функции полностью определяются свойствами одношаговой сжимающей функции.

Выделяют два важных вида криптографических хэш-функций — ключевые и бесключевые. Ключевые хэш-функции называют кодами аутентификации сообщений. Они дают возможность без дополнительных средств гарантировать как правильность источника данных, так и целостность данных в системах с доверяющими друг другу пользователями. Бесключевые хэш-функции называются кодами обнаружения ошибок. Они дают возможность с помощью дополнительных средств (шифрования, например) гарантировать целостность данных. Эти хэш-функции могут применяться в системах как с доверяющими, так и не доверяющими друг другу пользователями.

Основным требованием к хэш-функциям является равномерность распределения их значений при случайном выборе значений аргумента. Для криптографических хэш-функций также важно, чтобы при малейшем изменении аргумента значение функции сильно изменялось. Это называется лавинным эффектом.

К ключевым функциям хэширования предъявляются следующие требования:

– невозможность фабрикации,

– невозможность модификации.

Первое требование означает высокую сложность подбора сообщения с правильным значением свертки. Второе — высокую сложность подбора для заданного сообщения с известным значением свертки другого сообщения с правильным значением свертки.

К бесключевым функциям хэширования предъявляют требования:

– однонаправленность,

– устойчивость к коллизиям,

– устойчивость к нахождению второго прообраза.

Под одно направленностью понимают высокую сложность нахождения сообщения по заданному значению свертки. Следует заметить, что на данный момент нет используемых хэш-функций с доказанной одно направленностью.

Под устойчивостью к коллизиям понимают сложность нахождения пары сообщений с одинаковыми значениями свертки. Обычно именно нахождение способа построения коллизий криптоаналитиками служит первым сигналом устаревания алгоритма и необходимости его скорой замены.

Под устойчивостью к нахождению второго прообраза понимают сложность нахождения второго сообщения с тем же значением свертки для заданного сообщения с известным значением свертки.

2.1.1 Общие требования к алгоритмам хэширования

Хэш-функции являются одним из важнейших криптографических примитивов для получения цифровых отпечатков сообщений, в том числе заверяемых электронной подписью, кодов аутентификации сообщений по симметричному ключу, генерации псевдослучайных чисел, разворачиванию ключей из входного значения и множества других протоколов.

SHA-3 должен поддерживать размер выходного блока 224, 256, 384 и 512 бит. 160-битные блоки не предусмотрены из-за возможности нахождения коллизий атаками грубой силы (перебора вариантов). Сохраняются те же требования, что и к предыдущим хэш-функциям: максимальный размер входного значения, размер выходного значения, коллизионная стойкость, стойкость к нахождению прообраза и второго прообраза, потоковый режим вычисления "за один проход".

Алгоритмы функций вычисления для разного размера блоков должны быть идентичны и иметь минимум отличий в реализации. Использования абсолютно разных наборов алгоритмов для получения четырёх фиксированных значений длины выхода не допускается.

Однако выдвигаются и пожелания по усовершенствованию существующих алгоритмов работы хэш-функций: возможно включение опции рандомизированного хэширования, улучшенное распараллеливание, оптимальная работа на множестве современных платформ (включая как 64-битные процессоры так и 8-битные процессоры и смарт-карты). Отход от потенциально нестойкого метода Дамгарда-Меркла, на котором построено большинство существующих хэш-функций.

Сохранение или увеличение эффективности по сравнению с SHA-2 и создать класс хэш-функций, на которые потенциально не будут действовать атаки, нацеленные на SHA-2. Допускается параметризация (изменение параметров): например, числа раундов с учётом допустимых пределов

В хэш-функции класса SHA-3 могут быть встроены процедуры согласования для вычисления кодов аутентификации сообщений (HMAC): например, передача в качестве одного из входных параметров длины блока входного сообщения, если она заранее точно известна.

Значение всех встроенных параметров и констант должны быть сконструированы таким образом, чтобы не дать возможности встраивания лазеек и отмычек со стороны разработчиков, для чего должны быть приведены соответствующие доказательства и процедуры проверки.

Специфические требования к стойкости (значения указаны на уровне теоретических пределов, в реальности они могут быть немного меньше) на размер выходного блока в n бит такие:

– в конструкциях кодов аутентификации сообщений (HMAC) и псевдослучайных функций (PRF) стойкость по числу запросов должна быть не менее $2^{(n/2)}$ бит против атак-различителей;

– стойкость рандомизированного хэширования против атаки нахождения $H(M_1, r_1) = H(M_2, r_2)$ — не менее n бит, при условии, что значение рандомизатора r_1 не контролируется атакующим;

Стандартные и дополнительные параметры стойкости:

– стойкость к нахождению коллизий — не менее $n/2$ бит;

– стойкость к нахождению прообраза — n бит;

– стойкость к нахождению второго прообраза — n-k битов для любого сообщения, короче 2^k битов;

– устойчивость к атакам на изменение длины сообщения;

– подмножество хэш-функций (полученное усечением числа битов выхода) размером m битов должно сохранять свойства n-битного множества в пересчёте на m, с учётом статистических (неотличимых от случайных) отклонений. Не должно существовать атаки на быстрое нахождение малостойких подмножеств хэш-функции из исходного множества n;

– устойчивость к новым типам атак, на основе мультиколлизий.

2.1.2 Критерии выбора финалистов конкурса SHA-3

НИСТ выбрал пять кандидатов на алгоритм SHA-3, которые прошли в третий (и финальный) раунд: BLAKE, Grøstl, JH, Keccak, Skein. Основным критерием выбора алгоритмов стал высочайший уровень стойкости и достаточно высокая производительность, гибкость и оптимальность в программной и аппаратной реализации.

Производительность многомерна: ни один алгоритм не является лучшим в каждом из измерений. Каждый кандидат второго раунда достиг по крайней мере приемлемого уровня производительности на лидирующих десктопных и серверных системах, хотя уровни производительности значительно отличаются. Ещё больше разницы на ограниченных платформах и в аппаратном исполнении, в такой области, где наиболее важным фактором производительности является скорость. Множество алгоритмов уменшило свою привлекательность или было отвергнуто за счёт очень больших требований к объёму — стало заметно, что требуемый ими объём не позволяет использовать их в достаточно большом числе потенциальных приложений.

Некоторые алгоритмы позволяли очень высокий уровень тонко спланированного параллелизма, который может быть реализован в аппаратном исполнении, некоторые использовали параллелизм векторных устройств, другие выглядели способными полностью эксплуатировать значительный параллелизм конвенциональных суперскалярных арифметико-логических устройств (АЛУ), которые могут одновременно запускать множество инструкций за один вычислительный шаг. Многие алгоритмы также использовали мощь 64-битных АЛУ.

2.2 АНАЛИЗ ПЕРСПЕКТИВНЫХ АЛГОРИТМОВ ХЭШИРОВАНИЯ ФИНАЛЬНОГО РАУНДА SHA-3

2.2.1 Основные параметры и свойства хэш-функции BLAKE

Авторы алгоритма BLAKE – Жан-Филипп Омассон (Jean-Philippe Aumasson), Лука Хензен (Luca Henzen), Уилли Мейер (Willi Meier) и Рафаэль Фан (Raphael Phan).

BLAKE построен по усиленной схеме Меркля-Дамгаарда [1]. Его основой являются преобразования двух более ранних алгоритмов хэширования – LAKE и ChaCha.

Допустимые размеры выходных значений: 224, 256, 384 и 512 бит. Алгоритм разбивает данные на блоки по 512 бит (для 224- и 256-битных хэш-значений) или по 1024 бит (для 384- и 512-битных хэш-значений). Дополнение последнего блока хэшируемых данных производится следующим образом [2]:

- добавляется единичный бит;
- добавляется необходимое количество нулевых бит;
- добавляется еще один единичный бит;
- добавляется 64-битное значение, равное размеру исходного сообщения.

Помимо основных требований к алгоритму хэширования (включая криптостойкость и быстродействие), при разработке BLAKE авторы ставили целью обеспечить следующие свойства алгоритма:

- прозрачное использование дополнительного параметра (salt) при необходимости;
- возможность распараллеливания вычислений;
- возможность различных оптимизаций при реализации алгоритма;
- возможность реализации на платформах с минимальными ресурсами.

Функция сжатия алгоритма инициализируется выходным значением предыдущей функции сжатия (или вектором инициализации – Initialization Vector, *IV*) с участием дополнительного параметра (при его использовании) и количества уже обработанных бит сообщения (счетчик). Затем выполняется 10 или 14 раундов преобразований (в зависимости от размера выходного значения) с участием собственно блока хэшируемых данных. Внутреннее состояние функции сжатия является достаточно большим и представляет собой двумерный массив 4 X 4, каждая ячейка которого представляет собой слово данных – 32 или 64 бита в зависимости от размера выходного значения. Выходное значение функции сжатия формируется в рамках финального преобразования функции сжатия, в котором также участвует и предыдущее выходное значение – для обеспечения прямой связи. Размер выходного значения – 8 слов. Каждый раунд состоит из 8 вызовов функции *G*(), которая обрабатывает 4 из 16 слов состояния: сначала поочередно или параллельно обрабатываются столбцы массива состояния, затем – диагонали. Функция *G*() обрабатывает входные слова (обозначенные как *a*, *b*, *c*, *d*) следующим образом (см. Рисунок 1.1):

$$\begin{aligned} a &= a + b + (m_x \oplus c_y); \\ d &= (d \oplus a) >>> 16; \\ c &= c + d; \\ b &= (b \oplus c) >>> 12; \\ a &= a + b + (m_y \oplus c_x); \\ d &= (d \oplus a) >>> 8; \\ c &= c + d; \\ b &= (b \oplus c) >>> 7, \end{aligned} \qquad (2.2)$$

где m_i – используемое слово блока сообщения, i зависит от номера раунда и номера вызова функции *G*();

c_i – используемая константа.

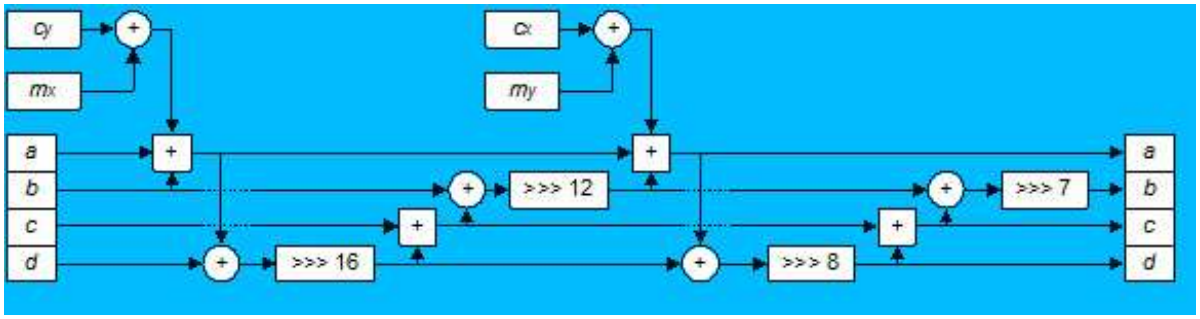


Рисунок 2.1 – Основное преобразование алгоритма BLAKE

Операции сложения выполняются по модулю 2^w, где w – размер слова в битах. Выше указаны преобразования функции $G()$ для 224- или 256-битного выходного значения. Функция $G()$, используемая при вычислении хэш-значений остальных размеров, отличается лишь количеством бит, на которые выполняются операции вращения слов.

Как видно, алгоритм имеет четкую и ясную структуру и легко реализуем, как программно, так и аппаратно. Основное преобразование алгоритма – функция $G()$ – унаследована в незначительно модифицированном виде от алгоритма ChaCha, анализ которого не выявил каких-либо проблем с криптостойкостью.

Авторы работ, в которых проводится анализ криптостойкости алгоритма BLAKE, предложили только сертификационные атаки на варианты алгоритма с усеченным количеством раундов:

– несколько атак на BLAKE с 2,5 раунда; пример атаки – нахождение прообраза для 512-битного хэш-значения с трудоемкостью 2^{481} операций

– метод поиска near-коллизии для функции сжатия 256-битного варианта алгоритма с 4 раундами с трудоемкостью 2^{42} операций [4].

Таким образом, можно утверждать, что алгоритм BLAKE имеет весьма высокий запас криптостойкости.

BLAKE-256 использует 32-разрядные слова, в то время как BLAKE-512 использует 64-разрядные слова. BLAKE – одна из пяти хэш-функций в

финале NIST SHA 3. BLAKE – один из самых простых для реализации проектов и полагается на ранее созданные компоненты: функцию HAIFA и ядро ChaCha.

BLAKE присуща высокая степень безопасности и высокопроизводительная универсальность:

На Core-i5-2400M Intel (Sandy Bridge), BLAKE-256 может хэшировать в 7.49 циклах/байт и BLAKE-512 в 5.64 циклах/байт (подробные данные).

На AMD FX-8120 (Бульдозер) BLAKE-256 может хэшировать в 11.83 циклах/байт и BLAKE-512 в 6.88 циклах/байт (подробные данные).

На базируемом микроконтроллере M3 (32-разрядный процессор), BLAKE-256 может быть реализован с 280 байт RAM и 1320 байт ROM, а BLAKE-512 с 516 байт RAM и 1776 байт ROM (подробные данные).

На микроконтроллере ATmega1284P (8-разрядный процессор), BLAKE-256 может быть реализован с 267 байт RAM и 3434 байт ROM и BLAKE-512 с 525 байт RAM и 6350 байт ROM (подробные данные).

На Xilinx Virtex 5 FPGA, BLAKE-256, реализованный с 56 блоками, могут достигнуть пропускной способности больше чем 160 Мбит/с, и BLAKE-512 с 108 блоками может достигнуть пропускной способности больше чем 270 Мбит/с (подробные данные).

В ASIC на 180 нМ BLAKE-256 может быть реализован с 13.5 kGE. В ASIC на 90 нМ BLAKE-256, реализованный с 38 kGE, может достигнуть пропускной способности больше чем 10 Гбит/с, и BLAKE-512 с 79 kGE может достигнуть пропускной способности больше чем 15 Гбит/с

Увеличенная версия BLAKE состоит из циклов: 14 вместо 10 для BLAKE-224 и BLAKE-256, и 16 вместо 14 для BLAKE-384 и BLAKE-512. Это мотивируется высокой скоростью BLAKE, и таким образом возможно выбрать запас надежности при этом BLAKE остается более быстрым чем SHA 2 на ряде платформ.

2.2.2 Основные параметры и свойства хэш-функции Grøstl

Алгоритм Grøstl разработан большим коллективом специалистов из Технического университета Дании и Технологического университета г. Грац, Австрия. При разработке данного алгоритма его авторы преследовали следующие цели [5]:

- простота анализа алгоритма;
- доказанная криптостойкость алгоритма;
- хорошее быстродействие на различных платформах;
- «встроенная» защита от атак, использующих утечки данных по побочным каналам;
- возможность распараллеливания вычислений.

Алгоритм построен по схеме Меркля-Дамгаарда с финальным преобразованием. Возможные хэш-значения – от 8 до 512 бит, кратные 8 битам. При размерах, меньших или равных 256, входные данные разбиваются на блоки по 512 бит, в остальных случаях – по 1024 бита. Размер внутреннего состояния алгоритма равен размеру блока хэшируемых данных.

Дополнение последнего блока выполняется следующим образом:

- добавляется единичный бит;
- добавляется необходимое количество нулевых бит;
- добавляется 64-битное количество блоков входных данных после дополнения.

Функция сжатия является крайне простой (Рисунок 2.2). Она построена на двух преобразованиях – P() и Q(), которые определенным образом обрабатывают блок сообщения и внутреннее состояние (с предварительно наложенным на него операцией XOR блоком сообщения), после чего результаты работы данных функций объединяются операцией XOR. Такая структура обеспечивает быстрое и эффективное наложение блока сообщения на внутреннее состояние, а также связь вперед, усиливающую криптостойкость

против некоторых видов атак. Функция сжатия алгоритма хэширования Grøstl представлена на рисунке 2.2

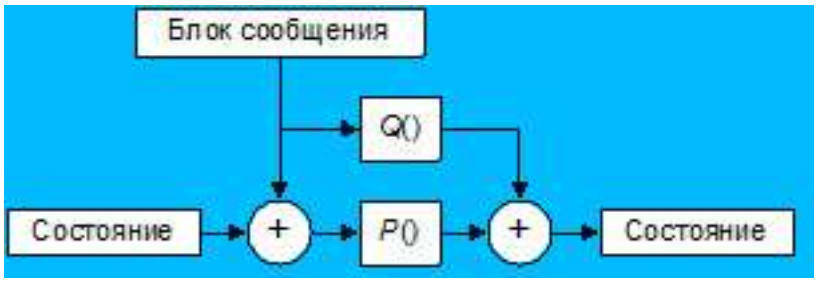


Рисунок 2.2 – Функция сжатия алгоритма Grøstl

Преобразования P() и Q() целиком основаны на незначительно модифицированных преобразованиях алгоритма шифрования AES, каждое из них выполняет несколько раундов (10 или 14 в зависимости от размера состояния) следующих операций над состоянием, которое представляется в виде двумерного байтового массива:

– наложение раундовой константы (преобразования P() и Q() отличаются друг от друга только различными значениями раундовых констант);

– байтовая замена (используется таблица замен алгоритма AES);

– вращение строк массива;

– умножение столбцов массива на матрицу.

Финальное преобразование выполняет обработку состояния (полученного после обработки последнего блока сообщения) функцией P(); результат данной обработки накладывается на состояние операцией XOR, после чего выполняется усечение состояния до требуемого размера хэш-значения.

Алгоритм весьма простой в реализации и обладает прозрачной и легкой для анализа структурой.

Известны несколько работ, посвященных криптоанализу алгоритма Grøstl (например, [6-8]), однако, какие-либо уязвимости данного алгоритма в этих работах не обнаружены.

Grøstl в качестве основной конструкции использует схему AES. Авторы достигают скоростей до 21.4 циклов на байт на Intel Core 2 Duo.

Grøstl – выполненная с помощью итераций хэш-функция, где функция сжатия создана из двух фиксированных, больших, различных перестановок. Проект Grøstl прозрачен и основан на принципах, очень отличающихся от используемых в семействе SHA. Как другие хэш-функции в семействе MD5/SHA, Grøstl делит ввод на блоки и многократно вычисляет ключ f. Однако, Grøstl поддерживает размер хэширование 512 или 1024 бита.

Функция сжатия f основана на 256-или 512-битных функциях

Grøstl – байтовая сеть SP, которая перенимает компоненты от AES.

Grøstl – так называемая конструкция широкого канала, где размер внутреннего состояния значительно больше чем размер вывода. Это приводит к эффекту, что все известные, универсальные атаки на хэш-функцию становятся почти не возможными. У Grøstl хорошая производительность на различных платформах, и контрмеры против атак на AES. Grøstl – набор хэш-функций, способных к возврату дайджестов сообщения любого числа байт от 1 до 64, то есть, от 8 до 512 битов в 8-разрядных словах. Различный возврат и биты вызывают Grøstl-n. Размеры сообщения 224, 256, 384 и 512 бит.

2.2.3 Основные параметры и свойства хэш-функции JH

Автор алгоритма JH – Хонгджун Ву (Hongjun Wu) из Института исследований в области телекоммуникаций, Сингапур. Алгоритм построен по схеме Меркля-Дамгаарда. Входное сообщение разбивается на блоки по 512 бит. Возможные размеры хэш-значений – 224, 256, 384 и 512 бит. Независимо от размера выходного значения алгоритма в нем используется

одна и та же функция сжатия, формирующая после обработки последнего блока сообщения 1024-битное значение, которое усекается до требуемого размера [9].

Функция сжатия в качестве входных параметров использует 512-битный блок сообщения и 1024-битное выходное значение функции сжатия после обработки предыдущего блока H_{i-1} (или значение IV). Она выполняет следующие действия (Рисунок 2.3):

– блок сообщения M_i накладывается на левую 512-битную половину значения H_{i-1} операцией XOR,

– результат предыдущей операции обрабатывается функцией E(), которая подробно описана далее, M_i накладывается на правую половину 1024-битного выходного значения функции E() операцией XOR, в результате получается значение H_i.

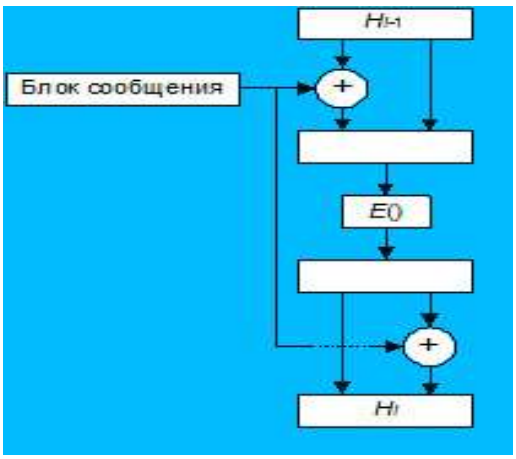


Рисунок 2.3 – Функция сжатия алгоритма JH

1024-битное состояние представляется функцией сжатия в виде 8-мерного массива, каждое измерение которого содержит 2 слова по 4 бита. Данная функция представляет собой блочный шифр в виде подстановочно-перестановочной сети и выполняет 35 раундов преобразований (а также дополнительный уровень табличных замен после последнего раунда – см. далее), в каждом из которых производятся следующие действия:

– уровень табличных замен – каждое слово состояния прогоняется через таблицу замен S_0 или S_1 в зависимости от значения определенного бита раундовой константы; используются таблицы замен известного алгоритма шифрования Lucifer;

– линейное преобразование, поочередно обрабатывающее по два слова состояния с помощью операций XOR над определенными битами входных слов;

– перестановка слов состояния по определенному закону.

Автор алгоритма JH привел также несколько его вариантов, отличающихся количеством измерений массива состояния, что позволяет без изменения используемых в алгоритме преобразований варьировать размером блока данных и состояния, что в итоге напрямую влияет на требования алгоритма к ресурсам при его реализации, а также на его быстродействие.

Флориан Мендель (Florian Mendel) и Сорен Томсен (Soren Thomsen) предложили сертификационную атаку на 512-битный вариант алгоритма JH, которой для успешного вычисления прообраза требуется $2^{510,3}$ операций и столько же блоков памяти. Однако, автор JH доказал в, что в работе Менделя и Томсена не учитываются операции сравнения и доступа к памяти, общее число которых составляет 2^{525}, т. е. трудоемкость данной атаки превышает теоретическую – 2^{512} операций. JH – семейство из четырех криптографических хэш-функций: JH-224, JH-256, JH-384 и JH-512.Алгоритмы этих хэш-функций отличаются только значением одного внутреннего параметра – длины (в битах) выходного значения.

2.2.4 Основные параметры и свойства хэш-функции Keccak

Авторы алгоритма Keccak – Гвидо Бертони (Guido Bertoni), Джоан Деймен (Joan Daemen), Михаэль Петерс (Michael Peeters) и Жиль Ван Аске (Gilles Van Assche).

Алгоритм построен по принципу криптографической Sponge (данная структура криптографических алгоритмов была предложена авторами алгоритма Keccak ранее). Входное сообщение разбивается на блоки по 1024 бит. Основой функции сжатия алгоритма является функция *f*(), выполняющая перемешивание внутреннего состояния алгоритма. Состояние (обозначим его *A*) представляется в виде двумерного массива 5 X 5, элементами которого являются 64-битные слова, инициализированные нулевыми битами (т. е. размер состояния составляет 1600 битов).

Функция *f*() выполняет 18 раундов, в каждом из которых производятся следующие действия (см. Рисунок 2.4, номера на рисунке обозначают последовательность операций):

$$C[x] = A[x, 0] \oplus A[x, 1] \oplus A[x, 2] \oplus A[x, 3] \oplus A[x, 4], \; x = 0\ldots4;$$

$$D[x] = C[x - 1] \oplus (C[x + 1] >>> 1), \; x = 0\ldots4;$$

$$A[x, y] = A[x, y] \oplus D[x], \; x = 0\ldots4, \; y = 0\ldots4;$$

$$B[y, 2x + 3y] = A[x, y] >>> r[x, y], \; x = 0\ldots4, \; y = 0\ldots4;$$

$$A[x, y] = B[x, y] \oplus (\sim B[x + 1, y] \; \& \; B[x + 2, y], \; x = 0\ldots4, \; y = 0\ldots4, \qquad (2.3)$$

где *B* – временный массив, аналогичный по структуре массиву состояния;

C и *D* – временные массивы, содержащие по 5 64-битных слов;

r – массив, определяющий количество битов вращения для каждого слова состояния;

~*x* – побитовый комплемент к *x*;

операции с индексами массива выполняются по модулю 5.

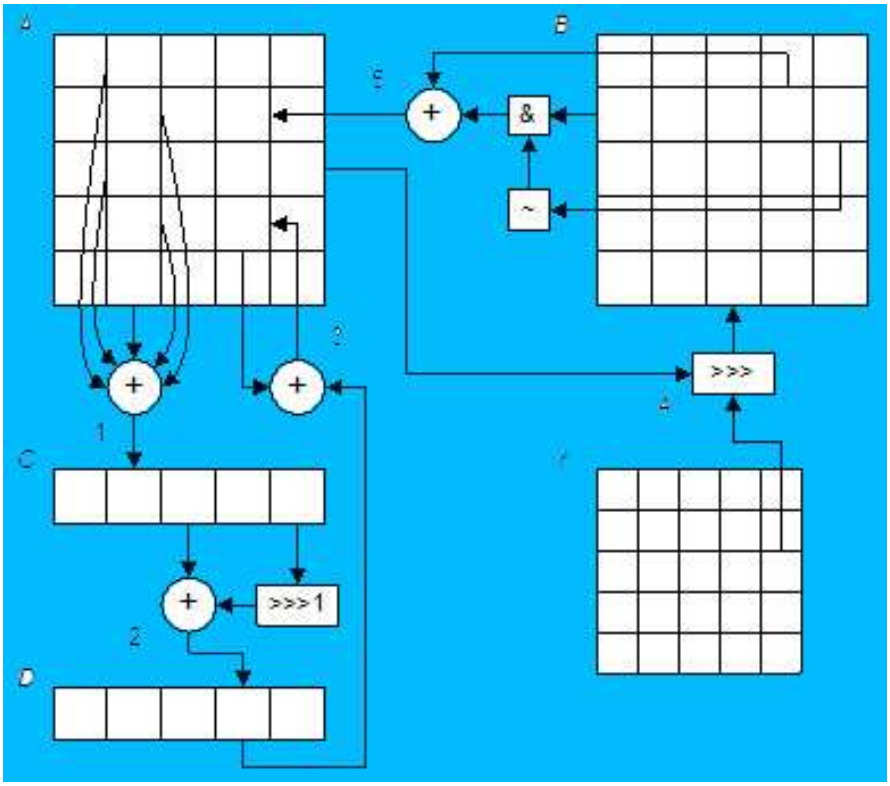


Рисунок 2.4 – Раунд алгоритма Keccak

Кроме приведенных выше операций, в каждом раунде также выполняется наложение операцией XOR раундовой константы на слово *A*[0, 0].Перед выполнением функции сжатия на часть состояния операцией XOR накладывается блок хэшируемых данных. Данное наложение в совокупности с функцией сжатия, выполняемые для каждого блока входных данных, представляют собой «впитывающую» (absorbing) фазу криптографической Sponge.Стоит отметить, что функция *f*() использует только операции, стойкие к атакам, использующим утечки данных по побочным каналам.

Результирующее хэш-значение вычисляется в процессе выполнения «сжимающей» (squeezing) фазы криптографической Sponge, основу которой также составляет описанная выше функция *f*(). Возможные размеры хэш-значений – 224, 256, 384 и 512 бит.

Выше описан вариант алгоритма Keccak, представленный на конкурс SHA-3. Оригинальный алгоритм Keccak имеет множество настраиваемых

параметров с целью обеспечения оптимального соотношения криптостойкости и быстродействия для определенного применения алгоритма на определенной платформе. Настраиваемыми величинами являются: размер блока данных, размер состояния алгоритма, количество раундов в функции *f*() и другие.

Авторы Keccak учредили ряд призов за достижения в криптоанализе данного алгоритма. В 2009 г. появилось несколько работ, посвященных криптоанализу алгоритма Keccak (например, [14-16]), однако, их авторы не обнаружили каких-либо уязвимостей в данном алгоритме.

Авторы заявляют о 12.5 циклах на байт при выполнении на ПК с процессором Intel Core 2. Keccak реализован на базе конструкции «sponge». Внутреннее состояние может быть длиной 25, 50, 100, 200, 400, 800 или 1600 бит. Заявленное число раундов, необходимое для защиты от предполагаемых атак — 18.

Хэш-алгоритм имеет два варианта реализации — 32-разрядный и 64-разрядный, который выиграл конкурс SHA-3.Sponge функция является обобщением понятия криптографической хэш-функции с бесконечным выходом и может выполнять почти все симметричные криптографические функции: хэширование, генерация псевдослучайных чисел.

Keccak называют одним из семи перестановок имени Keccak-F [б], = 25, 50, 100, 200, 400, 800 или 1600. В рамках SHA-3 конкурса, Keccak-F [1600], более "легкий" можно использовать в ограниченных средах. Выбор операции ограничены побитовое XOR, AND и NOT и поворотов.

2.2.5 Основные параметры и свойства хэш-функции Skein

Алгоритм Skein разработан командой всемирно известных специалистов-криптологов, представляющих известные университеты и корпорации (преимущественно из США).

Алгоритм предложен на конкурс SHA-3 в трех следующих вариантах [17]:

Skein-512 – основной вариант;

Skein-1024 – усиленный вариант;

Skein-256 – вариант для применений в условиях ограниченных ресурсов.

Число в названии варианта обозначает размер внутреннего состояния алгоритма в битах. Рекомендуемые размеры выходного значения алгоритма: 128, 160, 224, 256, 384 и 512 битов, что соответствует размерам хэш-значений алгоритмов MD5 и SHA-1/SHA-2, на замену которых претендует алгоритм Skein. Основой функции сжатия алгоритма Skein является блочный шифр Threefish, размеры блока которого соответствуют размерам состояния алгоритма Skein. Данный алгоритм шифрования представляет собой подстановочно-перестановочную сеть, основой которой является операция *mix*(), обрабатывающая два 64-битных слова $x0$ и $x1$ следующим образом (рис. 2.5).

$$y0 = x0 + x1 \bmod 2^{64};$$
$$y1 = (x1 <<< R) + y0, \qquad (2.4)$$

где $y0$ и $y1$ – выходные значения операции, а R – одна из заданных таблицей констант вращения.

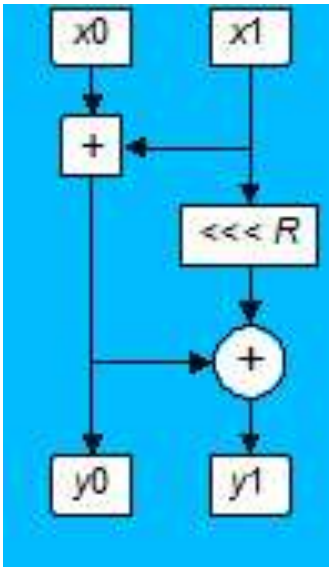


Рисунок 2.5 – Операция mix() алгоритма Threefish (Skein)

Раунд алгоритма представляет собой последовательность следующих операций:

– параллельное применение операций *mix*() к каждым двум соседним словам состояния;

– перестановка слов состояния.

Кроме того, перед первым раундом и после каждых четырех раундов выполняется наложение соответствующего фрагмента расширенного ключа шифрования операцией сложения по модулю 2^n, где n – размер состояния в битах.

Threefish выполняет 72 раунда для 256- или 512-битного блока или 80 раундов для 1024-битного блока. Размер ключа шифрования алгоритма эквивалентен размеру блока шифруемых данных. Кроме того, в качестве дополнительного параметра алгоритма используется 128-битное модифицирующее значение (tweak), которое имеет определенную структуру. На верхнем уровне алгоритм использует разработанную его авторами схему UBI (Unique Block Iteration – уникальная итерация блока).

Алгоритм Skein выглядит крайне простым как для анализа на предмет возможных уязвимостей, так и для реализации. Алгоритм гибок для применения на различных платформах и с различными целями и крайне эффективен при хэшировании как длинных, так и коротких сообщений. Криптоанализ алгоритма проводился, в основном, в работе [18], авторы которой достигли, в частности, следующих результатов против Skein-512:

Основной концепцией разработки была оптимизация под минимальное использование памяти, криптографически безопасное хэширование небольших сообщений, устойчивость ко всем существующим атакам на хэш-функции, оптимизация под 64-разрядные процессоры и активное использование обращений к таблицам.

Skein поддерживает размеры внутреннего состояния 256, 512 и 1024 бит и размер выходного блока до $2^{64}-1$ бит. Авторы заявляют о 6.1 тактах на байт для любого размера выходного блока на ПК с процессором Intel Core 2

Duo. Из числа кандидатов на SHA-3 Skein входит в пятерку наиболее быстрых, однако лидирует лишь в 64-разрядном варианте, который превосходит по скоростным характеристика 32-разрядный в более чем четыре раза. Это объясняется тем, что авторы изначально ориентировались на оптимизацию под 64-разрядные процессоры. Skein-512 может быть реализована с использованием всего 200 байт состояния, Skein-256 — 100 байт, что оптимально для аппаратной реализации алгоритма в смарт-картах. Как заявляют авторы, хэш-функция Skein на распространённых процессорах работает в среднем два раза быстрее SHA-512, Threefish в два раза быстрее AES.Skein защищена от новых видов атак на хэш-функций — подбора удлинённых сообщений и псевдоколлизий. Threefish, лежащий в основе Skein имеет очень простую структуру и может быть использован для замены алгоритмов блочного шифрования, будучи быстрым и гибким шифром, работающим в произвольном режиме шифрования.

2.3 КОНСТРУКТИВНЫЕ ОСОБЕННОСТИ УНИВЕРСАЛЬНЫХ КРИПТОПРИМИТИВОВ KECCAK И SKEIN

В финале конкурса SHA-3 competition, два финалиста представляют собой универсальные криптопримитивы, которые могут использоваться не только для хэширования, но и для выполнения множества других криптографических операций. Это может привести к значительному упрощению криптографических протоколов, как уже существующих, так и вновь проектируемых. Традиционный дизайн хэш-функций основан на использовании функции сжатия. Эта функция отображает значение $m \rightarrow n$ $(m > n)$ псевдослучайным образом. При этом значение n должно быть до 512 бит, а m порядка 2n. Повторяя эту функцию в течении нескольких раундов с различными константами, достигают нужного значения стойкости. Дополняя и сцепляя между собой блоки от разных фрагментов исходного текста, получают возможность вычислить хэш от сообщения произвольной длины.

При этом возникает существенная проблема: сконструировать функцию сжатия трудно. Многораундовые повторения сгладят её дефектность, но заведомое наличие быстрой возможности найти частичную коллизию в исходной функции сжатия ставит под вопрос стойкость всей конструкции. Авторы алгоритма Keccak (Guido Bertoni, Joan Daemen, Michaël Peeters и Gilles Van Assche) утверждают, что сконструировать надёжную функцию сжатия вида m → n (m > n) как нораундовый блок криптопримитива, крайне сложно (или невозможно вообще).

2.3.1 Хэш-функция Skein основанная на блочном шифре Threefish

Хэш функция Skein представляет собой универсальный криптографический примитив, состоящий из блочного шифра Threefish,

(Рисунок 2.6) который используется в режиме UBI- хэширования (Unique Block Iteration).

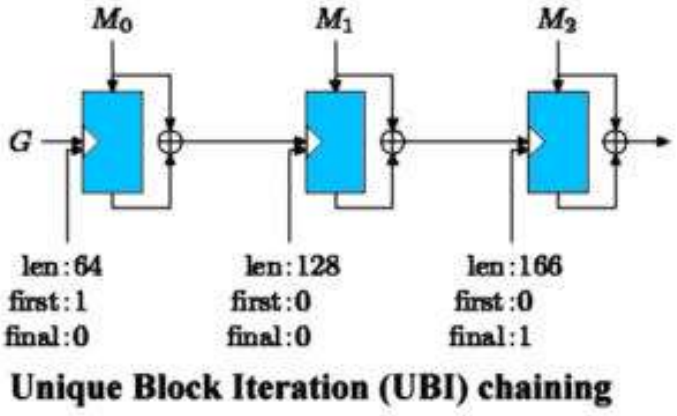


Рисунок 2.6 – UBI- хэширование (Unique Block Iteration)

Этот блочный шифр имеет простую структуру и может использоваться для замены обычных алгоритмов блочного шифрования в качестве нового быстрого и универсального шифра в любом распространённом режиме шифрования, кроме того разрабатывается версия Threefish со специальным режимом шифрования со встроенной аутентификацией. Авторы отказались от использования тяжеловесного дизайна (Рисунок 2.7), S-блоков и сложных операций и использовали простейшую комбинацию инструкций XOR-ADD-Rotation.

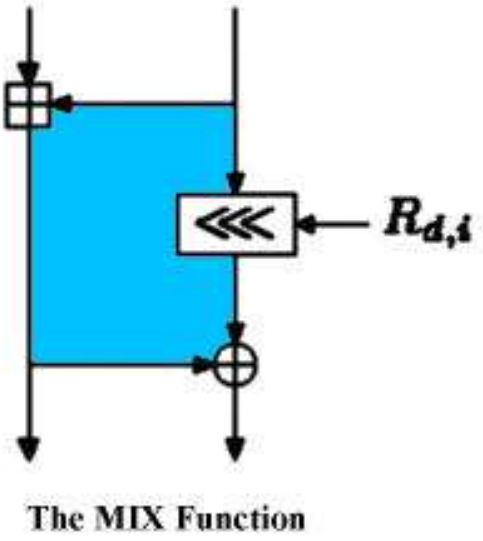


Рисунок 2.7 – XOR-ADD-Rotation

Эта конструкция повторяется в течении 72 раундов (Рисунок 2.8) при размере ключа и блока 256 и 512 бит или 80 раундов для размера ключа и блока 1024 бита.

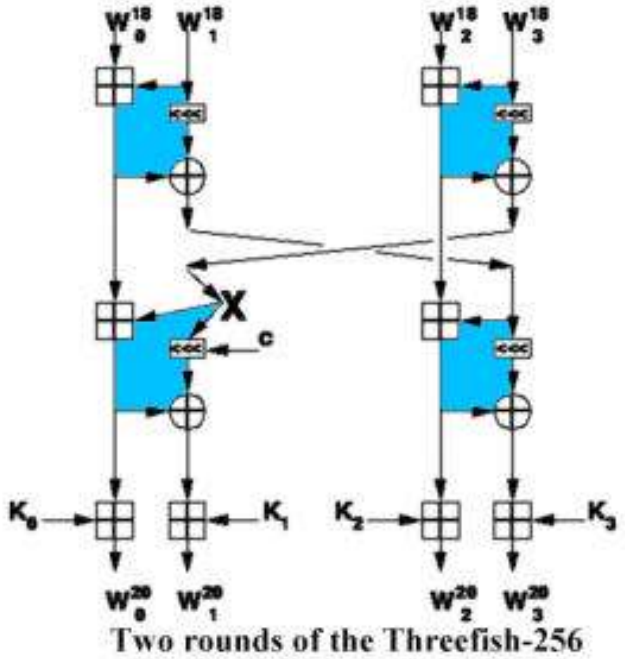


Рисунок 2.8 – Два раунда Threefish-256

Также используется перестановка слов между раундами, а нелинейность создаётся путём простого сложения ключа. Максимально необходимый ключ для использования внутри хэш-функции был-бы достаточен и 512-бит, но значение 1024 может быть использовано для защиты с запасом прочности в случае появления атак на шифр (Рисунок 2.9)

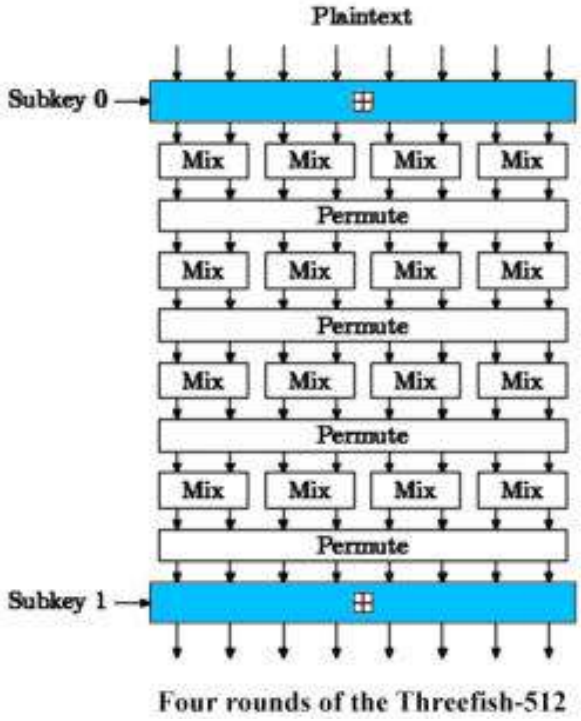


Рисунок 2.9 – Четыре раунда Threefish 512

Шифр представляет собой SP-сеть на обратимых операциях и не является шифром Фейстеля. Threefish поддерживает вход не только для ключа и открытого текста, но и tweak-значения (аналог вектора инициализации), что позволяет изменять значение выхода, не меняя значения ключа (что позволяет использовать новые режимы шифрования).

Функция хэширования Skein обладает широким набором свойств. При использовании различных переключаемых параметров она может использоваться не только как простая хэш-функция, но и как функция

древовидного хэширования (для ускоренной обработки больших объёмов данных), в качестве кода аутентификации сообщений (MAC), или в качестве составной части HMAC, есть поддержка хэширования, специальный режим для использования в цифровых подписях (хэш связывается не только с сообщением, но и с самим публичным ключом), в качестве функции вычисления производных ключей (KDF), в качестве функции вычисления ключа из пароля (PBKDF), в качестве генератора псевдослучайных чисел (PRNG) и в качестве потокового шифра (этот режим отличается от PRNG, так как не требует уничтожения данных о предыдущих использованных внутренних состояниях), режим персонализации (включения адресных форматированных полей для управления криптопреобразованиями).Кроме того Skein защищена от новых видов специфических атак на хэш-функций – подбор удлинённых сообщений, псевдоколлизии и др.Раньше обычную хэш-функцию нельзя было использовать напрямую для разных целей, но использование концепции "доказуемой безопасности" и редукционистских аргументов стойкости позволило авторам создать универсальный криптопримитив, укладывающийся в модель случайного оракула, а блочный шифр Threefish – в модель идеального блочного шифра. Вместо выбора разных схем и стандартов и изучения особеностей их применения, работы и реализации, разработчикам криптоприложений можно использовать Skein с различными параметрами.

Кроме стандартных размеров выхода 128, 160, 224, 256, 384 и 512 бит, Skein позволяет получить значение выхода любой длины (в пределах 2^{64}-1). Авторы делают стандартные заявления, что Threefish и Skein не содержат отмычек, так как все константы содержат 512-бит и если бы в них можно было спрятать какой-либо ключ, то это было бы революционным открытием в криптографии однако ничего не утверждается о возможных слабостях алгоритма перед неизвестными видами криптоанализа, никаких заявлений по поводу того, как такая простая структура может противостоять алгебраическому криптоанализу, кроме как из-за очень большого количества

раундов также сделано не было, впрочем подробных работ по криптоанализу также пока не представлено.

В целом хэш-функция Skein на наиболее распространённых процессорах работает в два раза быстрее, чем SHA-512, а блочный шифр Threefish в два раза быстрее, чем AES.

2.3.2 Хэш-функция Keccak основанная на конструкции Sponge

Хэш-функция Keccak не использует функцию сжатия в виде отдельного строительного блока, а в качестве стойкого криптопреобразования использована бесключевая PRP. Всё это составляет простую конструкцию Sponge ("Губка"). В фазе абсорбции ("поглощения", "впитывания" губкой) (Рисунок 2.10) сначала задаётся исходное состояние из нулевого вектора размером до 1600 бит. Затем производится операция xor фрагмента исходного сообщения p_0 с фрагментом исходного состояния размером r, оставшаяся часть состояния ёмкостью c остаётся незатронутой.

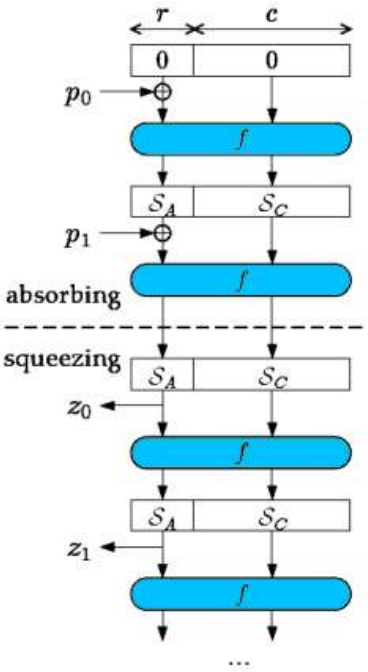


Рисунок 2.10 – Фаза абсорбции хэш-функции Keccak

Результат обрабатывается функцией f — многораудовой бесключевой PRP и так повторяется до исчерпания блоков исходного сообщения. Затем наступает фаза сжатия или отжатия (как если бы отжимали губку), при которой можно вывести хэш произвольной длины.

При этом сложность проведения атак гарантируется только до необходимой величины. После увеличения длины выхода хэш-функции за пределы внутреннего состояния, её стойкость перестаёт зависеть от размера выхода. Атаки на нахождение коллизий (двух произвольных сообщений, дающих одинаковый хэш) и вторых прообразов (сообщения, дающего такой же хэш, что и имеющееся) имеют важное практическое и теоретическое значение, но наряду с неинвертируемостью не обеспечивают полной оценки стойкости хэш-функции. Существует целый ряд экзотических атак на удлинение сообщения, атаки на частичные коллизии с подобранным префиксом и др. Чтобы доказать стойкость конструкции Sponge ко всем возможным атакам, авторы приняли ещё одно радикальное решение: считать единственным и достаточным общим критерием стойкости неразличимость хэш-функции от случайного значения. Случайный оракул (Random Oracle) — это идеализированная функция, описывающая работу машины с практически бесконечным объёмом памяти, которая на любой запрос может выдать идеально случайное число и запомнить пару "запрос-ответ". Если такой же запрос будет когда-либо повторён, то ответ будет не генерироваться заново, а взят из запомненного списка. Если перестановка f, лежащая в основе Sponge идеальна, то и хэш-функция доказуемо неразличима от RO, значит никакие из возможных атак действовать не будут. Конечно, есть оговорка на наличие универсального тривиального различителя для всех возможных хэш-функций: построение случайного оракула на алгоритме с конечным числом состояний невозможно. Например коллизия внутреннего состояния приведёт к неслучайному выходу, что было бы невозможно в RO, у которого никаких внутренних состояний нет. Однако, авторы доказывают, что такая

конструкция обеспечивает стойкость для N запросов к функции f или f^{-1}, равную $N^2 \cdot 2^{-(c+1)}$, что при обычной длине выхода укладывается в $2^{c/2}$ и невозможно без наличия различителя в используемой PRP.Именно эти теоретические результаты, основанные на одном из самых строгих доказательств неразличимости от случайного оракула в конкурсе SHA-3, дают удивительные практические возможности для простого использования хэш-функции Keccak в качестве практически универсального криптопримитива. Простое добавление секретного ключа на вход хэш-функции Keccak/Sponge превращает её в код аутентификации сообщений. (Рисунок 2.11)

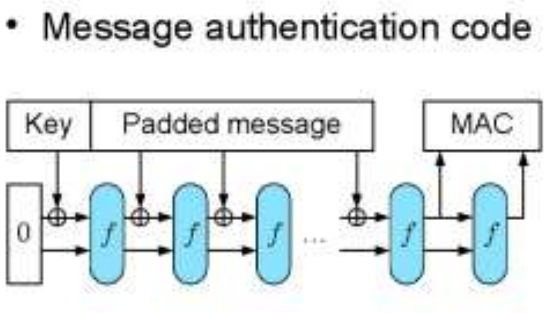


Рисунок 2.11 – Добавление секретного ключа

Это было невозможно в обычных хэш-функциях SHA-1 или SHA-2 и требовало громоздкой конструкции HMAC. Добавление секретного ключа с открытым вектором инициализации и вывод гаммы произвольной длины превращает конструкцию Sponge в потоковый шифр. Добавление на вход ключа (Рисунок 2.12), вектора инициализации и номера блока позволяет получить потоковый шифр с произвольным доступом — аналог блочного шифра в режиме счётчика (AES-CTR).

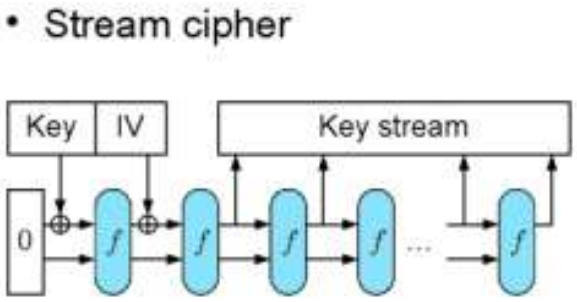


Рисунок 2.12 – Добавление секретного ключа с открытым вектором

Это может быть использовано совместно с MAC для шифрования сообщений или пакетов данных (Рисунок 2.13).

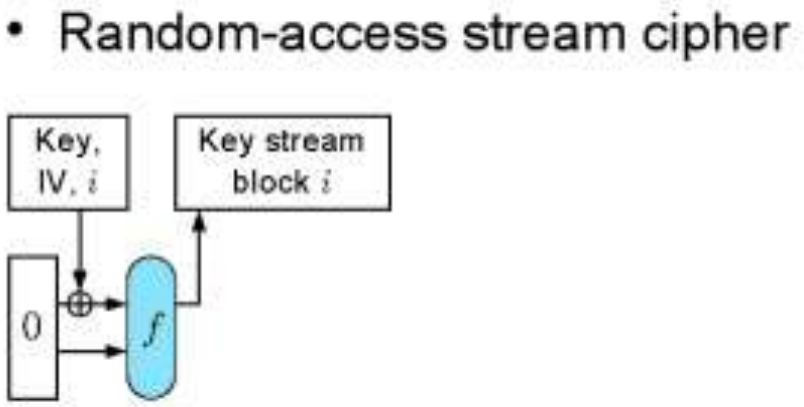


Рисунок 2.13 – Добавление на вход ключа, вектора инициализации и номера блока

Использование входа и выхода переменной длины может применяться для генерации симметричных ключей из паролей (Рисунок 2.14), из протоколов согласования ключа по асимметричным алгоритмам, для связывания сообщения с электронной подписью, рэндомизированного хэширования, для дистилляции случайных данных из шумов. Если периодически уничтожать использованные предыдущие состояния, то из такой конструкции легко построить генератор псевдослучайных чисел.

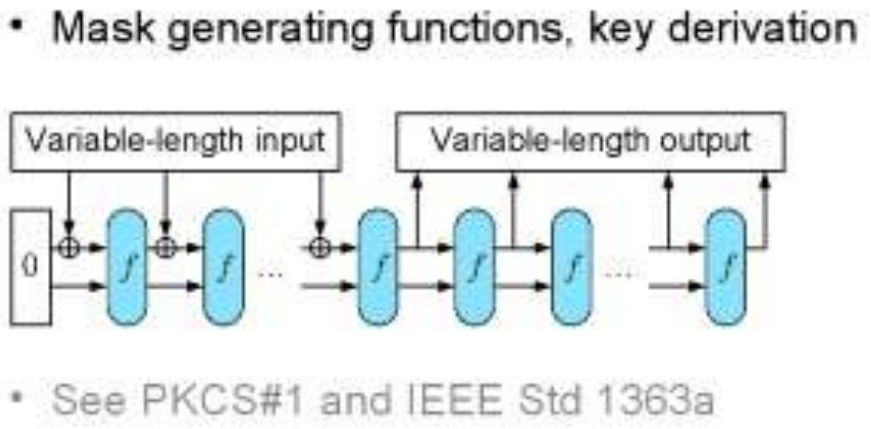


Рисунок 2.14 – Генерация симметричных ключей из паролей

Ещё одной интересной особенностью конструкции Sponge является лёгкость персонализации данных. Предположим, один и тот же пароль используется как для генерации симметричного ключа, так и для ключа MAC — для предотвращения изменения сообщения. Нежелательно, чтобы они были одинаковыми. Например, в хэш-функции Skein для этого предусмотрен ограниченный набор битов для переключения режимов использования. А в конструкции Sponge хэш-функции Keccak это не требуется. Любая персонализация данных может быть добавлена на вход функции, например, по рекомендации авторов, в XML-формате и кодировке UTF, что делает число возможных вариантов персонализации и режимов использования практически неограниченным.

Кроме конструкции Sponge, интересна также и сама функция перестановки хэш-функции Keccak. Текущее внутреннее состояние представлено в ней в виде набора битов, сгруппированных в виде виртуального объекта в трёхмерном пространстве (трёхмерного массива Рисунок 2.15).

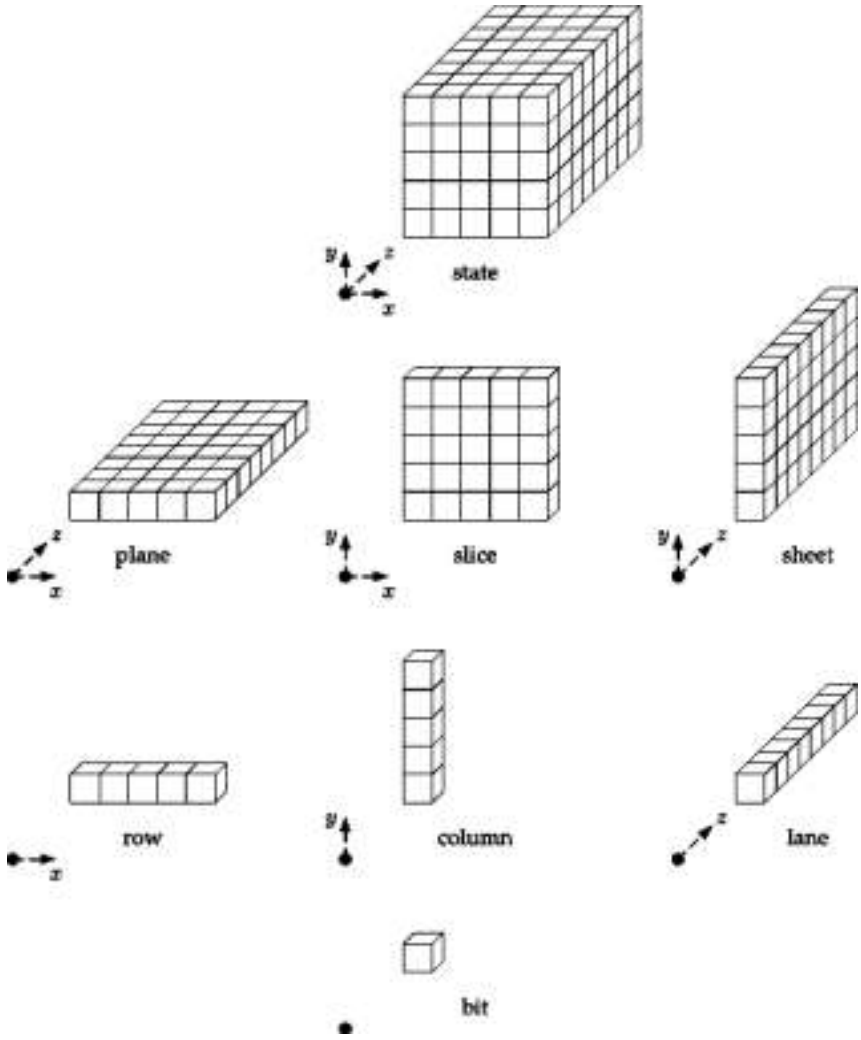


Рисунок 2.15 – Трёхмерное представление текущего внутреннего состояния

Сам объект можно разбить на плоскости или точнее слои вдоль трёх осей координат, а элементы каждого слоя — на фрагменты в виде столбцов или векторов. При этом для обработки внутренних состояний не используются S-блоки или операции, параметры которых менялись бы в зависимости от данных. Для создания нелинейных преобразований используется набор простейших логических операций (XOR, AND, NOT), которые существуют в любом процессоре и минимальный набор констант. А трёхмерное представление текущего внутреннего состояния помогает применить эти операции оптимальным образом. Функции Chi, Theta, Pi, Rho и Iota последовательно обрабатывают внутреннее состояние на каждом раунде.

2.3.3 Выводы по разделу

Хэш-функция Keccak является таким же универсальным криптопримитивом, как и Skein, за исключением возможности использования в режиме блочного шифра. Режим блочного шифра остаётся незаменимым в прозрачном шифровании данных на дисках компьютеров, но в этой области нет существенного ограничения по ресурсам, поэтому для шифрования можно использовать отдельный блочный шифр в соответствующем режиме. Для решения многих других задач Keccak также универсален, как и Skein, а фактически его область применения может быть шире. При необходимости этот алгоритм может быть внедрён как в миниатюрные устройства с ограниченными ресурсами, так и на высокопроизводительные серверы, работающие с большим объёмом соединений. Оба алгоритма имеют самую сильную доказательную базу среди всех финалистов, но по некоторым оценкам, Keccak имеет более строгие доказательства в модели RO. В частности он не подвержен атакам на функции конструкции Narrow Pipe.

Алгоритмы Keccak и Skein оказывают влияние на практическое использование симметричных алгоритмов и смешанных протоколов, а также и на теоретические исследования в различных областях криптографии. Skein не содержат отмычек, так как все константы содержат 512-бит и если бы в них можно было спрятать какой-либо ключ, то это было бы революционным открытием в криптографии (однако ничего не утверждается о возможных слабостях алгоритма перед неизвестными видами криптоанализа, никаких заявлений по поводу того, как такая простая структура может противостоять алгебраическому криптоанализу, кроме как из-за очень большого количества раундов также сделано не было, впрочем подробных работ по криптоанализу также пока не представлено).В целом хэш-функция Skein на наиболее распространённых процессорах работает в два раза быстрее, чем SHA-512, а блочный шифр Threefish в два раза быстрее, чем AES.

KeccaK обладает множеством качеств, включая элегантный дизайн и возможность эффективной реализации на большом количестве аппаратных платформ. Ясная конструкция KeccaK облегчает проведение его анализа. Алгоритм удачно показывает себя в специализированном исполнении на аппаратных платформах, превосходя в этом плане всех других финалистов и алгоритмы SHA-2.

KeccaK имеет преимущество в том, что атаки, рассчитанные против SHA-2 не действуют против него, поскольку эти два алгоритма устроены на совершенно разных принципах.

Новые полезные свойства Keccak найдут применение спустя годы, после его принятия. Так, он может быть использован в миниатюрных встраиваемых устройствах, которые не являются полноценными устройствами.

Дается возможность использовать KeccaK в качестве хэш-функции с произвольным размером выхода, в качестве потокового шифра, в качестве функции выработки ключей из пароля, в качестве кодов аутентификации сообщений, в качестве криптостойкого генератора псевдослучайных чисел с подкачкой энтропии из внешнего источника и с затиранием внутреннего состояния, в качестве возможного нового режима дополнения цифровых подписей.

Так же был представлен дуплексный режим работы KeccaK — потоковое шифрование с аутентификацией за один проход, что может представлять альтернативу использовавшимся ранее блочным шифрам в шифровании пакетной связи.

Все эти универсальные возможности не являются отдельными нагромождаемыми модулями, как это пытались реализовать в других алгоритмах. Смена режимов использования также не требует каких-либо переключателей предопределённого формата и никак не ухудшает простоту конструкции.

2.4 РЕЗУЛЬТАТЫ БАЗОВЫХ СТАТИСТИЧЕСКИХ ТЕСТОВ АЛГОРИТМОВ ХЭШИРОВАНИЯ SHA-3

Тестирование проводиться на MacOS X, с использованием библиотеки OpenSSL. Функции SHA-3 реализованы как категории. Механизм категории – эффективный способ расширить класс Objective C. С категориями мы можем добавить новые методы к классу, так же как переопределить существующие методы. Методы созданы отдельно и присоединены к классу во временя выполнения. В отличие от протоколов, категории могут быть реализованы, не разделяя целевой класс на подклассы. В отличие от разделения на подклассы, категории не нуждаются в предварительных знаниях исходного кода класса.

Тестирование проводилось на официальных спецификациях и на анализе, сделанном NIST. Далее представлены результаты тестов сравнения SHA -3, а так же алгоритмов MD 5 и SHA-1 и SHA-2.

Наши тесты используют хэш-функции OpenSSL в качестве сравнения. За исключением MD 5 и SHA 1, все тесты включают 256-разрядные значения хэширования. SHA 3 кандидаты основаны на коде, предоставленном их авторами. Код не оптимизирован, только изменен, чтобы решить проблемы пространства имен.

Тесты были сделаны на MacBook 2,4 ГГц Intel Core 2 Duo и MacOS X 10.6.4. Сообщением, используемом для испытания является фраза "Большая коричневая лиса прыгает через ленивую собаку"." The big brown fox jumps over the lazy dog."

2.4.1 Тест времени алгоритмов SHA-3

Тест показывает затраченное время на обработку одного байта сообщения мС/байт, и отклонение от заданных параметров. SHA-2 показал третье наилучшее время, только немного медленнее чем SHA-1 на 1.5 микросекунды. Из пяти кандидатов SHA-3 BLAKE был самым быстрым, затрачивая почти на 4 микросекунды больше чем SHA-2, чтобы произвести 256-хэширование. Skein второй, приблизительно в три раза медленнее чем BLAKE. Самым медленным из группы был JH. Потребовалась почти миллисекунда на байт сообщения, чтобы произвести хэширование.

Таблица 4.1 показывает работу хэш-алгоритмов SHA-3 и OpenSSL с точки зрения времени. MD5 и SHA-1 показали наилучшее время. Это происходит из-за подпрограмм, являющихся библиотечными функциями, таким образом оптимизированными для использования. Примите во внимание, что функции не выводят результат 256-разрядного хэширования.

Таблица 2.1 – Тест времени алгоритмов SHA-3

Алгоритм хэширования	Затраченное время (*мс*/byte)	Отклонение (+/-)
MD-5	3.361777	0,13194
SHA-1	6.42435	0,55307
SHA-2	7.87993	0,31271
Blake	10.44315	0,89484
Grøstl	524.99692	3,10271
JH	991.50034	2,39611
Keccak	145.29962	0,63526
Skein	33.60781	0,56023

2.4.2 Каскадный тест хэш-алгоритмов

Этот тест определят хэш алгоритм с хорошими лавинными свойствами

Если два данных о сообщении отличаются единственным байтом, их соответствующие перемешивании должны измениться на, половину полных байтов, но не больше, чем двух третей тех тех же самых байтов. В идеальном 256-битном хэшировании, которое показывает изменение по крайней мере на 16 бит за сообщение и не больше, чем 21 бит. Что-либо меньше или больше может показывать восприимчивость к атакам.

Таблица 4.2 показывает результаты испытаний и для SHA 3 и для OpenSSL. Первые два хэширования OpenSSL показали приличные лавинные свойства. При изменений MD 5 на один бит хэширования, выход изменялся на 9 бит за сообщение, на один байт больше чем минимальный идеал 8 (для 128-разрядного хэширования). При изменении SHA-2 -29 бит за сообщение. Это больше чем идеальный максимум 21 бит. SHA-3 все показали среднее изменение 17 бит за сообщение, больше чем минимум 16.

Таблица 2.2 – Результаты каскадного теста

Алгоритм хэширования	Количество измененных бит
MD-5	9
SHA-1	18
SHA-2	29
Blake	17
Grøstl	17
JH	17
Keccak	17
Skein	17

2.4.3 Тест на однородность выходных бит

Хэш-функция должна распределить свои выходные биты однородно. Это означает, что у первой половины вывода хэширования должно быть то же самое число 1s и 0s как и во второй половине (Таблица 4.3). Таким образом для идеального 256-разрядного хэширования, каждая из его половин должна иметь точно 64 1s и 0s. Слишком много 1s или 0s на одной половине хэширования подразумевают, что хэширование смещено. Это признак плохой статистической случайности, делатn функцию открытой для атак.

В Таблице 4.3 приведены результаты разрядного теста. Все алгоритмы хэширования показали небольшой скос в разрядном распределении, но скос никогда не превышает 12 бит. MD 5 отклонился на два бита. И у обеих половин его вывода хэширования среднее число 1s. SHA 2 немного более интересен. Обе его половины равняются двум 1s.

Таблица 2.3 – Частотный побитовый тест

Алгоритм хэширования	Биты (левая/правая)
MD-5	30/30
SHA-1	38/38
SHA-2	63/62
Blake	70/61
Grøstl	62/64
JH	67/64
Keccak	69/60
Skein	62/57

Все пять SHA 3 алгоритмов отклоняются от идеального распределения 64 1s и 0s. У BLAKE самое большое отклонение на левую половину на шесть 1s больше чем идеал. У Keccak самое большое отклонение правой половины, Grøstl и JH показали наименьшее отклонение.

2.2.4 Тест коллизии

Коллизией хэш-функции H называется два различных входных блока данных *x* и *y* таких, что *H(y)=H(x).* Коллизии существуют для большинства

хэш-функций, но для «хороших» хэш-функций частота их возникновения близка к теоретическому минимуму (Таблица 4.4). В некоторых частных случаях, когда множество различных входных данных конечно, можно задать инъективную хэш-функцию, по определению не имеющую коллизий. Однако для хэш-функций, принимающих вход переменной длины и возвращающих хэширование постоянной длины (таких как MD5), коллизии обязаны существовать, поскольку хотя бы для одного значения хэш-функции соответствующее ему множество входных данных (полный прообраз) будет бесконечно — и любые два набора данных из этого множества образуют коллизию.Если два сообщения отличаются байтом, функция должна присвоить разные значения для каждого уникального хэширования. Числа в скобках указывают на число хэш-функций на коллизию.

Таблица 2.4 – Результаты теста коллизии

Алгоритм хэширования	Коллизии(сообщений/ возникновение)
MD-5	0
SHA-1	0
SHA-2	0
Blake	0(3556)
Grøstl	0(2667)
JH	0
Keccak	0
Skein	0

В используемом сообщении 41 символ (принимающий кодирование ASCII). Подпрограмма производит 10,668 уникальных изменений заданного сообщения. Но число изменений слишком маленькое, чтобы вызвать коллизии. Чтобы должным образом протестировать 256-разрядную хэш-функцию, мы нуждаемся по крайней мере в 65,536 изменений. Чтобы достигнуть этого надо изменить тестовую задачу так, чтобы это изменило сообщение на разрядном уровне.

Алгоритмы BLAKE и Grøstl производили коллизии. У BLAKE среднее число 3556 уникальных хэш-функций на коллизию, Grøstl приблизительно 2667 уникальных хэш-функций на коллизию. Эти коллизии, не могли быть определены достоверно Данные результаты подразумевают нестабильность в тестовой задаче. SHA-3 алгоритмы показали не плохие показатели устойчивости производительности и безопасности. Однако в связи с недостаточной оптимизаций данные алгоритмы уступают более ранним версиям построенным на встроенных библиотеках.

2.4.5 Выводы по разделу

Алгоритм BLAKE оказался наиболее производительным, затрачивая почти на 4 микросекунды больше чем SHA-2, чтобы произвести хэширование. Skein показал второй результат, что приблизительно в три раза медленнее чем BLAKE. Самым медленным из группы был JH. Потребовалась почти миллисекунда на байт сообщения, чтобы произвести хэширование. По итогам побитового теста все пять SHA 3 алгоритмов отклоняются от идеального распределения 64 1s и 0s. У BLAKE самое большое отклонение на левую половину на шесть 1s больше чем идеал. У Keccak самое большое отклонение правой половины, Grøstl и JH показали наименьшее отклонение.

В ходе тестирования алгоритмы BLAKE и Grøstl показали возможность возникновения коллизии. У BLAK – 3556 уникальных хэш-функций 3556 на коллизию, у алгоритма Grøstl приблизительно 2667 уникальных хэш-функций на коллизию.

Алгоритм JH-показал средние значения во всех тестах но потребовалась почти миллисекунда на бит сообщения, чтобы произвести хэширование. JH по результатам теста времени оказался на последнем месте.

В тесте на лавинные свойства кандидаты SHA-3 показали среднее изменение сообщения 17 бит, немного больше чем минимум 16.

Частотный побитовый тест выявил что у алгоритма BLAKE самое большое отклонение на левую половину на шесть 1s больше чем идеал. У Keccak самое большое отклонение правой половины, Grøstl и JH показали наименьшие отклонения.

Алгоритм показал средние результаты Keccak по всем тестам. SHA-3 хэш-алгоритмы проявили не плохие показатели устойчивости, производительности и безопасности. Однако в связи с недостаточной оптимизацией данные алгоритмы уступают более ранним версиям построенным на встроенных библиотеках.

2.5 ОСОБЕННОСТИ АППАРАТНОЙ РЕАЛИЗАЦИЯ АЛГОРИТМОВ ХЭШИРОВАНИЯ

2.5.1 Структурные схемы алгоритмов хэширования для аппаратной реализации

Функция Blake основана на поточном шифре ChaCha. Состоящем из двух блоков G функций параллельно вычисляемыми с четырьмя G функциями на каждом уровне. G функция использует операции модульного сложения, неравенства и вращательные сдвиги.

Как показано на рисунке 2.15, данные входного сигнала сначала проходят через процесс перестановки, и затем они перенаправляются к первому блоку G функций наряду с сохраненными внутренними состояниями. Так как один блок использует два уровня G функции, каждый цикл обрабатывает половину блока. Вывод половины функции будет сохранен в регистрах внутреннего состояния для следующего цикла. После завершения последовательности блоков для текущего блока начнется процесс завершения, и значение объединения в блок для следующего сообщения входного сигнала сохранится в регистрах внутреннего состояния. В данном проекте, блок G функций может быть развернут в два уровня, чтобы уменьшить задержку за счет максимальной частоты и площади. Grøstl реализует расширенный вариант конструкция Merkle-Damgard Расширенный вариант конструкции Merkle-Damgard подобен простой конструкции Merkle-Damgard, но размер регистров внутреннего состояния расширенного варианта больше чем размер выходного сообщения. У Grøstl, есть две функции, P и Q, которые напоминают блочный шифр Rijndael. Обе функции включают четыре цикла преобразования, который, выполняет операции неравенства, перестановки и замены. Затем выводы P и Q будут использоваться в качестве вводов логических элементов неравенства и выводы которого будут сохранены как сообщение хэширования.

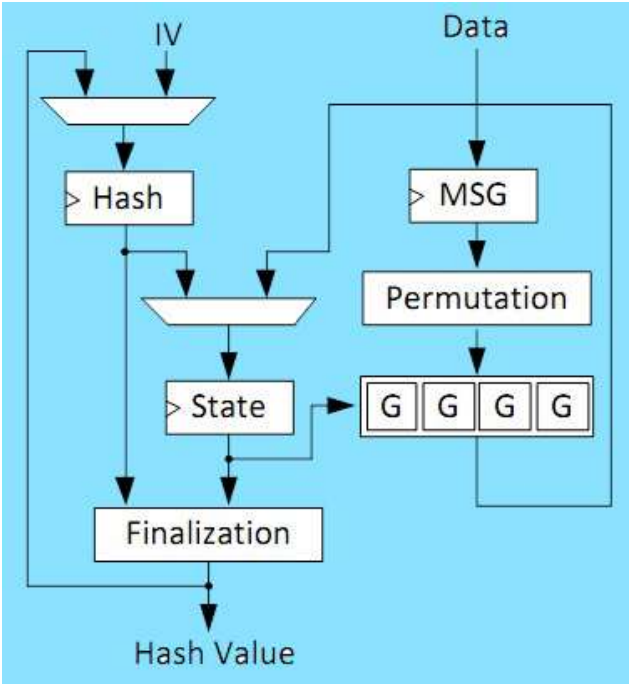


Рисунок 2.16 – Строение ядра алгоритма Blake-256

Входной сигнал и внутреннее состояние, такое же как у конструкции Merkle-Damgard.

Как показано на рисунке 5.2 функции реализации P и Q реализованы параллельно. Однако, вследствие их структурного подобия, их аппаратные ресурсы могут быть использованы совместно. Реализация Grøstl может быть построена, комбинируя функции P и Q, чтобы сохранить аппаратные ресурсы, которые улучшат производительность на FPGA.

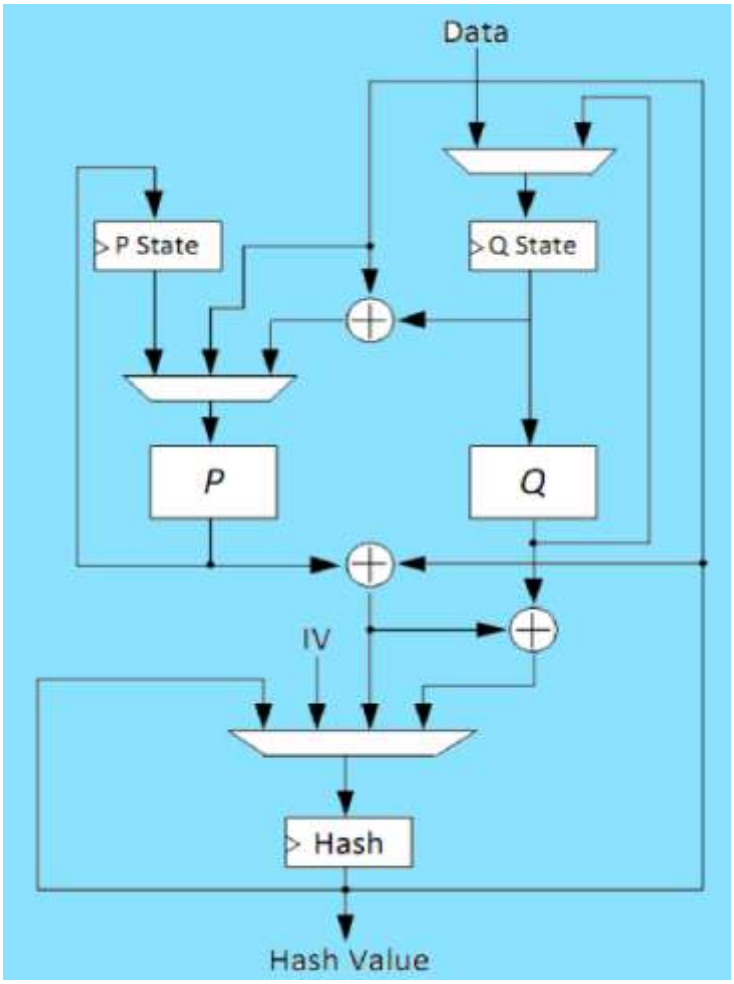


Рисунок 2.17 – Строение ядра Grøstl-256

Функция сжатия JH состоит из перестановки, замены и логического неравенства. Перед каждым циклом первый блок значений объединяется в блок – XOR с входным сигналом. Затем в конце каждого цикла, вторая половина значений добавляется в блок – XOR с входным сигналом.

Как показано на рисунке 5.3, основная логика оценки JH – модуль R6, R6 – образующий цикл. В каждом цикле основанные константы сгенерированы или на цикловых константах от предыдущего цикла или на начальных данных. Модуль генерирует новые значения объединения в блок, основанные на входном сигнале, объединяя значения в блок от предыдущего цикла, и генерирует константы. Уровень замены с блоками S, уровень

линейного преобразования и уровень перестановки представлены, внутри цикла.

Доступны тонкие настройки для алгоритма JH. Сначала S блоки могут быть реализованы вместо логических функций для таблиц поиска, R6, может быть заменена ROM, так как все возможные константы могут быть вычислены заранее.

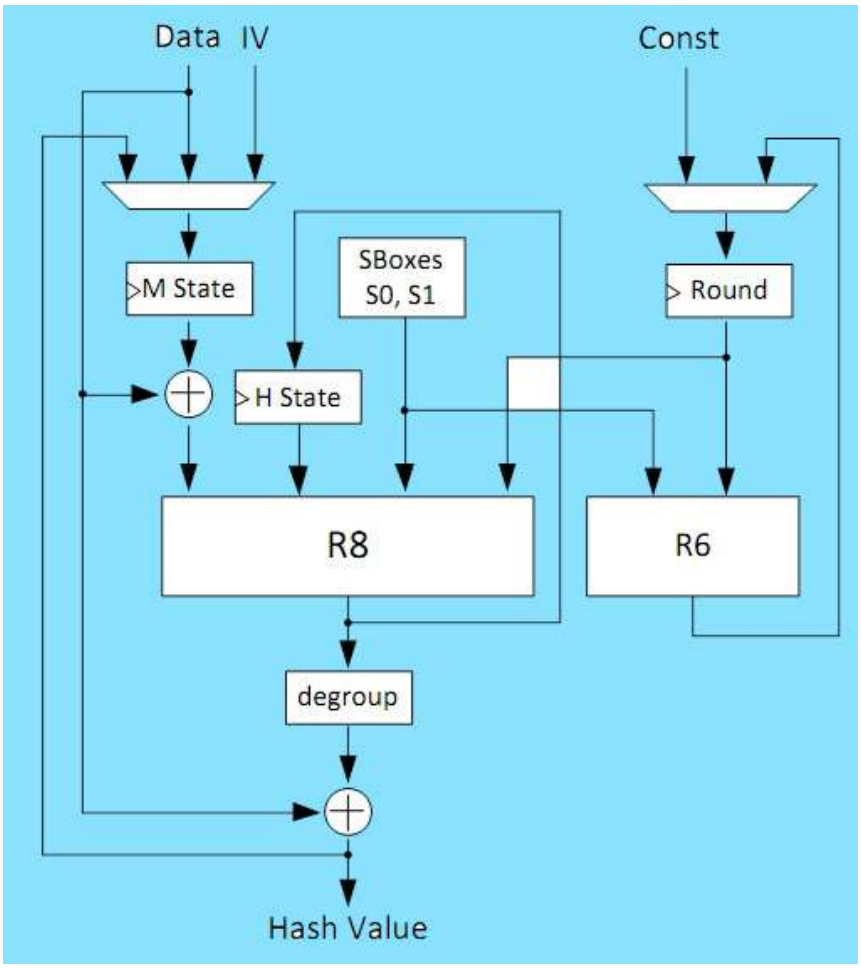


Рисунок 2.18 – Строение ядра хэш-функции JH-256

Keccak использует модель Sponge. В конструкции Sponge есть две фазы, впитывание и выжимание. Во время фазы поглощения входной сигнал будет XOR с данными, накопленными в первом блоке. Значение будет обновлено наряду с данными от второго блока данных внутреннего состояния через цикл. Во время фазы отжима данные, накопленные в первом блоке, будут использоваться в качестве блока вывода. Новые выходные

значения будут присутствовать регистрах, поскольку они обновляются в каждом цикле. В отличие от схем, упомянутых ранее, Sponge может генерировать последовательность любого произвольного размера. Рисунок 5.4 показывает общую схему ядра Keccak-256.

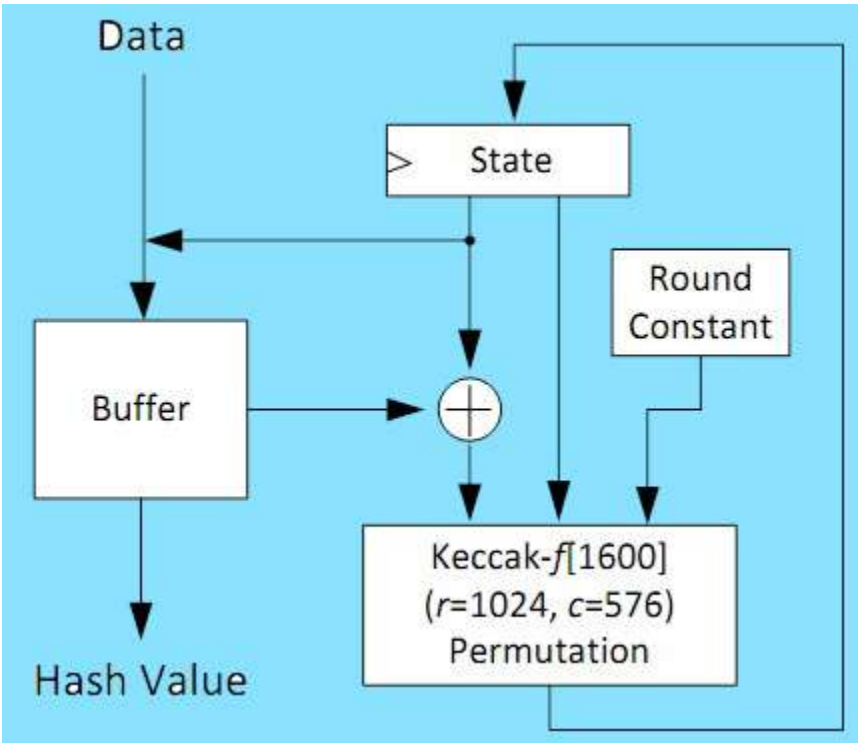


Рисунок 2.19 – Строение ядра Keccak-256

Реализация Skein 512-256 показана на рисунке 2.19. Цикл алгоритма представляет собой последовательность операций: параллельное применение операций *mix*() к каждым двум соседним блокам; перестановка бит блока. Кроме того, перед первым циклом и после каждых четырех циклов выполняется наложение соответствующего фрагмента расширенного ключа шифрования операцией сложения по модулю 2^n, где n – размер состояния в битах. Threefish выполняет 72 раунда для 256- или 512-битного блока.

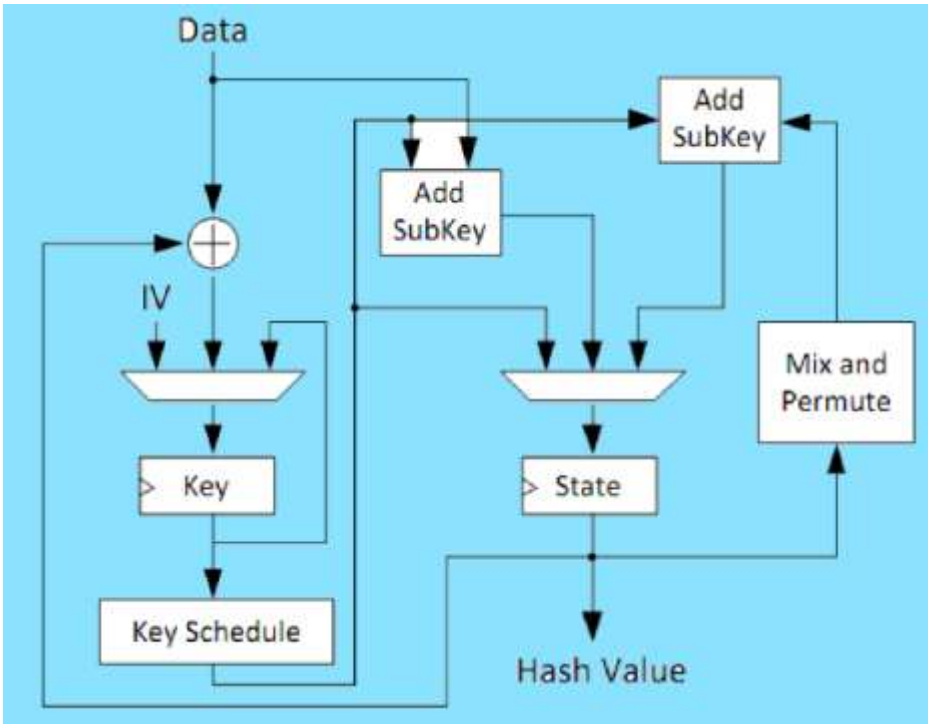


Рисунок 2.20 – Строение ядра Skein 256

2.5.2 Описание аппаратных средств реализации алгоритмов хэширования

ASIC—интегральная схема, специализированная для решения конкретной задачи. В отличие от интегральных схем общего назначения, специализированные интегральные схемы применяются в конкретном устройстве и выполняют строго ограниченные функции, характерные только для данного устройства; вследствие этого выполнение функций происходит быстрее и, в конечном счёте, дешевле. Примером ASIC может являться микросхема, разработанная исключительно для управления мобильным телефоном, микросхемы аппаратного кодирования/декодирования аудио- и видео-сигналов (сигнальные процессоры). Микросхема ASIC имеет узкий круг применения, обусловленный жёстко предопределённым набором её функций. Современные ASIC часто содержат полноценный 32-битный процессор, блоки памяти (как ПЗУ, так и ОЗУ) и другие крупные блоки. Такие ASIC часто называют система на кристалле (англ. System-on-a-Chip).

При разработке цифровых ASIC для описания их функциональности используют языки описания аппаратных устройств (HDL), такие как Verilog и VHDL.

FPGA—полупроводниковое устройство, которое может быть сконфигурировано производителем или разработчиком после изготовления; отсюда название: «программируемая пользователем». ППВМ программируются путём изменения логики работы принципиальной схемы, например, с помощью исходного кода на языке проектирования (типа VHDL), на котором можно описать эту логику работы микросхемы. ППВМ является одной из архитектурных разновидностей программируемых логических интегральных схем (ПЛИС). ПВМ включают в себя три главных программируемых элемента: нескоммутированные программируемые логические блоки (ПЛБ), блоки ввода-вывода (БВВ) и внутренние связи. ПЛБ являются функциональными элементами для построения логики пользователя, БВВ обеспечивают связь между контактами корпуса и внутренними сигнальными линиями. Программируемые ресурсы внутренних связей обеспечивают управление путями соединения входов и выходов ПЛБ и блоков ввода-вывода на соответствующие сети [13]. Все каналы трассировки имеют одинаковую ширину (одинаковое количество проводников). Большинство блоков БВВ вписываются либо в одну строку (по высоте), либо в один столбец (по ширине) массива вентилей.

ППВМ могут быть модифицированы практически в любой момент в процессе их использования. Они состоят из конфигурируемых логических блоков, подобных переключателям с множеством входов и одним выходом (логические вентили или gates). В цифровых схемах такие переключатели реализуют базовые двоичные операции AND, NAND, OR, NOR и XOR. В большинстве современных микропроцессоров функции логических блоков фиксированы и не могут модифицироваться. Принципиальное отличие ППВМ состоит в том, что и функции блоков, и конфигурация соединений между ними могут меняться с помощью специальных сигналов, посылаемых

схеме. В некоторых специализированных интегральных схемах (ASIC) используются логические матрицы, аналогичные ППВМ по структуре, однако они конфигурируются один раз в процессе производства, в то время как ППВМ могут постоянно перепрограммироваться и менять топологию соединений в процессе использования. Однако, такая гибкость требует существенного увеличения количества транзисторов микросхемы.

Процесс проектирования микросхемы ASIC больше громоздкий чем процесс проектирования FPGA. В большинстве случаев весь процесс проектирования FPGA сводиться к мат-обеспечению.

Чтобы сравнить различные конструкции, требуются метрики для оценки. Кандидаты характеризуется следующими пятью основными показателями.

Площадь: размер конструкции. Площадь конструкции на FPGA и ASIC вычисляется по количеству логических ячеек которые он занимает. Площадь логических ячеек является одним из наиболее важных показателем ресурсо-стоимости конструкции.

Максимальная частота: параметр быстродействия микрочипа. Максимальная пропускная способность: пропускная способность конструкции вычисляется следующим образом.

Размер блока умножить на частоту и разделить на задержку

$$TP=(BlockSize*Maxfreq)/Latency. \quad (5.1)$$

Размер блока – объем данных, который алгоритм хэширования обработает за один раз. Параметр фиксируется алгоритмом и не является параметром для аппаратного разработчика. Задержка, показанная здесь, является базовой задержкой, которая является числом циклов, которые берутся чтобы хэшировать сообщение.

Максимальная пропускная способность важный параметр, указывает сколько данных проект может обработать в течении фиксированного

количества времени. Из данных максимальной пропускной способности можно узнать размер блока, характеристику алгоритма; максимальную частоту, характеристику аппаратных проектных показателей и задержку.

Минимальная площадь-проект минимальной области минимизирует использование логических ресурсов (GE) за счет производительности.

Максимальная скорость-проект максимальной скорости минимизирует вычислительную задержку проекта, за счет области

Питание: питание проекта состоит из активного питания и статического питания. Активное питание проекта требуется на выполнение алгоритма. Статическое используется для питание схемы.

Энергия: энергия затраченная на реализацию проекта выражена как энергия деленная на бит входного сообщения.

Данные метрики используются для представления аппаратных результатов производительности алгоритмов хэширования. Основная задача оптимизации алгоритмов увеличить пропускную способность по отношению к площади, и вычислить максимальную пропускную способность.

2.5.3 Выбор оптимальной технологии для сравнения алгоритмов

Аппаратная производительность SHA-3 алгоритмов в FPGA, ASIC использует различные технологии для выполнения функций. Большинство алгоритмов реализуется на нескольких видах устройств (Рисунок 5.6).

При использовании технологии FPGA: FPGA на 65 нм- 56%, FPGA на 90 нм – 34%, FPGA на 130 нм-10%.

Показатели для ASIC: ASIC на180 нм- 48%, ASIC на 130 нм. -33% ASIC на 90 нм- 10%

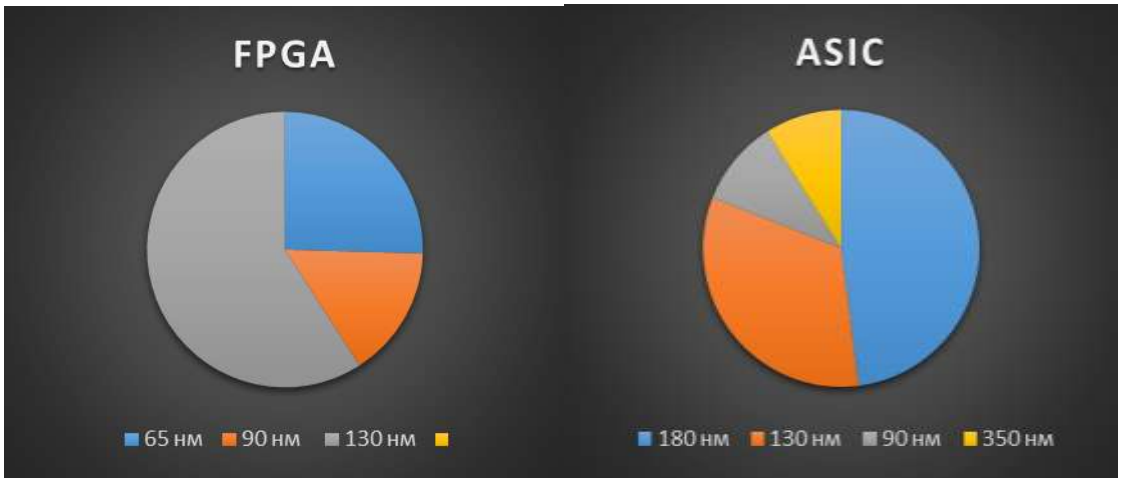


Рисунок 2.21 – Архитектуры процессоров ASIC и FPGA

У различных семейств FPGA может быть решительно различная архитектура. Например, портируя из Xilinx Spartan3E к Xilinx Virtex-5, изменялся размер элемента от 90 нм до 65 нм, но основной логический элемент также изменяется от с 4 LUT до с 6 LUT.

Для технологии ASIC коэффициент изменения будет более линейным.

Для аппаратной оценки результатов деятельности на FPGA все алгоритмы реализованы на Xilinx 3, Virtex 4, Virtex 5 и Virtex 6.

Реализация базовой функциональности включает логику ядра хэш-функции, такой как функция сжатия. Что дает быстрые но грубые оценки относительно производительности проекта. Рисунок 5.7 показывает различные варианты исследования: а) показывает полностью автономную стратегию; б) показывает проект с внешней памятью; в) показывает проект с базовой функциональностью.

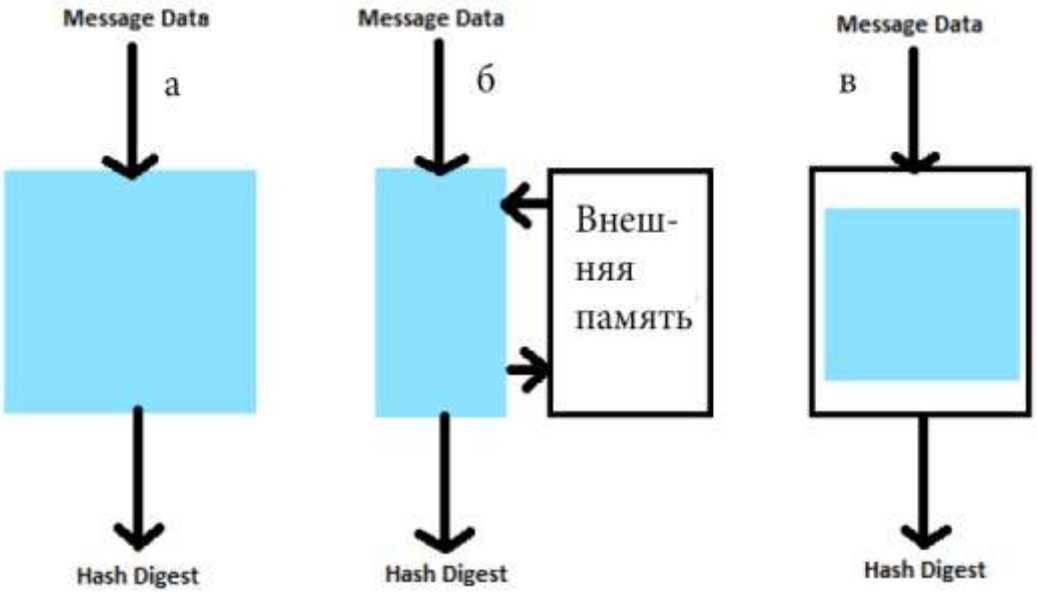


Рисунок 2.22 – а) показывает полностью автономную стратегию; б) показывает проект с внешней памятью; в) показывает проект с базовой функциональностью

Полные структуры платформ оценки для FPGA и устройств ASIC показываются на рисунке 5.8. Можно заметить, что они очень подобны. Платформы оценки содержат PC, плату SASEBO и осциллограф.

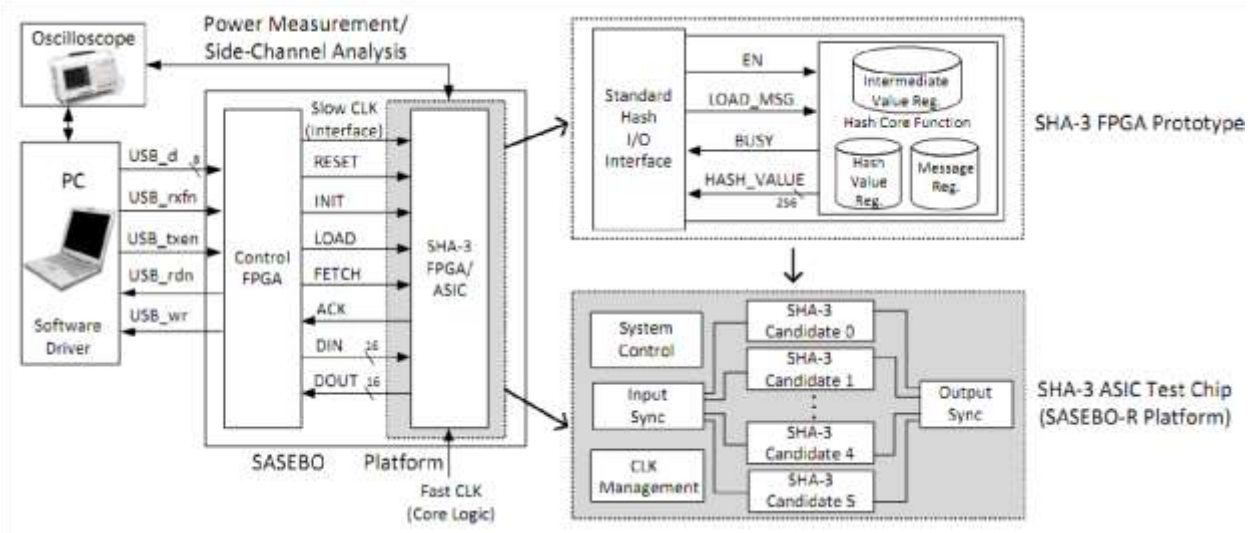


Рисунок 2.23 – Общая схема анализа прототипа FPGA и тестирования ASIC

PC используется для проверки, реализации программного обеспечения алгоритмов SHA-3. PC отправляет сообщения криптографическим аппаратным средствам, которые содержат аппаратные реализации SHA-3 алгоритмов. Когда криптографические аппаратные средства заканчивают хэшировать и возвращают результат хэширования назад на PC, PC сравнивает результаты между аппаратными средствами и программным обеспечением, чтобы проверить правильность.

Платы SASEBO выбраны, для криптографической аппаратной оценки. Для анализа прототипа FPGA используется SASEBO-GII. SASEBO-GII использует 2 блока FPGA, которые используется в качестве интерфейса между PC и аппаратной реализацией SHA-3 алгоритма. SHA 3 алгоритмы будет реализован на Virtex 5. Для анализа прототипа ASIC используется SASEBO-R. Общая схема технической реализации алгоритмов SHA-3 на микросхеме ASIC.

2.5.4 Аппаратные результаты оценки ASIC

Результаты работы алгоритмов на платформе ASIC представлены на рисунке 5.9 основными параметрами сравнения выступали: отношение пропускной способности к площади, пропускная способность относительно затраченной энергии. Алгоритмы которые показывают наилучшее соотношение, поскольку они занимают не большой объем при высокой скорости выполнения, в то время как кандидаты с наибольшей разницей являются большими по объему и медленными.

Кандидат Keccak показывает наилучшее соотношение с точки зрения пропускной способности на занимаемую площадь. Данный показатель в три раза лучше, чем показатель хэш-алгоритма Grøstl. Keccak занимает не большой объем и при этом показатели скорости и максимально достижимой частоты находиться на высоте. Кроме того, размер блока функции Keccak позволяет максимально увеличить пропускную способность. В совокупности это позволяет Keccak, превзойти других кандидатов с большим отрывом в ASIC реализаций.

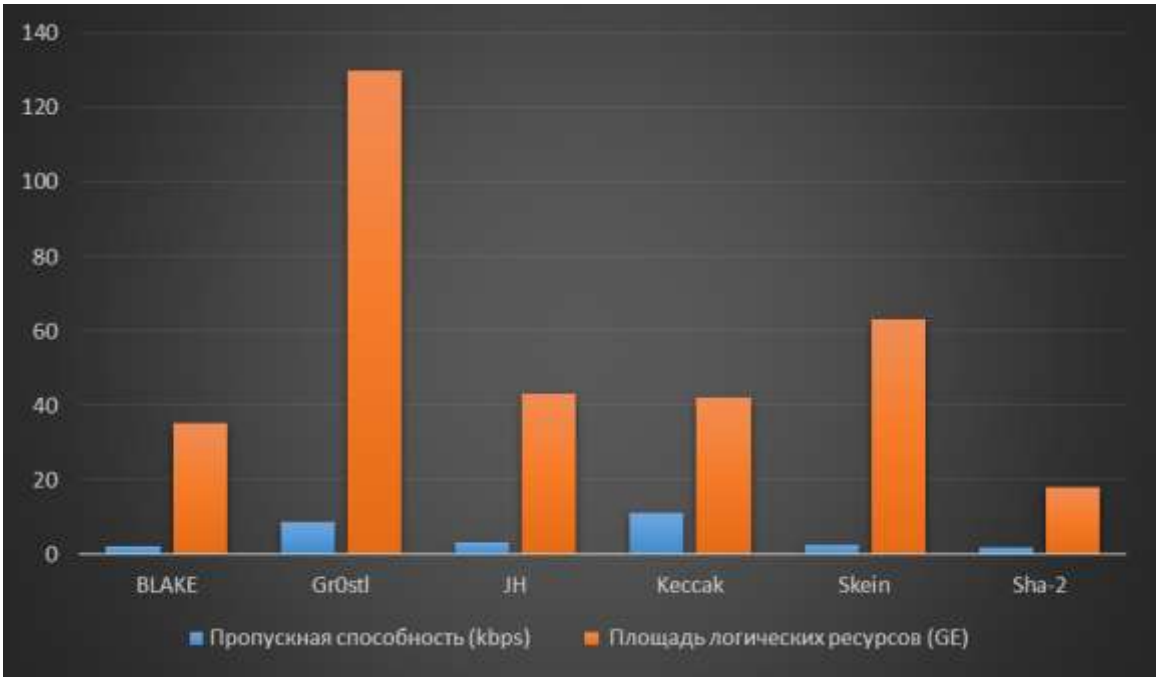


Рисунок 2.24 – Пропускная способность по отношению к площади для реализаций ASIC алгоритмов SHA-3

Кандидат Grøstl второй. Grøstl – крупнейший кандидат всего конкурса SHA-3; алгоритм Grøstl почти в дважды больше, чем следующий алгоритм Skein. Однако, у Grøstl самая короткая задержка, сравнимая с другими кандидатами. Это восполняет недостаток размера и посредственную скорость Grøstl и позволяет хэш-алгоритму выделяться с точки зрения отношения пропускной способности к площади (Рисунок 5.10).

Кандидат Skein работает не оптимально на реализациях ASIC. Это плохо и с точки зрения области применения и достижимой скорости. Можно заметить, что максимальная пропускная способность Skein ниже чем JH будучи больше в площади реализации. Несмотря на относительно низкую задержку Skein, показал не лучшее отношение пропускной способности к области. Кандидаты Keccak и Grøstl, имеют более высокие показатели пропускной способности.

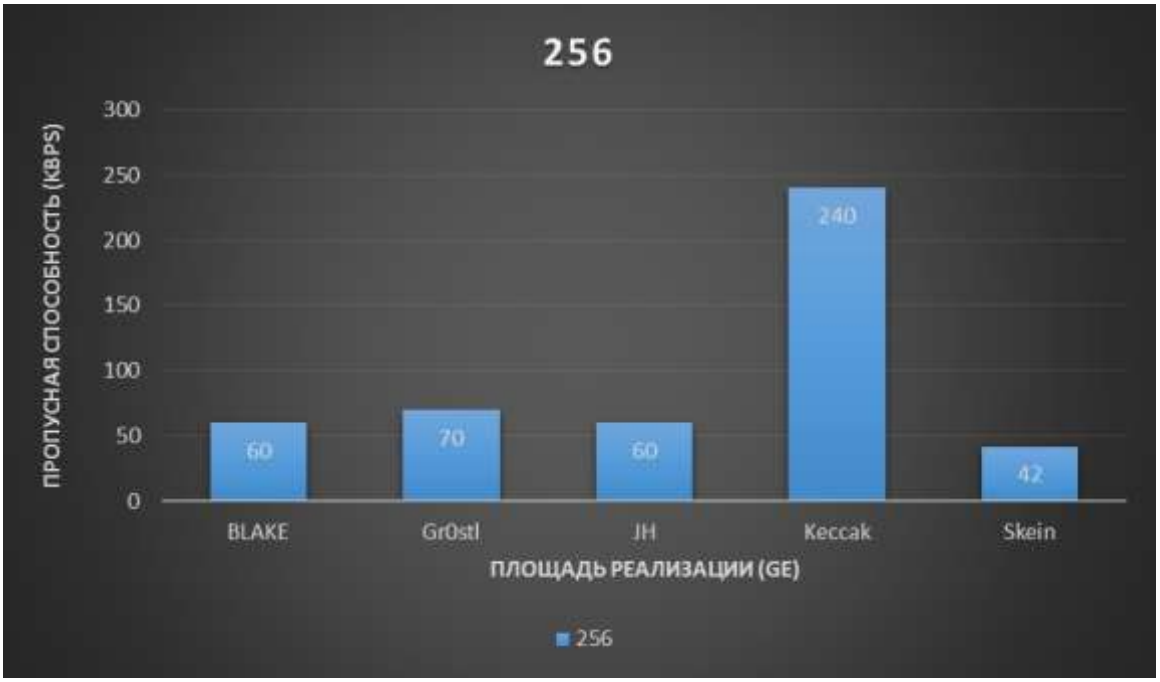


Рисунок 2.25 – Отношение пропускной способности к площади реализации для алгоритмов SHA-3 на схеме ASIC

На рисунке 5.11 кандидаты с наилучшим соотношением – энергосберегающие, в то время как кандидаты с большой разницей в показаниях гистограммы используют большое количество энергии, обрабатывая небольшие объёмы данных.

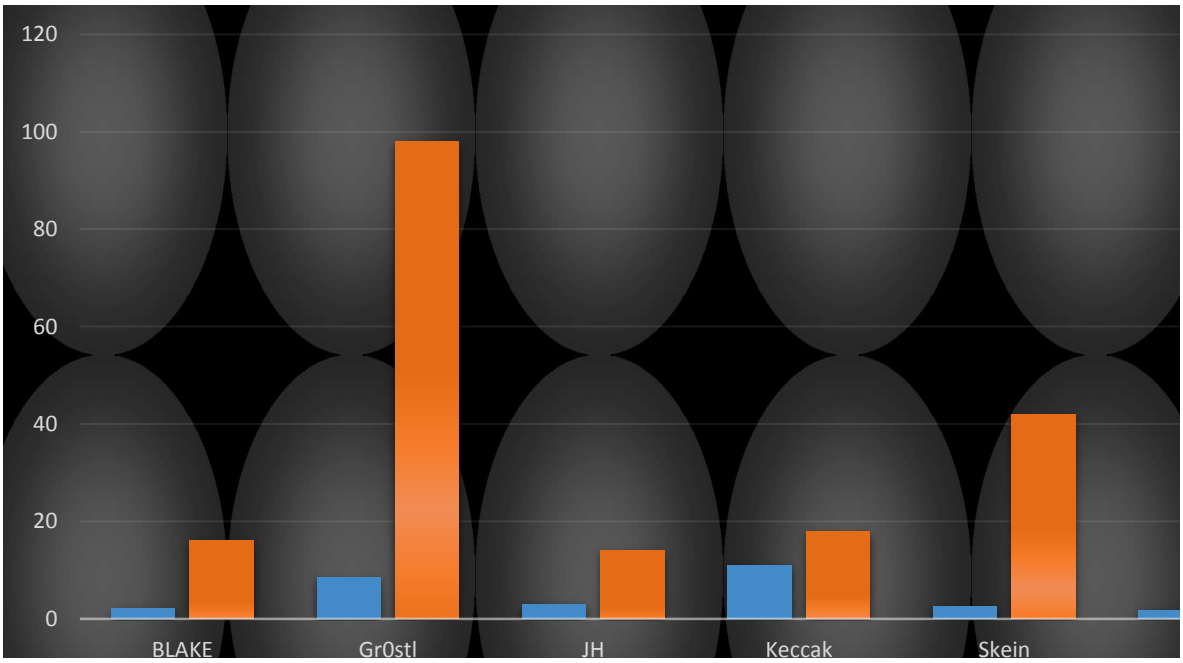


Рисунок 2.26 – Пропускная способность относительно потребляемой мощности для алгоритмов SHA-3 на схеме ASIC

Кандидат Keccak энергосберегающий и обладает высокой пропускной способности. Алгоритм хэширования Grøstl будучи быстрым, использует много энергии, что может быть приписано его большому размеру (Рисунок 5.12). Другие проекты с более низкой пропускной способностью, используют очень немного энергии.

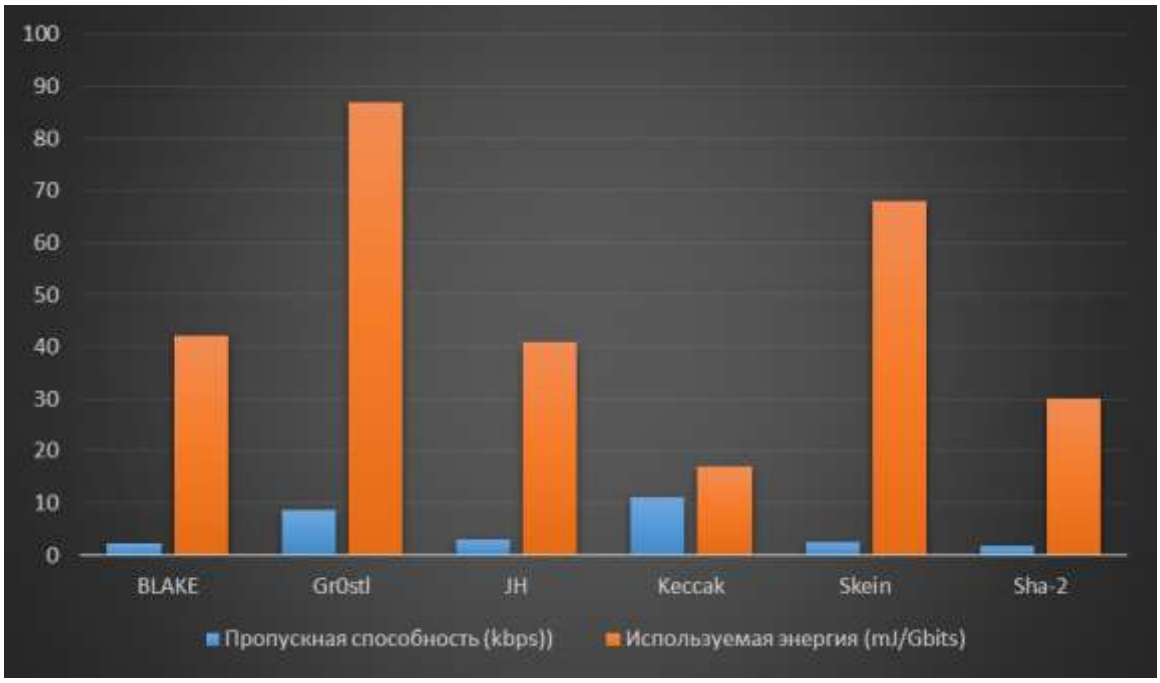


Рисунок 2.27 – Пропускная способность относительно затраченной энергии для SHA 3 реализованной на ASIC

В ходе проведения тестов были составлены рейтинги для реализаций алгоритмов хэширования кандидатов конкурса SHA-3.Основным критерием построения рейтинга является затрата энергии на обработку одного бита информации. Данный параметр позволяет оценить затраты на работу крипто-алгоритма. Представленные ниже таблицы (5.1-5.3) показывают рейтинги алгоритмов хэширования по параметрам затрат энергии и потребляемой мощности.

Таблица 5.1 – Сравнение затрат энергии и потребляемой мощности для реализаций на платформе ASIC-130-нм

130-нм			
Алгоритм хэширования	Тип архитектуры	Потребляемая мощность [MW]	Энергия / бит [МДж / Гбит]
Keccak	Basic Iterative	19.780	8.960
JH	Basic Iterative	13.010	20.630
BLAKE	Folded /2(h)	19.770	25.000
Skein	Unrolled x4	51.090	26.040
Grøstl	Basic Iterative	139.290	33.700

Таблица 5.2 – Сравнение затрат энергии и потребляемой мощности для реализаций на платформе ASIC-65-нм

65-нм			
Алгоритм хэширования	Тип архитектуры	Потребляемая мощность [MW]]	Энергия / бит [МДж / Гбит]
Keccak	Basic Iterative	8.160	5.160
JH	Basic Iterative	17.800	7.150
BLAKE	Folded /2(h)	16.470	21.616
Skein	Unrolled x4	26.190	16.730
Grøstl	Basic Iterative(P/Q)	46.010	24.333

Сравнение затрат энергии и потребляемой мощности для реализаций алгоритмов на платформе FPGA. Рейтинг построен относительно показателя затраченной энергии на обрабатываемый бит информации (Таблица 5.3).

Таблица 5.3 – Сравнение затрат энергии и потребляемой мощности для реализаций алгоритмов на платформе FPGA

Алгоритм хэширования	Тип архитектуры	Потребляемая мощность [мW]	Энергия / бит [мДж / Гбит]
BLAKE	Folded /4(h)	196.900	1,565.000
BLAKE	Folded /4(h)	120.800	1,283.000

JH	Basic Iterative	187.000	1,245.000
JH	Basic Iterative	185.500	1,240.000
Skein	Unrolled x4	205.200	1,060.000
Skein	Unrolled x4	205.400	1,058.000
Keccak	Basic Iterative	191.700	906.000
Keccak	Basic Iterative	186.900	698.000
Grøstl	Basic Iterative (P+Q)	223.900	675.000
Grøstl	Basic Iterative (P+Q)	215.500	628.000

2.5.5 Выводы по разделу

Результаты оценки на платформах ASIC и FPGA выявили, что алгоритм Keccak показывает высокие показатели производительности как на FPGA так и на ASIC из-за небольшого размера программного кода и высокой пропускной способности. Grøstl показал хорошие результаты на платформе ASIC с высокой пропускной способностью, но намного худшие результаты в FPGA из-за объёмной реализации функции. JH и как Skein выполняется умеренно хорошо в FPGA и ASIC в то время как хэш-функция BLAKE на всех системах замыкает список алгоритмов.

На ASIC, BLAKE является наименьшим по объёму SHA-3 кандидатом, однако он является вторым по объёму реализации на FPGA.

Результаты остальных кандидатов остаются относительно стабильными на двух платформах.

Данное исследование показало, что Keccak является наиболее энерго-эффективными при достаточных показателях быстродействия на большинстве платформ ASIC и FPGA.

Самыми энегро не обоснованными на платформе FPGA являются алгоритмы BLAKE и JH. В тоже время на платформе ASIC Grøstl является наиболее затратным.

Крипто-алгоритм Grøstl показывает не плохие показатели скорости, однако потребление энергии компенсирует данный факт. Остальные

алгоритмы потребляют не много энергии но и максимальная пропускная способность находится на низком уровне.

2.6 ОЦЕНКА ВЫЧИСЛИТЕЛЬНЫХ ЗАТРАТ АЛГОРИТМОВ ХЭШИРОВАНИЯ НА 8-БИТНЫХ ПРОЦЕССОРАХ

Различия в работе криптографических алгоритмов на различных аппаратных платформах являются одним из наиболее важных критериев отбора алгоритмов.

Три главных параметра, используемых в нашем исследовании: производительность (Таблица 6.1), количество блоков и производительность по отношению к количеству блоков[12].

Производительность определена как производительность для длинных сообщений или совокупная производительность для множества маленьких сообщений Такая определенная производительность не принимает во внимание время, потраченное для того, чтобы прочитать самый первый блок первого сообщения, инициализации сообщения, завершения сообщения, и записи, хэшь оценивает затраченную память.

Таблица 6.1 – Основные параметры 256- 512-битных кандидатов SHA-3

Функция	256bit вариант s			512-bit вариант		
	Размер блока, b	Состояние, s	Число раундов, r	Размер блока, b	Состояние, s	Число раундов, r
BLAKE	512	512	14	1024	1024	16
Grøstl	512	512	10	1024	1024	14
JH	512	1024	42	512	1024	42
Keccak	1088	1600	24	576	1600	24
Skein	512	512	72	512	512	72
SHA-2	512	256	64	1024	512	80

2.6.1 Исследования архитектуры аппаратных средств

Исследования архитектуры аппаратных средств хэш-функций – основная итеративная архитектура, показанная на рисунке. 6.1(а). Характерные функции этой архитектуры следующие: ширина информационного канала а) = размер состояния (обозначенный s), 6.1(б) один блок выполняется в единственном такте, 6.2(в) только одно сообщение обработано за один раз. Минимальным блочным временем обработки обычно называют

$$T_{block} = (r + f) \cdot T , \tag{6.1}$$

где r – число блоков ,

f – число тактов, требуемых для завершения вычисления блока (обычно 0 или 1),

T – минимальный такт. Соответствующей пропускной способностью называют

$$Tp = b/Tblock, \tag{6.2}$$

где b – размер блока сообщения в битах.

Основная итеративная архитектура обычно – предпочтительная архитектура для высокоскоростных аппаратных реализаций SHA 1, SHA 2 и SHA 3 кандидатов

Если у раунда хэш-функции есть симметричная структура с двумя или более подобными операциями, выполняемыми один за другим, возможно произвести горизонтальное сворачивание. На Рисунке 6.1 демонстрируется горизонтальное сворачивание, сворачивающаяся коэффициентом два. Мы обозначим эту архитектуру (h)/2. В этой архитектуре половина раунда реализована как комбинационная логика, и весь раунд выполняется, используя два цикла. Площадь остается такой же как в основной итеративной

архитектуре и равна размеру блока, s. Блочное время обработки представляется как

$$Tblock\text{-}/2(h) = (2 \cdot r + f) \cdot T/2(h). \qquad (6.3)$$

где T/2 <T/2(h) <T, T/2(h) ≈ T/2, и block/2 <block/2(h) < block.

В результате блочное время обработки (и пропускная способность) остается приблизительно такой же. Эти зависимости приводят к увеличению пропускной способности по отношению к обьему. Вообще, сворачивание коэффициентом k могло бы быть возможным, и соответствующая архитектура будет обозначена/k (h).

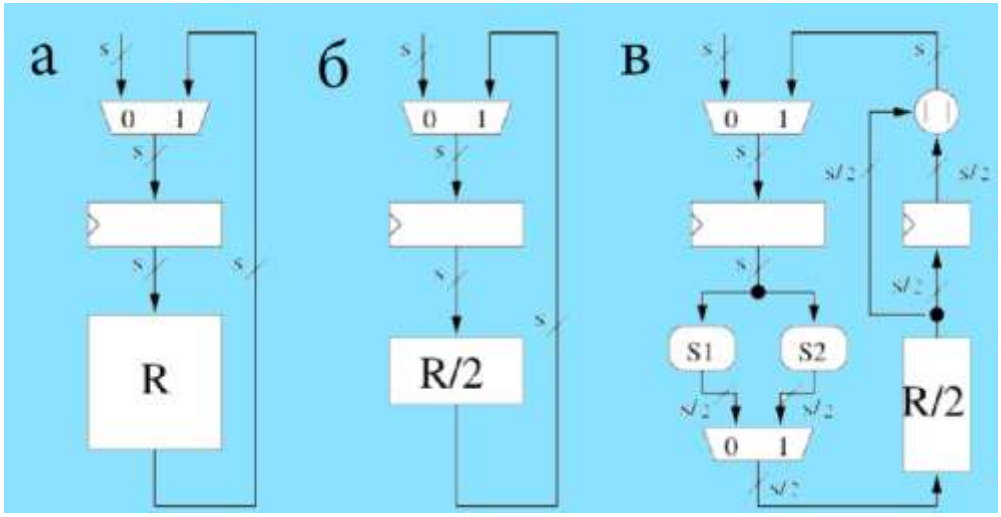


Рисунок 2.28 – Три аппаратные архитектуры хэш-функции: а), основной итеративный: x1, б) свернутый горизонтально коэффициентом 2:/2(h),
в) свернутый вертикально коэффициентом 2, /2(v). R – раунд, S1, S2 – функции отбора

Среди пяти финалистов единственный кандидат, который увеличил производительность при горизонтальном сворачивании-BLAKE.

BLAKE состоит из двух горизонтальных уровней идентичного G функции, разделенных только перестановкой. Реализовывая только один уровень в комбинационной логике, горизонтальное сворачивание с коэффициентом 2 может быть легко достигнуто. Каждая G функция имеет

очень симметричное строение вдоль горизонтальной оси и может быть легко свернута горизонтально – коэффициентом 2. В результате достигается фактор сворачивания 4, для всего блока. Другие SHA 3 финалисты, Grøstl, JH, Keccak, и Skein, не демонстрирует подобной симметрии.

В случае, если горизонтальное сворачивание или не возможно или не достигает необходимого сокращения площади, может применяться вертикальное сворачивание. На рисунке B, мы демонстрируем вертикальное сворачивание с коэффициентом 2. Мы обозначим это сворачивание /2(v).

В этой архитектуре datapath ширина уменьшена коэффициентом два. В результате два такта завершают цикл. В первом такте только биты внутреннего состояния, влияющего на первую половину вывода, обеспечены для ввода R/2. Во втором такте обработаны остальные биты внутреннего состояния. Первый вывод сохранен во вспомогательном регистре размера s/2 бит. Этот вывод связан с выводом от второй итерации, чтобы сформировать новое внутреннее состояние.

Период времени этой архитектуры приблизительно равен периоду времени основной итеративной архитектуры, T/2(v)≈T. В результате блочное время обработки, увеличится приблизительно в 2 раза по сравнению с основной архитектурой, как показано в уравнении ниже:

$$T_{block\text{-}/2(v)} = (2r+f) \cdot T_{/2(v)} \approx (2r+f) \cdot T. \qquad (6.4)$$

Уменьшение поперечного сечения менее актуально чем горизонтальная свертка. В результате пропускная способность к соотношению блока, потеряет работоспособность. В общем вертикальная свертка коэффициентом k стала возможной, и соответствующая архитектура будет обозначена/k(v).

Из пяти конечных SHA 3 кандидатов, BLAKE и Grøstl являются самыми оптимальными для вертикальной свертки. JH может быть свернут, но усиление в блоке, не принесет изменений, потому что блок JH очень прост, и не занимает большую площадью схемы. Для Skein и Keccak,

внутренняя круговая симметрия, необходимая для реализации вертикального сворачивания. Чтобы увеличить пропускную способность хэш-функции, можно применить различную архитектуру. Три общих подхода включают, конвейерную обработку и параллельную обработку. Для того, чтобы увеличить пропускную способность длинного сообщения используется развертка. Конвейерная обработка и параллельная обработка увеличивают комбинированную пропускную способность в случае обработки пакетов сообщений.

На рисунке 6.2(a), архитектура с разворачиванием коэффициентом 2. Мы обозначим эту архитектуру x2. datapath ширина остается как в основной повторяющейся архитектуре. К сожалению, в то же самое время, площадь цикла, увеличится с коэффициентом близкому к разворачивающему фактору. В результате в большинстве случаев пропускная способность по отношению к области уменьшается по сравнению с основной повторяющейся архитектурой. Также, архитектура с разворачиванием обычно используется только, когда пропускная способность для длинных сообщений представляет предельный интерес, и блок при множестве сообщений.

Однако, есть исключения к этому правилу. Развертка может улучшить пропускную способность и производительность, когда раунды, используемые алгоритмом в последующих повторениях, не являются одинаковыми. Среди пяти заключительных финалистов SHA-3 эта ситуация подходит только для Skein. В Skein операция объединения, используемая в качестве основного действия раунда, использует различную константу вращения в каждом из 8 последовательных повторений. В результате 8 к 1 мультиплексор должен использоваться в выполнении цикла для основной повторяющейся архитектуры Skein (x1). В 8x развернутой архитектуре, никакой мультиплексор не нужен. В результате мы можем ожидать, что пропускная способность к отношению к площади может стать оптимальной для одной из развернутых архитектур.

В наиболее распространенных интернет-протоколах безопасности, таких как IPSec, SSL и WLAN (802.11), входные данные представлены – пакетами. Максимальный размер пакета для Интернета ограничен так называемой Maximum Transmission Unit (MTU). Размер MTU для Ethernet базировался, на сообщениях 1500 байтов. Максимальная Единица Передачи для Интернет протокола IPv4 даже меньше, и установлена пределах 576 байт. Результат указан на рисунке 6.3. В типичном интернет-узле, у 80% обработанных пакетов размер 576 байт или меньше, и у 100% пакетов размеры, равные или меньшие чем 1500 байт. Рисунок 6.4 показывает зависимость потребляемой энергии от объёма обрабатываемого пакета.

Такие небольшие размеры пакетов означают, что сотни пакетов буферизованы в узлах обработки в форме очередей пакета, не представляя значительного увеличения времени прохождения пакета от источника до предназначения.

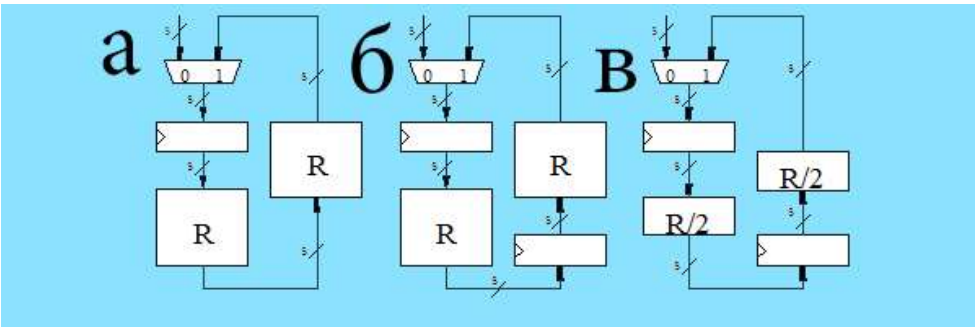


Рисунок 2.29 – Три архитектуры аппаратных средств функции хэширования а) развернутый коэффициентом 2: x2, б) развернутый коэффициентом с двумя настройками канала связи: x2-PPL2, в) основной повторяющийся с двумя настройками канала связи: x1-PPL2

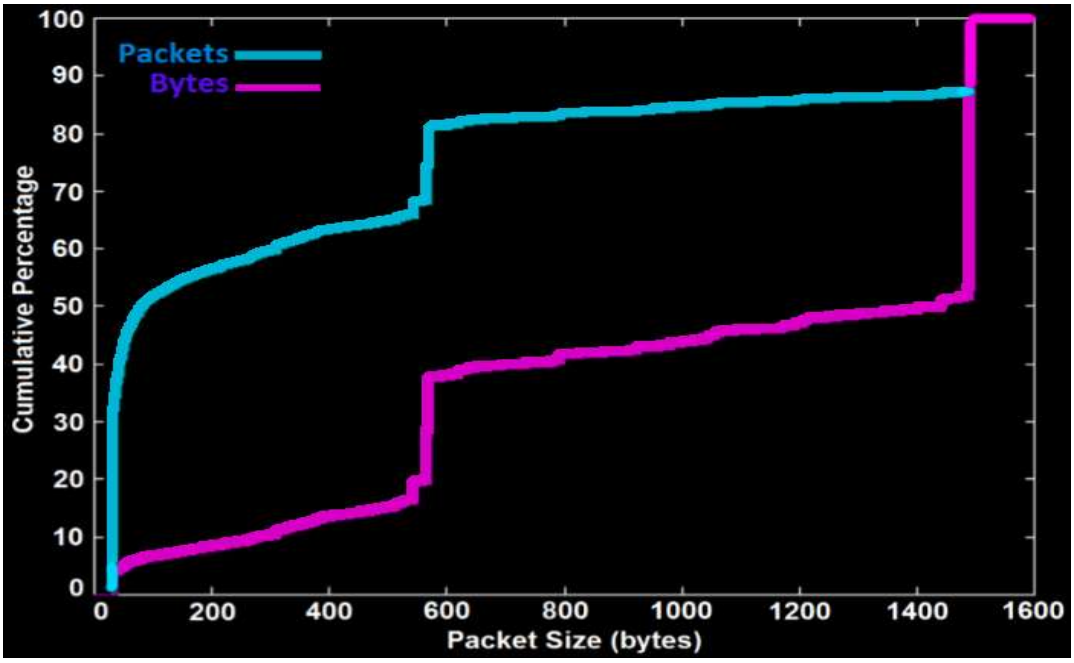


Рисунок 2.30 – Распределение размеров пакета, в интернет-узле обработки

Рисунок 2.31 – Затраты энергии на обработку пакетов данных

По этому, способности к параллельной обработке (включая конвейерную обработку), прежде всего ограничены общим размером блоков, а не числом сообщений, доступных параллельно. В этой статье мы предположим, что число сообщений, доступных параллельно, большое (по

крайней мере 10), и мы будем смотреть на комбинированную пропускную способность для всех доступных потоков данных.

Самый легкий способ осуществить конвейерную обработку в функциях хэширования состоит в том, чтобы сначала развернуть, и затем представить регистры между смежными раундами. Самый простой случай – архитектура, которая является два раза развернута и имеет две настройки канала связи как показано на рисунке 6.2(б). Мы обозначим эту архитектуру как x2-PPL2. Период времени этой архитектуры приблизительно равен периоду времени основной повторяющейся архитектуры, T. Обработка единственного блока требует то же самое число тактов как в основной повторяющейся архитектуре. Однако, так как два блока, принадлежащие двум различным сообщениям, обработаны одновременно, комбинированные увеличения пропускной способности в 2 раза. Пропускная способность к отношению области остается примерно той же в зависимости от специфического алгоритма.

Более оптимальный способ использовать конвейерную обработку состоит в том, чтобы представить регистры в функции хэширования. Пропускная способность к отношению области может увеличиться, но усовершенствование не гарантируется для всей Таблицы 6.2. Формулы времени обработки блока сообщения: b – размер блока, r – число раундов, f – число тактов, требуемых завершить вычисления для блока (f = 0 для Keccak и Grøstl (P+Q), f=1 для всех остающихся алгоритмов), k – складывающегося фактора или разворачивающего фактора, n – числа настроек канала связи, T – период времени.

Таблица 6.2 – Архитектура и время выполнения алгоритмов

Архитектура	Время, обработки 1 блока сообщения	Пропускная способность
Основной повтор, x1	$T_{block}= (r + f) \cdot T$	$Tp = b/T_{block}$
Свернутый коэффициентом k, /k	$T_{block}= (k \cdot r + f) \cdot T$	$Tp = b/T_{block}$
Развернутый коэффициентом k, xk	$T_{block}= (r/k + f) \cdot T$	$Tp = b/T_{block}$
Основной повтор с n настройками канала связи, x1-PPLn	$T_{block}= (n \cdot r + f) \cdot T$	$Tp = n \cdot b/T_{block}$
Свернутый коэффициентом k с n настройками канала связи, /k-PPLn	$T_{block}= (n \cdot k \cdot r + f) \cdot T$	$Tp = n \cdot b/T_{block}$
Развернутый коэффициентом k с n настройками канала связи, xk-PPLn	$T_{block}= (n \cdot r/k + f) \cdot T$	$Tp = n \cdot b/T_{block}$

Алгоритмы семейства FPGA, показывают нулевые или отрицательные улучшения показателей в случае, если основная повторяющаяся архитектура уже работает близко к минимальному времени, доступного для данного семейства FPGA.

Формулы продолжительности обработки блока и пропускной способности всей вышеупомянутой архитектуры получены в Таблице 6.2.

2.6.2 Исследование пропускной способности

Представляем подробную пропускную способность представленную комбинированными гистограммами для всех реализаций архитектуры 256-разрядных шести исследованных алгоритмов в Xilinx Virtex 5 FPGA.

Для алгоритма BLAKE две лучшее архитектуры с точки зрения пропускной способности к отношению блока:/4(h)/4(v), то есть, архитектура с горизонтальным сворачиванием коэффициентом 4, объединение с вертикальным сворачиванием коэффициентом 4; и x1-PPL2, основная архитектура с двумя настройками канала связи. Хорошее исполнение этих

двух архитектуры привело к значительным сокращениям сложности функции BLAKE [16].

Для Grøstl мы рассматриваем две главные архитектуры: а) подобная архитектуре, обозначенной (P+Q), в котором осуществлены перестановки Grøstl P и Q, используя две независимых схемы, работающих в параллели и б) архитектура, обозначены (P/Q), в которой используется, та же самая схема, чтобы произвести вычисления и P и Q, принадлежащие этим двум перестановкам Вертикальное сворачивание с коэффициентом 2 обеспечивает существенное увеличение производительности.

Для JH рассматриваем два главных типа архитектуры: а) с константами, сохраненными в памяти, JH и б) с константами, мгновенными вычислениями, JH (OTF). Оба варианта, приводят к подобным результатам для основной повторяющейся архитектуры, x1. Ни вертикальное сворачивание, ни конвейерная обработка, не эффективны когда применяются непосредственно к основной архитектуре. Вертикальное сворачивание, не улучшают пропускную способность. В результате основная повторяющаяся архитектура остается самой эффективной с точки зрения пропускной способности по отношению к площади.

Отношение близкое к 1 показывает, что функция может быть портирована на 8-разрядную архитектуру без большой потери производительности. Высокое отношение означает потерю производительности когда портировано на 8-разрядную архитектуру.

Таблица 6.3 – Вычислительные затраты алгоритмов реализованных на схеме FPGA для 256-битных функций

Архитектура	Особенности строения	Частота [MHz]	Пропускная способность [Mbit/s]	Блок [CLB slices]	Производ [(Mbit/s)/ CLB_slices]
	BLAKE-32 (Round 3 S pecification)				
/2(h)	64-bit FIFO	128.9	3143	1523	2.06

	интрерфейс				
/2(h)	ideal interface	115.0	2676	1660	1.61
/4(h)	ideal interface	118.1	1512	1118	1.35
	Grøstl-0-256 (Round 3 Specification)				
x1(P+Q)	-	200.7	10276	1722	5.97
x1(P/Q)	64-bit FIFO interface	323.4	7885	1597	4.94
x1(P+Q)	S-boxes in dist. memory	154.0	7885	2616	3.01
	JH-256 (Round 3 Spe cification)				
x1	64-bit FIFO interface	380.8	5416	1018	5.32
x1	ideal interface	250.1	3557	1291	2.76
x1	ideal interface	201.0	2639	2661	0.99
	Keccak				
x1	64-bit FIFO interface	282.7	12817	1272	10.08
x1	ideal interface	189.0	8225	1117	7.36
x1	ideal interface	205.0	8397	1433	5.86
	Skein-512-25 6				
x4	64-bit FIFO interface	118.0	3178	1621	1.96

Проведено систематическое исследование быстродействующей архитектуры аппаратных средств для пяти заключительных кандидатов SHA-3.

Исследованная архитектура была основана на принципе повторяющейся архитектуры, горизонтального сворачивания, вертикального сворачивания, разворачивания, конвейерной обработки и параллельной обработки, используя разнообразные независимые схемы. Каждая архитектура была реализована, используя четыре высокоэффективных

семейства FPGA: Virtex 5 и Virtex 6 от Xilinx, и Stratix III и Stratix IV от Altera. Из полученных результатов, мы идентифицировали самую эффективную архитектуру аппаратных средств для каждого исследованного алгоритма, основанного на лучшей производительности по отношению к области.

В случае четырех из пяти кандидатов (все кроме JH), самая эффективная архитектура, казалось, pipelined архитектурой. Оптимальное число настроек канала было равным два для Keccak и Grøstl, и четыре для BLAKE. Оптимальной архитектурой для Skein является 4 цикловая свертка.

Результаты всех исследованных функций показали, что Keccak – единственный кандидат, который выигрывает у SHA-2 для всех сравнений FPGA и двух разновидности функции хэширования с 256-битовой и 512-битовой последовательностями. Единственный недостаток этой функции, не возможность сворачивания, и таким образом требует, минимального количества блоков в диапазоне 1400 блоков CLB в Virtex 5

JH выступил лучше чем SHA-2 на трех из четырех схемах, JH является самым эффективным при основной повторяющейся архитектуре.

Grøstl единственный кандидат, выигрывающим у SHA-2 в одном тесте для 256-битных реализаций, осуществленных, используя Virtex 5. Данное преимущество было достигнуто только на относительно больших площадях приблизительно 3000 блоков CLB.

Хотя Grøstl был подходящим для вертикального сворачивания, структура алгоритма вызвала очень существенное уменьшение в скорости.

Skein- единственный финалист, который может существенно улучшить производительность через сворачивание и является единственным алгоритмом, который может быть pipelined до 10 раз. Это дает преимущество по сравнению с другими алгоритмами для 512-битовых разновидностей функций хэширования, осуществленных, используя Altera.B LAKE – алгоритм с самой высокой гибкостью и наибольшим числом разнообразной архитектуры. BLAKE может быть легко свернут горизонтально и

вертикально при коэффициентах два и четыре. BLAKE также может быть легко pipelined даже в свернутой архитектуре. Это единственный алгоритм, у которого есть относительно эффективная архитектура.

2.6.3 Оценка вычислительных затрат алгоритмов хэширования на 8-битных процессорах

NIST поощряет оценку кандидатов на 8-битных платформах. Встроенные микропроцессоры широко используются в различных приложениях, включая смарт-карты, бытовую технику, медицинские устройства, системы транспортировки и еще много. Многие из этих приложений требуют эффективной криптографии. Возникает вопрос об эффективности реализации на 8-битных встроенных микропроцессорах крипто алгоритмов. Вычислительные затраты алгоритмов финалистов конкурса SHA-3 оптимизированных для 8-битных платформ рассматриваются в таблице 6.5. Отношение близкое к 1 означает, что функция может быть портирована на 8-разрядную архитектуру без больших затрат на реализацию. Большой показатель показывает потерю производительности при портировании на 8-разрядную архитектуру.

Таблица 6.5 –Возможность потрирования финалистов конкурса SHA-3 на 8-битные платформы

Алгоритм	8051		ARM7TDMI		Коэффициент оптимизации
	cycles	cycles/byte	cycles	cycles/byte	
Blake 224/256	41135	642.7	8215	128.4	5
Blake 384/512	81390	635.9	19157	149.7	4.2
Grøstl 224/256	84915	1326.8	56618	884.7	1.5
Grøstl 384/512	228781	1787.4	155542	1215.2	1.5
Keccak 224	237352	1648.3	50834	353	4.7

Keccak 256		1745.2		373.8	
Keccak 384		2282.2		488.8	
Keccak 512		3296.6		706	
Skein 224/256	62222	1944.4	10079	315	6.2

По сравнению с SHA-3 на базовых платформах 8-разрядные микропроцессоры в большой степени ограничены в ресурсах, таких как память и RAM(Таблица 6.6).

Помимо пропускной способности, у эффективности есть дополнительное значения: ресурсы, необходимые для реализации хэш-функции, должны быть минимальны, так как встраиваемые приложения очень часто ограниченны (Рисунок 6.5). Потребление оперативной памяти может быть более важным критерием чем пропускная способность.

Рассмотрим реализацию нескольких SHA3 кандидатов на 8-разрядной платформе микроконтроллера. Мы представляем результаты для хэш-алгоритмов BLAKE, Skein JH Keccak, Grøstl.

Таблица 6.6 – Размер и результаты производительности

Хэшь функция	Размер кода bytes	RAM bytes	Короткое сообщение cycles/byte	Длинное сообщение cycles/byte
BLAKE	1804	251	332	324
JH	3558	320	247	227
Grøstl	4852	261	1044	687
Keccak	2088	104	600	588
Skein	5254	232	338	331

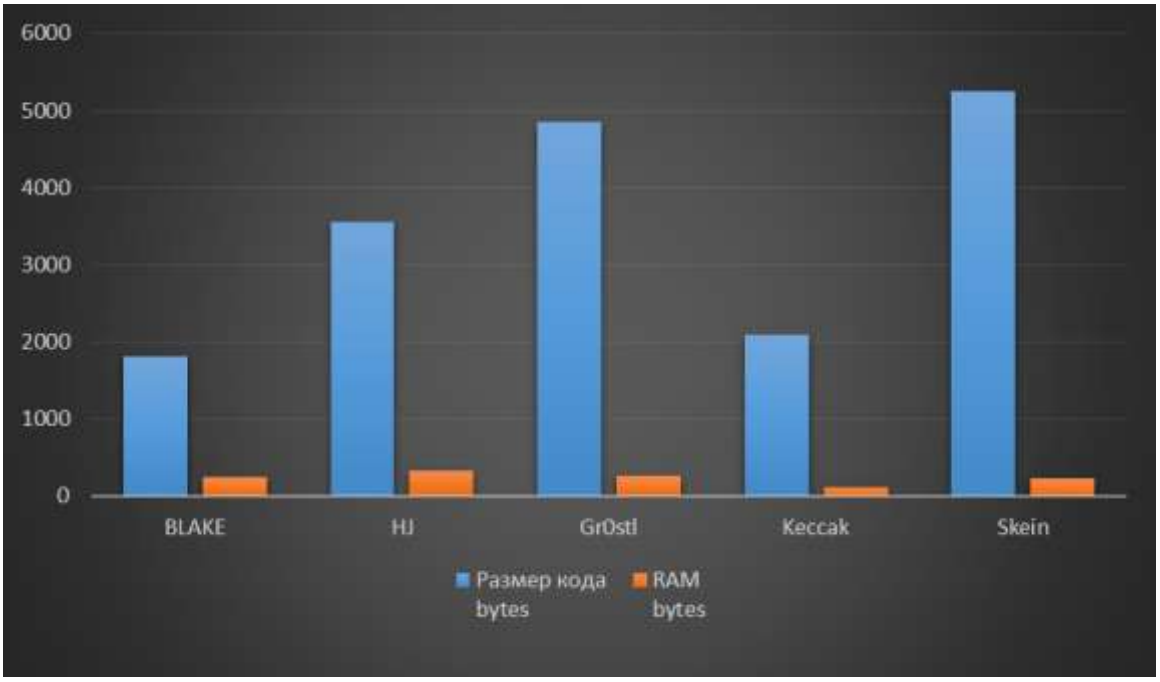


Рисунок 2.32 – Потребление оперативной памяти

Алгоритмы разработаны для 8-разрядных процессоров AVR, семейства микроконтроллеров RISC на 8-бит, широко используемых во многих встраиваемых приложениях. Процессоры Atmel AVR работают в частотах процессора до 32 мгц, SRAM, до сотен килобайтов памяти программы Flash, и дополнительный EEPROM или ROM. У устройств семейства AVR есть 32 регистра общего назначения 8-разрядного размера слова. Пример – Atmel микроконтроллер общей цели ATmega128 (Таблица 6.7). Ядро AVR также доступно как процессор смарт-карты, известный как семейство AT90SCxxx. Микроконтроллеры AVR могут быть запрограммированы в ассемблере AVR и в C.

Таблица 6.7 – Рейтинг оптимизации алгоритмов на 8-разрядной архитектуре

	256 bit вывод	
Рейтинг	8051	ATmega1281 [7]
1	Keccak	Skein

2	Skein	JH
3	JH	BLAKE
4	BLAKE	Keccak
5	Grøstl	Grøstl
	512 bit вывод	
Рейтинг	8051	ATmega1281 [7]
1	Keccak	Skein
2	JH	Keccak
3	BLAKE	Grøstl
4	Skein	BLAKE
5	Grøstl	JH

Выявление возможности перенесения алгоритмов на встраиваемые архитектуры.

Представленные реализации разработаны для процессора AVR, обеспечивающем 1-килобайт SRAM и несколько килобайтов памяти.

Все реализации тестировались и обрабатывались на смарт-картах разработчика, оборудованная ATMega163 microcontoller полностью реализованы в ассемблере AVR. Не все шифры одинаково хорошо подходят для реализации на 8-разрядной платформе. Мы рассматривали алгоритмы, которые соответствуют 8-разрядной платформе. (Таблица 6.8).

Таблица 6.8 – Результирующие коэффициенты оптимизации для 8-разрядной архитектуры

256 bit вывод			
Хэшь функция	8051	ATmega1281 [7]	Коэффициент оптимизации
BLAKE	642.7	884	0.7
JH	566.2	485	1.2

Grøstl	1326.8	1309	1.0
Keccak	1745.2	4572	0.4
Skein	1944.4	1204	1.6
512 bit вывод			
Хэшь функция	8051	ATmega1281 [7]	Коэффициент оптимизации
BLAKE	635.9	379.5	0.2
JH	590.8	296.9	0.2
Grøstl	1787.4	3365	0.5
Keccak	3296.6	7945	0.4
Skein	2046	1443	1.4

Из полученных коэффициентов можно определить наиболее универсальные алгоритмы для работы на различных системах. При хэшировании последовательности в 256 бит наименьший показатель у двух алгоритмов Keccak и BLAKE. В то время как при 512 выходе только функция Skein показывает результат больше единицы.

2.6.4 Итоги оценки вычислительных затрат на 8 битных процессорах

Хотя производительность всегда имеет высокий приоритет для выбора алгоритма для встроенных систем, размер кода и потребление RAM часто играют существенную роль. Чем меньше ресурсов используется, тем меньший и следовательно более дешевый используемый микроконтроллер. Производительностью Keccak выигрывает у других реализаций, BLAKE и Skein. Grøstl – единственный кандидат, который показывает значительную разницу в производительности для хэширования длинных сообщений, но его эффективность работы не очень высока. С точки зрения размера кода BLAKE и Keccak вполне реализуются на 2 килобайтах кода. Другие реализации значительно больше. Требование RAM для всех алгоритмов минимальны.

BLAKE показывает лучшую эффективность работы, обладая небольшим потреблением ресурсов и приличной пропускной способностью. JH дает немного лучшую пропускную способность за счет удвоения размера кода. Grøstl и Skein менее интересны из-за большого размера кода и, в случае Grøstl, значительно более низкой пропускной способности. Наименьшие затраты на реализацию при портировании на 8 битную архитектуру возможны для алгоритмов Keccak, BLAKE, JH. Хотя структура алгоритма и являлась оптимальной для хэш-алгоритма Grøstl требуемые затраты RAM и обьем кода программы уменьшают данное преимущество.

2.7 МЕТОДОЛОГИЯ ПРОЕКТИРОВАНИЯ

Проекты для основной итеративной, свернутой, и развернутой архитектуры используют интерфейс и протокол связи. Проекты для архитектуры с конвейерной обработкой, используют интерфейс и окружающую логику, показанную на рисунке. 3. Ввод FIFO обслуживается пакетными очередями. Каждый в порядке поступления связывается с соответствующим дополнительным модулем и ассоциированным Finite State Machine 1 (FSM1). FSM1 ответственен за то, что загрузил следующий блок данных и дополнил последний блок сообщения, при необходимости (возможно параллельно с информационным каналом, обрабатывающим предыдущий блок под управлением FSM2). Выводы, соответствующие четырем независимым пакетам, сначала сохранены в соответствующей параллели - в последовательных модулях, и затем мультиплексированы к выводу в порядке поступления.

Вся архитектура была смоделирована в VHDL 93. Все коды VHDL были полностью проверены, используя всеобщий испытательный стенд, способный к тестированию произвольного ядра хэш-функции. Дополнительный сценарий специальности был разработан в Perl, чтобы дополнить сообщения, включенные в файлы Known Answer Test (KAT), распределенные как часть пакета представления каждого кандидата.

Для синтеза и реализации, использовались инструменты, разработанные поставщиками FPGA:

- для Xilinx: Xilinx ISE Design Suite v. 13.1, включая Xilinx XST;
- для Altera: Quartus II v 11.1 Subscription Edition Software.

Генерация большого количества результатов и оптимизации опций инструмента была облегчена открытым исходным кодом, тестирующим в сравнении с эталоном среды, названной ATHENa [2,16]. Все графики результата включали в это бумажное использование цветовые коды, представленные Bernstein и Lange в [6,12].

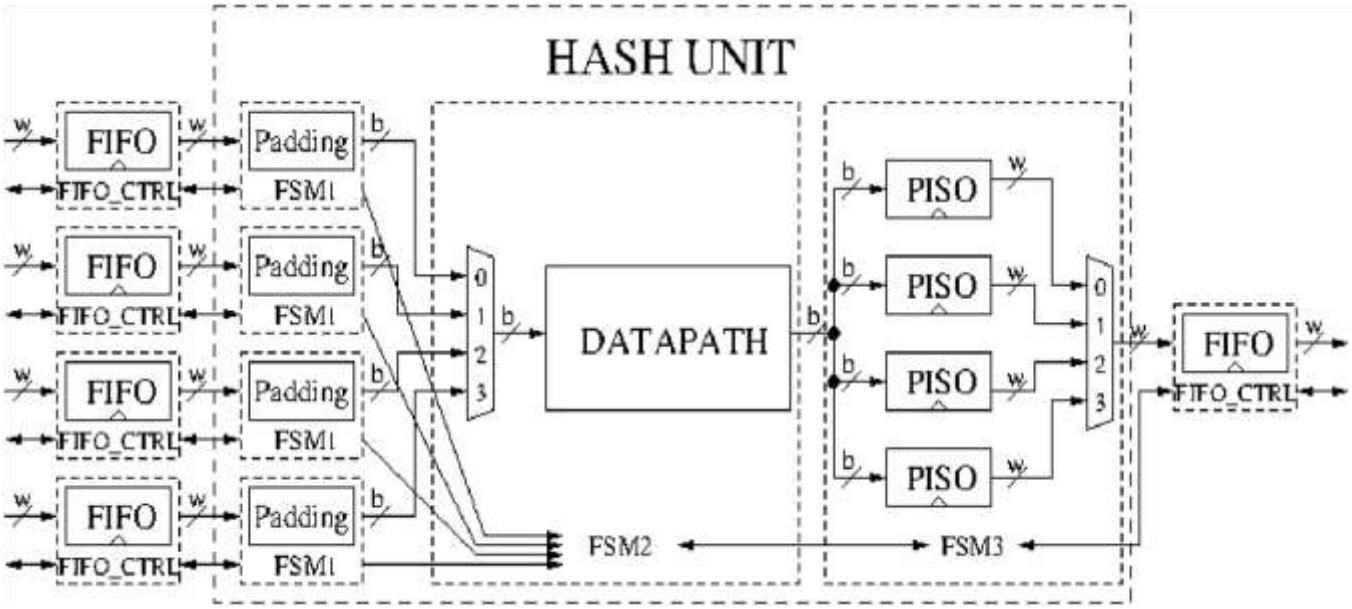


Рисунок 2.33 – интерфейс, высотная блок-схема и окружающая логика модуля хэш для архитектуры с конвейерной обработкой с четырьмя стадиями конвейерной обработки.

2.7.1 Результаты тестирования с дополнительными модулями

Результаты реализаций с дополнительными модулями получены в итоге и представлены на рис. 7.2-7.7 и в табл. 7.1 и 7.2. На рис. 7.2 и 7.3 показано подробную пропускную способность по сравнению с комбинированными гистограммами для всей реализованной архитектуры 256-разрядных разновидностей шести исследованных алгоритмов в Xilinx Virtex 5 и Altera Stratix, соответственно.

2.7.1.1 Результаты для хэш-функции BLAKE

Для BLAKE (рис. 7.2a и 7.3a), лучшая архитектура - x1-PPL4, то есть, основная архитектура с четырьмя стадиями конвейерной обработки. Хорошая производительность этой архитектуры связана с симметричными строениями основного раунда и каждой отдельной G функции, которые облегчают деление информационного канала на четыре устойчивых стадии

конвейерной обработки (рисунок 14 и 15). Лучшая архитектура без конвейерной обработки:

– для Virtex 5:/4 (h)/4 (v)-m, то есть, архитектура с горизонтальным сворачиванием фактором 4, объединенный с вертикальным сворачиванием фактором 4, и внутреннее состояние сохранено в памяти;

– для Stratix:/2 (h), то есть, архитектура с горизонтальным сворачиванием фактора 2.

Хорошая производительности из этих двух архитектур связана со значительным сокращением сложности входного модуля перестановки в результате вертикального сворачивания фактора 4. Две менее успешные архитектуры включая x1 и /2 (h)-PPL4 для Virtex 5, и x1 и x1-PPL2 для Stratix. Эта архитектура не включена в наши объединенные графики, показанные в рисунках 7.4-7.7.

2.7.1.2 Результаты для хэш-функции Groestl

Для Groestl (рис. 7.2b и 7.3b), рассматривается два главных типа архитектуры: a) проводится параллель архитектуре, обозначенной (P+Q), в котором перестановки Groestl P и Q реализованы, используя два автономных устройства, параллель, и архитектуру с подобной конвейерной обработкой b), обозначено (P/Q), в котором, тот же самый модуль используется, чтобы реализовать и P и Q, и вычисления, принадлежащие этим двум перестановкам.

Применяется вертикальное сворачивание и конвейерная обработка к обеим архитектурам. Лучшая архитектура: x1-PPL2 (P+Q) для Virtex 5, и x1-PPL4 (P+Q) для Stratix. Лучшая архитектура без конвейерной обработки: x1 (P/Q) для Virtex 5, и/2 (v) (P/Q) для Stratix. Свернутая параллельная архитектура, /k (v) (P+Q), медленнее, чем архитектура с подобной конвейерной обработкой (P/Q) использование сопоставимой области. То же самое для основной итеративной параллельной архитектуры, x1 (P+Q). Попытка конвейерной обработки Groestl, использовать 7 стадий конвейерной

обработки (x1-PPL7), используя, реализацию только для логики S-блоков, была довольно неуспешна.

2.7.1.3 Результаты для хэш-функции JH

Для JH (рис. 7.2c и 7.3c), рассматривается два главных типа архитектуры: a) с круглыми константами, сохраненными в памяти, JH (MEM) и b) с круглыми константами, быстро рассчитанными, JH (OTF). Оба подхода, кажется, имеют результатом подобную производительность для основной итеративной архитектуры, x1. Ни вертикальное сворачивание, ни конвейерная обработка, кажется, не эффективна, когда применено прямо к основной архитектуре. Вертикальное сворачивание два, несколько неожиданно, область увеличений и основная архитектура с двумя стадиями конвейерной обработки не улучшает пропускную способность. Оба нежелательных результата могут быть прослежены назад к простоте основного раунда. Сворачивание не уменьшает область из-за дополнительных регистров и мультиплексоров, представленных очень простому раунду. Конвейерной обработка не увеличивает пропускную способность, потому что простой основной раунд уже имеет очень малую задержку и разделяется на две хорошо сбалансированные стадии конвейерной обработки. В целом, лучшая архитектура: x1 (MEM) для Virtex 5 и x2-PPL2 (MEM) для Stratix.

2.7.1.4 Результаты для хэш-функции Keccak

Для Keccak (рис. 7.2d и 7.3d), лучшая архитектура - основная итеративная архитектура. Конвейерная обработка, кажется, вполне неуспешна в Virtex 5 и несколько более успешна в Stratix, где три различные архитектуры конвейерной обработки (x1-PPL2, x2-PPL2, и x2-PPL4) дают подобную пропускную способность соотношению площадей как основная итеративная архитектура, x1. Вертикальное сворачивание было предпринято

только для версии без дополнения, и поэтому в соответствующих результатах не показывают, рисунки 4-9. Как показано в Таблице 5, вертикальное сворачивание фактором 8, с внутренним состоянием, сохраненным в памяти, приводит к уменьшению площади фактором приблизительно 4 для Virtex 5 и приблизительно 1.5 для Stratix, за счет значительного сокращения пропускной способности, фактором приблизительно 16 в Virtex 5 и фактором приблизительно 18 в Stratix. Таким образом, пропускная способность к соотношению площадей уменьшается фактором 4 для Virtex 5 и фактором 12 для Stratix.

2.7.1.5 Результаты для хэш-функции Skein

Для Skein (рис. 7.2e и 7.3e), развернутый 4 архитектурой, x4, значительно более эффективен, чем основная архитектура, x1. Одновременно, разворачивание 8 не дает дополнительного улучшения пропускной способности к соотношению площадей. Лучшая система конвейерной обработки для архитектуры получена первой разматывающей основной архитектурой с фактором четыре, и затем конвейерной обработкой полученной схемы, используя, два этапа в Virtex 5 и пять этапов в Stratix. Пять стадий конвейерной обработки эффективны в Stratix из-за дополнительного добавления, выполняют каждый четвертый раунд, но они не улучшают полную пропускную способность к соотношению площадей в Virtex 5.

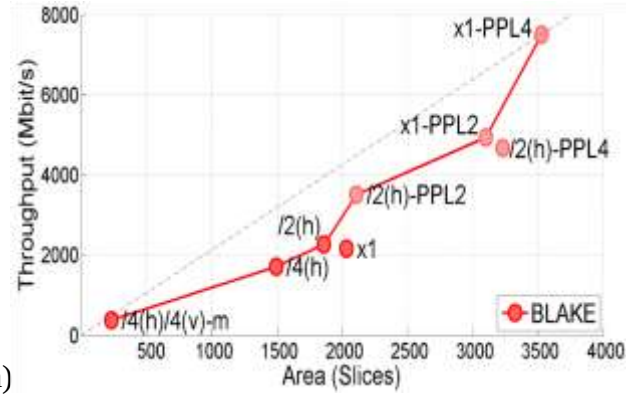
2.7.1.6 Результаты для хэш-функции SHA 2

Для SHA 2 (рис. 7.2f и 7.3f), ни один из обсужденных методов не применяется. Реализация этой функции уже мала, значит сокращение области не необходимо. Лучшим способом ускорить эту функцию являются при использовании кратных автономных устройств SHA 2 работы в

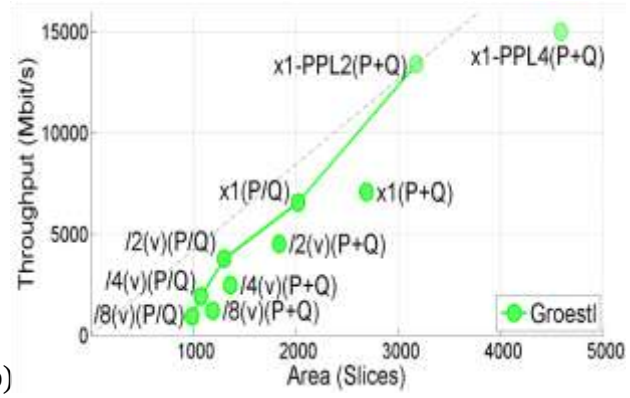
параллели. Мы обозначаем эту архитектуру MUn, где n обозначает число модулей хэша.

2.7.1.7 Общие результаты

Объединенные графики для 256-разрядных разновидностей и 512-разрядных разновидностей всех алгоритмов, реализованное использование Xilinx Virtex 5 FPGA, представлены на рис. 7.4 и 7.5. Отдельные точки, помещенные в равные интервалы на пунктирных линиях, представляют многокомпонентную архитектуру. Алгоритмы могут быть признаны первыми с точки зрения пропускной способности к соотношению площадей их лучшей архитектуры, как идентифицировано выше. Это вызвано тем, что эта архитектура может быть легко тиражирована, учитывая обработку и потоки данных в параллель. И пропускная способность и область увеличатся фактором n. Вторичный критерий - область лучшей архитектуры. Чем более маленькая область, тем более плотный график, представляющий возможные расположения заданной функции на пропускной способности по сравнению с комбинированной гистограммой.



a)



b)

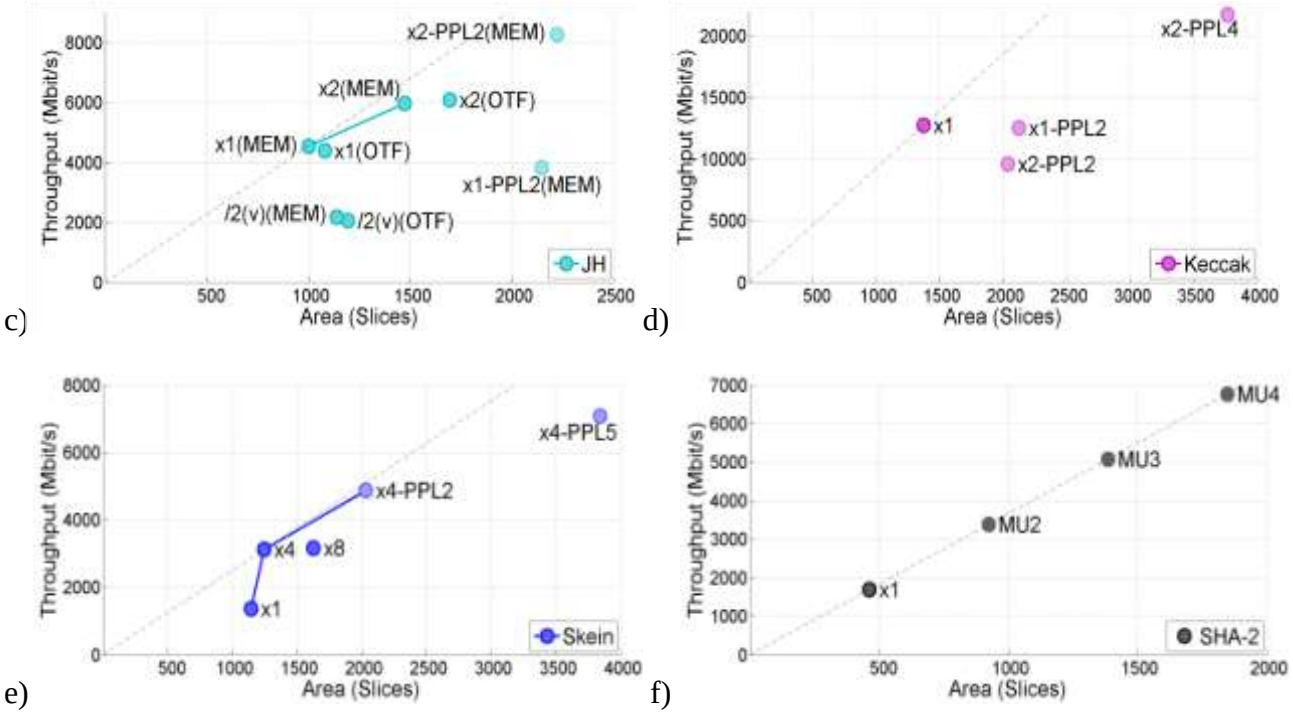


Рисунок 2.33 – Пропускная способность и область с комбинированными гистограммами для кратной архитектуры a) BLAKE-256, b) Groestl-256, c) JH-256, d) Keccak-256, e) Skein-256, и f) SHA 256, реализованный в Xilinx Virtex 5 FPGA.

Пояснение: x1 - основная итеративная архитектура,/k (h) - горизонтально свернутый фактором k,/k (v) - вертикально свернутый фактором k,/k (v)-m - вертикально свернутый фактором k с внутренним состоянием, сохраненным в памяти, xk - развернутый фактором k, PPLn – конвейерной обработкой с n стадии конвейерной обработки, (P+Q) - проводит параллель архитектуре Groestl, P/Q – без конвейерной обработкой архитектуры Groestl, MEM - архитектуры JH с круглыми константами, сохраненными в памяти, OTF - архитектура JH с круглыми константами.

a) b)

c) d)

e) f)

Рисунок 2.34 – Пропускная способность в области с комбинированными гистограммами для кратной архитектуры a) BLAKE-256, b) Groestl-256, c) JH-256, d) Keccak-256, e) Моток пряжи 256, и f) SHA 256, реализованный в Altera Stratix FPGA.

Пояснение: x1 - основная итеративная архитектура,/k (h) - горизонтально свернутый фактором k,/k (v) - вертикально свернутый фактором k,/k (v) метровый - вертикально свернутый фактором k с внутренним состоянием, сохраненным в памяти, xk - развернутый фактором k, PPLn - конвейерная обработка с n стадии конвейерной обработки, (P+Q) - проводит параллель архитектуре Groestl, P/Q – без конвейерной обработки архитектуры Groestl, MEM - архитектуры JH с круглыми константами, сохраненными в памяти, на пол - архитектура JH с круглыми константами.

Результаты для 256-разрядных разновидностей хэш-функций показывают на рис. 7.4. Keccak - единственная функция, которая значительно превосходит по быстродействию SHA 2 с точки зрения пропускной способности к соотношению площадей. Keccak также значительно более быстр чем SHA 2 для любой области, больше чем 1400 циклов CLB. Groestl и

JH демонстрируют производительность, подобную SHA 2, с реализациями JH немного быстрее чем SHA 2 последовательно запуска приблизительно с 1000 циклов CLB и реализации Groestl, превышающие скорость SHA 2 (и JH) только для одной реализации, показанной в схеме, сопровождая 3000 слоев CLB. Skein и BLAKE запаздывают значительно позади SHA 2, независимо от области.

Результаты для 512-разрядных разновидностей хэш-функций, показанных на рис. 7.5, вполне подобны, за исключением того, что, JH выполняется почти одинаково хорошо как Keccak (из-за уменьшения в размере блока сообщения Keccak от 1088 до 576 битов), Groestl превосходит по быстродействию SHA 2 и Skein только для области приблизительно 6000 циклов CLB, у Skein производительность близко к SHA 2, и BLAKE последний из 5.

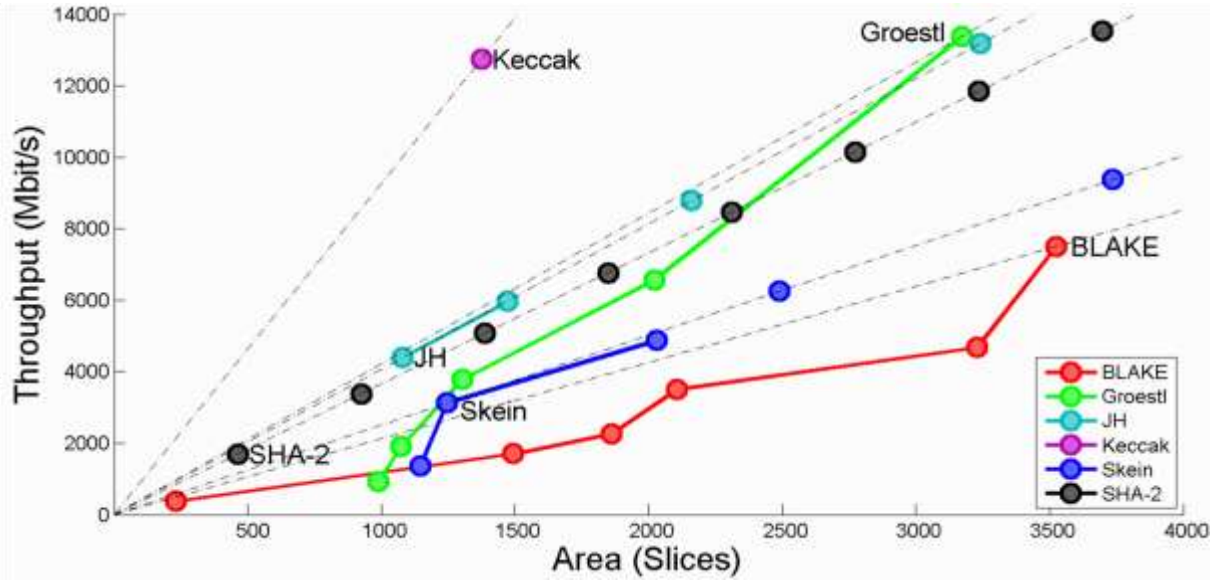


Рисунок 2.35 – Объединенная пропускная способность и область в комбинированной гистограмме для кратной аппаратной архитектуры 256-разрядных разновидностей BLAKE, Groestl, JH, Keccak, Skein и SHA 2 реализованных в Xilinx Virtex 5 FPGA.

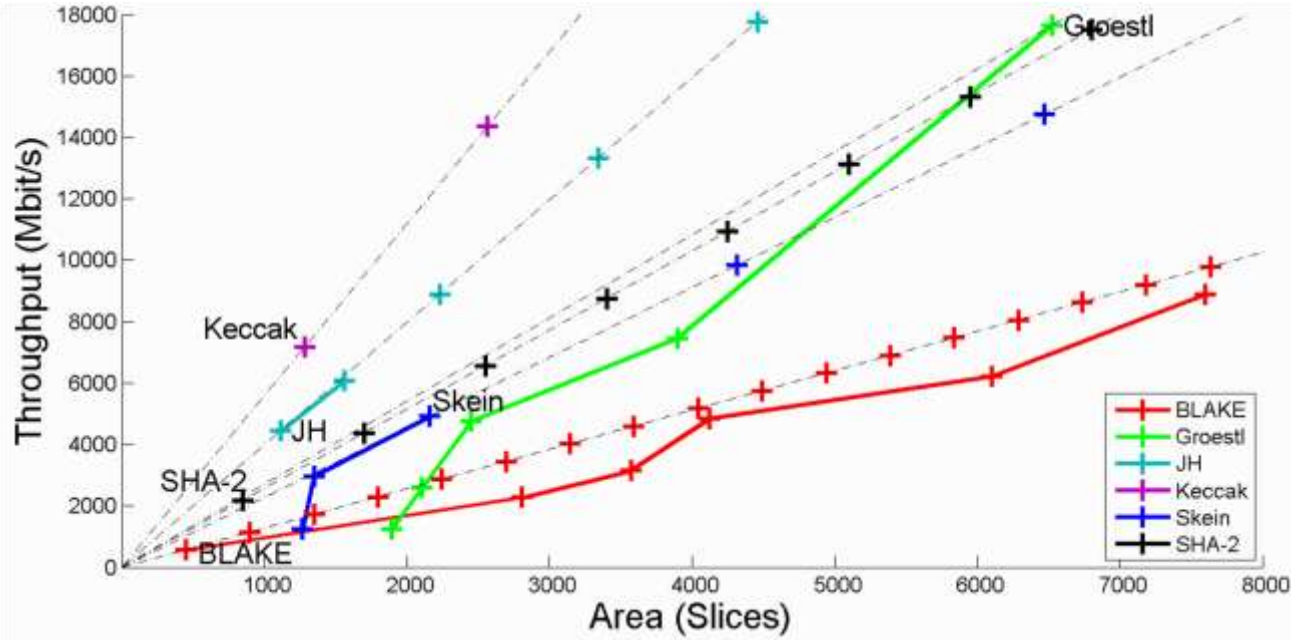


Рисунок 2.36 – Объединенная пропускная способность в области в комбинированной гистограмме для кратной аппаратной архитектуры 512-разрядных разновидностей BLAKE, Groestl, JH, Keccak, Skein и SHA 2 реализованных в Xilinx Virtex 5 FPGA.

Производительность для устройств Altera, показанных на рис. 7.6 и 7.7, несколько отличается. Для 256-разрядных версий алгоритмов Keccak безусловно превосходит по быстродействию всех оставшихся кандидатов и SHA 2. Лучшие реализации работы JH с такой скоростью, как реализации подобного размера SHA 2, но SHA 2 предлогает более тонкую детализацию, поскольку ее размер может быть увеличен в намного меньших инкрементах и пропускной способности, изменяются пропорционально. Groestl, BLAKE и Skein находятся в связи с каждым, с производительностью последовательно хуже, чем SHA 2.

Для 512-разрядных версий алгоритмов (см. рис. 7.7), Keccak и JH превосходят по быстродействию SHA 2, Skein находится в связи с SHA 2, Groestl и BLAKE отстают от текущей нормы. Численные результаты для всех наших реализаций получены в итоге в табл. 7.1 и 7.2. Оптимальные значения пропускной способности к соотношениям площадей и лучшей архитектуре для каждой хэш-функции перечислены полужирным в этих таблицах.

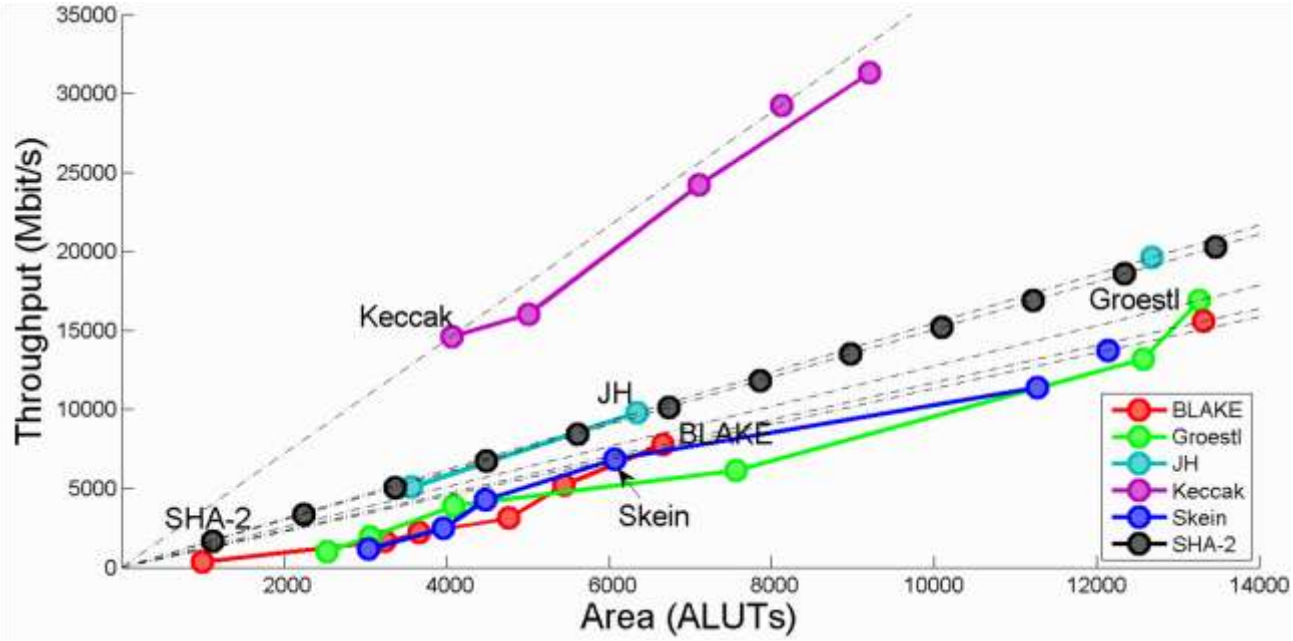


Рисунок 2.37 – Объединенная пропускная способность в области в комбинированной гистограмме для кратной аппаратной архитектуры 256-разрядных разновидностей BLAKE, Groestl, JH, Keccak, Мотка пряжи и SHA 2 реализованных в Altera Stratix III FPGAs.

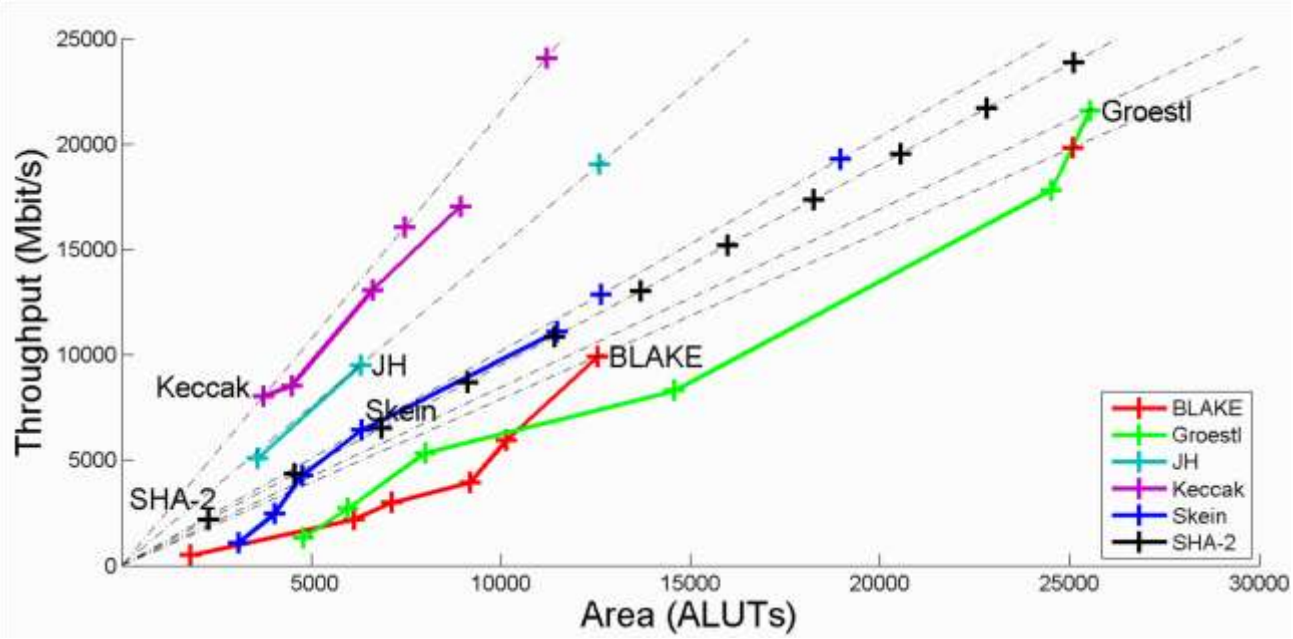


Рисунок 2.38 – Объединенная пропускная способность в области в комбинированной гистограмме для кратной аппаратной архитектуры 512-разрядных разновидностей BLAKE, Groestl, JH, Keccak, Мотка пряжи и SHA 2 реализованных в Altera Stratix III FPGAs.

Таблица 7.1 – Результаты для 256-разрядных разновидностей с дополнением модуля 3 раунда SHA 3 и SHA 2, реализованное

использование всей исследованной архитектуры и четырех семейств FPGA, Virtex 5 и Virtex 6 от Xilinx, и Stratix и IV Stratix от Altera. Нотация: Tp - пропускная способность, - область, Tp/A - Пропускная способность к Соотношению площадей. Оптимальные значения пропускной способности к соотношениям площадей и лучшей архитектуре для каждой хэш-функции перечислены полужирным.

Архитектура	**Virtex 5**			**Virtex 6**			**Stratix III**			**Stratix IV**		
	Tp	**A**	**Tp/A**	**Tp**	**A**	**Tp/A**	**Tp**	**A**	**Tp/A**	**Tp**	**A**	**Tp/A**
BLAK E-256												
/4(h)/4(v)---m	379	230	1.65	428	200	2.14	379	992	0.38	395	1022	0.39
/4(h)	1713	1493	1.15	1957	891	2.20	1665	3238	0.51	1691	3234	0.52
/2(h)	2266	1860	1.22	2363	1391	1.70	2206	3660	0.60	2316	3680	0.63
x1	2156	2032	1.06	2174	1505	1.44	2244	4807	0.47	2322	4802	0.48
/2(h)---PPL2	3510	2107	1.67	3260	1690	1.93	3118	4765	0.65	3421	4765	0.72
/2(h)---PPL4	4675	3228	1.45	4332	2268	1.91	5238	5442	0.96	5443	5449	1.00
x1---PPL2	4943	3099	1.59	4542	2040	2.23	4193	5619	0.75	4483	5642	0.79
x1---PPL4	7510	3526	**2.13**	8246	2609	**3.16**	7787	6657	**1.17**	7898	6657	**1.19**
Groestl -256												
/8(v) (P+Q)	1211	1191	1.02	1353	958	1.41	1248	3526	0.35	1172	3472	0.34
/4(v) (P+Q)	2486	1362	1.83	2901	1200	2.42	2533	4608	0.55	2391	4585	0.52
/2(v) (P+Q)	4508	1836	2.46	4915	1565	3.14	4685	7128	0.66	4597	7343	0.63

x1 (P+Q)	7081	2689	2.63	9187	2441	3.76	9760	11538	0.85	9181	11294	0.81
x1---PPL2 (P+Q)	13382	3172	**4.22**	11746	2968	3.96	13213	12572	1.05	12586	12570	1.00
x1---PPL4 (P+Q)	15015	4587	3.27	15624	4172	3.74	16903	13261	**1.27**	16126	13258	**1.22**
/8(v) (P/Q)	918	990	0.93	1105	750	1.47	1005	2526	0.40	966	2504	0.39
/4(v) (P/Q)	1920	1074	1.79	2099	811	2.59	1964	3061	0.64	1829	3059	0.60
/2(v) (P/Q)	3784	1302	2.91	4407	944	**4.67**	3925	4086	0.96	3644	4069	0.90
x1 (P/Q)	6572	2020	3.25	7071	1884	3.75	6140	7564	0.81	5640	7464	0.76
JH-2 56												
/2(v) (MEM)	2176	1139	1.91	1978	879	2.25	2124	3636	0.58	2086	3629	0.57
x1 (MEM)	4543	1001	**4.54**	5086	918	5.54	5024	3383	1.49	4815	3415	1.41
x2 (MEM)	5972	1473	4.05	6883	1381	4.98	6532	5830	1.12	6099	5793	1.05
x1---PPL2 (MEM)	3838	2147	1.79	4840	1486	3.26	5177	4280	1.21	5233	4278	1.22
x2---PPL2 (MEM)	7041	2099	3.35	8409	1781	4.72	9804	6339	**1.55**	9526	6331	**1.50**
x2---PPL4 (MEM)	8526	3085	2.76	7928	2424	3.27	9342	7962	1.17	9207	7960	1.16
/2(v) (OTF)	2054	1194	1.72	2206	1094	2.02	2114	3766	0.56	1981	3798	0.52
x1 (OTF)	4392	1080	4.07	5177	920	**5.63**	5091	3569	1.43	4807	3656	1.31
x2 (OTF)	6094	1695	3.60	7198	1604	4.49	6401	6372	1.00	5878	6360	0.92

Keccak -256												
x1	12745	1375	**9.27**	12451	1147	**10.86**	14624	4060	**3.60**	14009	4052	**3.46**
x1---PPL2	12523	2123	5.90	14942	1456	10.26	16047	5003	3.21	16878	5004	3.37
x2---PPL2	9610	2036	4.72	14444	2338	6.18	24242	7103	3.41	24942	7869	3.17
x2---PPL4	21717	3764	5.77	24644	2900	8.50	31296	9201	3.40	31691	9203	3.44
Skein256												
x1	1372	1145	1.20	1343	889	1.51	1152	3032	0.38	1269	3031	0.42
x4	3127	1245	**2.51**	2957	1026	2.88	2494	3960	0.63	2647	3970	0.67
x8	3168	1627	1.95	3548	1326	2.68	3193	5455	0.59	3336	5451	0.61
x4---PPL2	4873	2030	2.40	5679	1485	**3.82**	4280	4482	0.95	4688	4496	1.04
x4---PPL5	7077	3840	1.84	7325	2720	2.69	6869	6068	**1.13**	7824	6070	**1.29**
x8---PPL10	N/A	N/A	N/A	11118	5928	1.88	11390	11267	1.01	11485	11267	1.02
SHA256												
x1	1692	462	**3.66**	1665	305	**5.46**	1690	1122	**1.51**	1764	1114	**1.58**

Таблица 7.2 – Результаты для 512-разрядных разновидностей с дополнением модуля 3 раунда SHA 3 и SHA 2, реализованное использование всей исследованной архитектуры и четырех семейств FPGA, Virtex 5 и Virtex 6 от Xilinx, и Stratix и IV Stratix от Altera. Нотация: Тр - пропускная способность, - область, Тр/А - Пропускная способность к Соотношению площадей. Оптимальные значения пропускной способности к соотношениям площадей и лучшей архитектуре для каждой хэш-функции перечислены полужирным.

Архитектура	**Virtex 5**			**Virtex 6**			**Stratix III**			**Stratix IV**		
	Тр	**А**	**Тр/А**	**Тр**	**А**	**Тр/А**	**Тр**	**А**	**Тр/А**	**Тр**	**А**	**Тр/А**
BLAK E-512												
/4(h)/4(v)---m	575	449	**1.28**	625	350	1.79	495	1797	0.28	537	1791	0.30
/4(h)	2278	2805	0.81	2675	1639	1.63	2206	6124	0.36	2492	6399	0.39
/2(h)	3156	3569	0.88	3333	2608	1.28	3003	7115	0.42	3320	7114	0.47
x1	3046	3384	0.90	N/A	N/A	N/A	3010	9343	0.32	3387	9334	0.36
/2(h)---PPL2	4853	4123	1.18	4452	2882	1.54	3963	9188	0.43	4799	9177	0.52
/2(h)---PPL4	6225	6104	1.02	5412	3951	1.37	5938	10137	0.59	7668	10144	0.76
x1---PPL2	6296	5534	1.14	6430	4171	1.54	5705	10828	0.53	6356	10831	0.59
x1---PPL4	8882	7600	1.17	11030	5080	**2.17**	9908	12537	**0.79**	11048	12530	**0.88**
Groestl -512												
/8(v) (P+Q)	1602	2233	0.72	1683	1726	0.98	1697	6768	0.25	1589	6723	0.24
/4(v)	3147	2570	1.22	3078	1952	1.58	3562	9089	0.39	3323	8916	0.37

(P+Q)												
/2(v) (P+Q)	5413	3364	1.61	6051	3078	1.97	6642	14597	0.46	6175	14263	0.43
x1 (P+Q)	8781	5448	1.61	11608	5112	2.27	12976	22239	0.58	11903	21807	0.55
x1---PPL2 (P+Q)	17655	6525	**2.71**	16218	5740	**2.83**	17826	24518	0.73	16363	24612	0.66
x1---PPL4 (P+Q)	17918	8453	2.12	16033	7391	2.17	21595	25529	**0.85**	20357	25530	**0.80**
/8(v) (P/Q)	1238	1890	0.65	1271	1358	0.94	1317	4780	0.28	1296	4774	0.27
/4(v) (P/Q)	2602	2107	1.24	2684	1459	1.84	2710	5961	0.45	2586	5887	0.44
/2(v) (P/Q)	4750	2449	1.94	5040	1797	2.80	5352	7995	0.67	5023	7896	0.64
x1 (P/Q)	7462	3895	1.92	6843	3285	2.08	8310	14578	0.57	7882	14542	0.54
JH-5 12												
/2(v) (MEM)	2044	1145	1.79	2052	925	2.22	2124	3636	0.58	2086	3629	0.57
x1 (MEM)	4533	1125	**4.03**	4834	901	**5.37**	4309	3930	1.10	4201	3919	1.07
x2 (MEM)	6067	1561	3.89	6701	1417	4.73	6532	5830	1.12	6099	5793	1.05
x1---PPL2 (MEM)	3897	2244	1.74	4541	1589	2.86	5157	4537	1.14	5319	4536	1.17
x2---PPL2 (MEM)	8266	2223	3.72	8514	1977	4.31	9514	6297	**1.51**	9484	6303	**1.50**
x2---PPL4 (MEM)	6186	3645	1.70	8047	2744	2.93	9328	8084	1.15	9512	8079	1.18
/2(v) (OTF)	1985	1228	1.62	2131	1077	1.98	2114	3766	0.56	1981	3798	0.52
x1	4443	1114	3.99	4914	965	5.09	5091	3569	1.43	4807	3656	1.31

(OTF)												
x2 (OTF)	5940	1664	3.57	6732	1546	4.35	6401	6372	1.00	5878	6360	0.92
Keccak -512												
x1	7179	1283	**5.60**	7465	1052	**7.10**	8029	3734	**2.15**	7607	3723	**2.04**
x1---PPL2	7380	1774	4.16	8114	1263	6.42	8550	4484	1.91	8962	4481	2.00
x2---PPL2	7126	1996	3.57	N/A	N/A	N/A	13090	6617	1.98	12490	6580	1.90
x2---PPL4	13552	3428	3.95	13640	2550	5.35	17058	8934	1.91	17335	8934	1.94
Skein512												
x1	1258	1267	0.99	1446	987	1.47	1103	3086	0.36	1216	3088	0.39
x4	2972	1348	2.20	3141	1186	2.65	2493	4035	0.62	2597	4026	0.65
x8	2870	1556	1.84	3690	1454	2.54	3137	5527	0.57	3357	5536	0.61
x4---PPL2	4916	2157	**2.28**	5713	1567	**3.65**	4292	4756	0.90	4758	4767	1.00
x4---PPL5	5829	4377	1.33	7337	3160	2.32	6428	6319	**1.02**	7123	6324	**1.13**
x8---PPL10	5946	9032	0.66	9130	6544	1.40	11111	11485	0.97	10542	11486	0.92
SHA512												
x1	2189	850	**2.58**	2357	548	**4.30**	2171	2282	**0.95**	2409	2313	**1.04**

ВЫВОДЫ

Теоретическое и практическое обоснование криптографических методов и механизмов симметричных криптопреобразований, функций хэширования, реализация криптографических преобразований и криптографических протоколов на основе применения аппаратно-программных средств КЗИ позволило достичь обеспечения уровней гарантий согласно принятой политике безопасности, совершенствование и разработки перспективных стандартов КЗИ.

В результате выполнения работы было установлено, что хэш-функция Keccak является таким же универсальным криптопримитивом, как и Skein, за исключением возможности использования в режиме блочного шифра. Режим блочного шифра остаётся незаменимым в прозрачном шифровании данных на дисках компьютеров, но в этой области нет существенного ограничения по ресурсам, поэтому для шифрования можно использовать отдельный блочный шифр в соответствующем режиме. Для решения многих других задач Keccak также универсален, как и Skein, а фактически его область применения может быть шире. При необходимости этот алгоритм может быть внедрён как в миниатюрные устройства с ограниченными ресурсами, так и на высокопроизводительные серверы, работающие с большим объёмом соединений. Оба алгоритма имеют самую сильную доказательную базу среди всех финалистов, но по некоторым оценкам, Keccak имеет более строгие доказательства в модели RO. В частности он не подвержен атакам на функции конструкции Narrow Pipe.

Алгоритмы Keccak и Skein оказывают влияние на практическое использование симметричных алгоритмов и смешанных протоколов, а также и на теоретические исследования в различных областях криптографии). В целом хэш-функция Skein на наиболее распространённых процессорах

работает в два раза быстрее, чем SHA-512, а блочный шифр Threefish в два раза быстрее, чем AES.

KeccaK обладает множеством качеств, включая элегантный дизайн и возможность эффективной реализации на большом количестве аппаратных платформ. Ясная конструкция KeccaK облегчает проведение его анализа. Алгоритм удачно показывает себя в специализированном исполнении на аппаратных платформах, превосходя в этом плане всех других финалистов и алгоритмы SHA-2. KeccaK имеет преимущество в том, что атаки, рассчитанные против SHA-2 не действуют против него, поскольку эти два алгоритма устроены на совершенно разных принципах.

Новые полезные свойства Keccak найдут применение спустя годы, после его принятия. Так, он может быть использован в миниатюрных встраиваемых устройствах, которые не являются полноценными устройствами.

Можно использовать KeccaK в качестве хэш-функции с произвольным размером выхода, в качестве потокового шифра, в качестве функции выработки ключей из пароля, в качестве кодов аутентификации сообщений, в качестве криптостойкого генератора псевдослучайных чисел с подкачкой энтропии из внешнего источника и с затиранием внутреннего состояния, в качестве возможного нового режима дополнения цифровых подписей.

Также был представлен дуплексный режим работы KeccaK — потоковое шифрование с аутентификацией за один проход, что может представлять альтернативу использовавшимся ранее блочным шифрам в шифровании пакетной связи.

Все эти универсальные возможности не являются отдельными нагромождаемыми модулями, как это пытались реализовать в других алгоритмах. Смена режимов использования также не требует каких-либо переключателей предопределённого формата и никак не ухудшает простоту конструкции.

Алгоритм BLAKE оказался наиболее производительным, затрачивая почти на 4 микросекунды больше чем SHA-2, чтобы произвести хэширование. Skein показал второй результат, что приблизительно в три раза медленнее чем BLAKE. Самым медленным из группы был JH. Потребовалась почти миллисекунда на байт сообщения, чтобы произвести хэширование. По итогам побитового теста все пять SHA 3 алгоритмов отклоняются от идеального распределения 64 1s и 0s. У BLAKE самое большое отклонение на левую половину на шесть 1s больше чем идеал. У Keccak самое большое отклонение правой половины, Grøstl и JH показали наименьшее отклонение.

В ходе тестирования алгоритмы BLAKE и Grøstl показали возможность возникновения коллизии. У BLAK – 3556 уникальных хэш-функций 3556 на коллизию, у алгоритма Grøstl приблизительно 2667 уникальных хэш-функций на коллизию.

Алгоритм JH-показал средние значения во всех тестах, но потребовалась почти миллисекунда на бит сообщения, чтобы произвести хэширование. JH по результатам теста времени оказался на последнем месте.

В тесте на лавинные свойства кандидаты SHA-3 показали среднее изменение сообщения 17 бит, немного больше чем минимум 16.

Частотный побитовый тест выявил, что у алгоритма BLAKE самое большое отклонение на левую половину на шесть 1s больше чем идеал. У Keccak самое большое отклонение правой половины, Grøstl и JH показали наименьшие отклонения.

Алгоритм показал средние результаты Keccak по всем тестам. SHA-3 хэш-алгоритмы проявили не плохие показатели устойчивости, производительности и безопасности. Однако в связи с недостаточной оптимизацией данные алгоритмы уступают более ранним версиям построенным на встроенных библиотеках.

Результаты оценки на платформах ASIC и FPGA выявили, что алгоритм Keccak показывает высокие показатели производительности как на FPGA так и на ASIC из-за небольшого размера программного кода и высокой

пропускной способности. Grøstl показал хорошие результаты на платформе ASIC с высокой пропускной способностью, но намного худшие результаты в FPGA из-за объёмной реализации функции. JH и как Skein выполняется умеренно хорошо в FPGA и ASIC в то время как хэш-функция BLAKE на всех системах замыкает список алгоритмов.

На ASIC, BLAKE является наименьшим по объёму SHA-3 кандидатом, однако он является вторым по объёму реализации на FPGA.

Результаты остальных кандидатов остаются относительно стабильными на двух платформах.

Данное исследование показало, что Keccak является наиболее энерго-эффективными при достаточных показателях быстродействия на большинстве платформ ASIC и FPGA.

Самыми энегро не обоснованными на платформе FPGA являются алгоритмы BLAKE и JH. В тоже время на платформе ASIC Grøstl является наиболее затратным.

Криптоалгоритм Grøstl показывает не плохие показатели скорости, однако потребление энергии компенсирует данный факт. Остальные алгоритмы потребляют не много энергии, но и максимальная пропускная способность находится на низком уровне.

3 ДИФФЕРЕНЦИАЛЬНЫЕ И ЛИНЕЙНЫЕ СВОЙСТВА ШИФРОВ

3.1 ОБ УЧАСТИИ S-БЛОКОВ В ФОРМИРОВАНИИ МАКСИМАЛЬНЫХ ЗНАЧЕНИЙ ПОЛНЫХ ДИФФЕРЕНИАЛОВ И СМЕЩЕНИЙ ЛИНЕЙНЫХ КОРПУСОВ БЛОЧНЫХ СИММЕТРИЧНЫХ ШИФРОВ

В центре внимания этого раздела будут находиться малые версии шифров. Именно на малых моделях шифров впервые удалось поверить ряд положений и теоретических подходов, развиваемых в публикациях, касающихся формированию оценок доказуемой безопасности больших шифров, не подтверждённых практикой, ввиду непреодолимых вычислительных ограничений, присущих полноразмерным конструкциям.

Более обстоятельному обсуждению вопроса адекватности малых моделей и их больших прототипов будет посвящён отдельный раздел. А сейчас пока можно заметить, что свойство перехода максимальных значений дифференциальных и линейных показателей шифров после определенного числа циклов зашифрования (для Rijndael-я после четырех) к некоторому установившемуся значению отмечается на уровне предположений в ряде публикаций и для больших шифров (например, в [208] для шифра Rijndael). Именно на обосновании этого важного момента сосредотачивается внимание настоящего раздела.

Принципиальным для дальнейшего является признание того положения, что концепция оценки показателей стойкости БСШ, разрабатываемая в представленном во втором разделе обзоре публикаций, а также во многих других работах, строящаяся на привязке показателей стойкости шифров к свойствам применяемых в них S-блоков, является, ошибочной. На самом деле результирующие (т.е. получающиеся при

использовании полного набора цикловых преобразований) показатели стойкости шифров определяются практически только размером битового входа в шифр и от свойств S-блоков не зависят. Свойства S-блоков сказываются (и то не в существенной степени и не во всех случаях) только на динамике перехода к показателям случайной подстановки. Это факт зафиксирован в разделе 3 в виде утверждения.

Утверждение. *Для каждого блочного симметричного шифра* (*из числа известных итеративных БСШ*) *существует вполне определенное число циклов, после которого шифр приобретает свойства случайной подстановки. Дальнейшее наращивание числа циклов не влияет на итоговые дифференциальные* (*по операции XOR*) *и линейные свойства шифра. Это значение является одним и тем же для всех шифрующих преобразований с одинаковым битовым размером входа* (*независимо от способа введения в шифр цикловых подключей*).

Конечно, аккуратнее было бы говорить только об уже проверенных шифрах, но на наш взгляд, это свойство должно быть присуще любому сколько-нибудь внушающему доверие шифру.

Представляется уместным начать изложение нашей позиции в рассматриваемом вопросе с предметного анализа концепции определения стойкости блочных симметричных шифров, развиваемой в криптографической литературе, чему и посвящается ближайший подраздел монографии.

3.1.1 Сущность развиваемой в криптографической литературе концепции определения показателей доказуемой стойкости БСШ

Сначала изложим более детально понятийный аппарат и позицию авторов работы [130], которая отмечена последней во втором разделе при анализе существующих подходов к оценке показателей стойкости блочных

симметричных шифров к атакам дифференциального и линейного криптоанализа.

В этой работе рассматриваются вложенные SPN структуры, обобщающие метод широкого следа, использованный в шифре Rijndael, для построения доказуемо безопасной цикловой функции. Внимание авторов сосредотачивается, как было уже отмечено ранее, на получении оценок для верхних границ максимумов средних значений вероятностей дифференциалов и линейных корпусов (*MADP* и *MALHP*) SPN блочных шифров с оптимальными или квазиоптимальными диффузионными слоями.

В основе рассматриваемых SPN структур лежит конструкция, названная авторами *SDS* функцией. Она представлена на рис. 3.1.

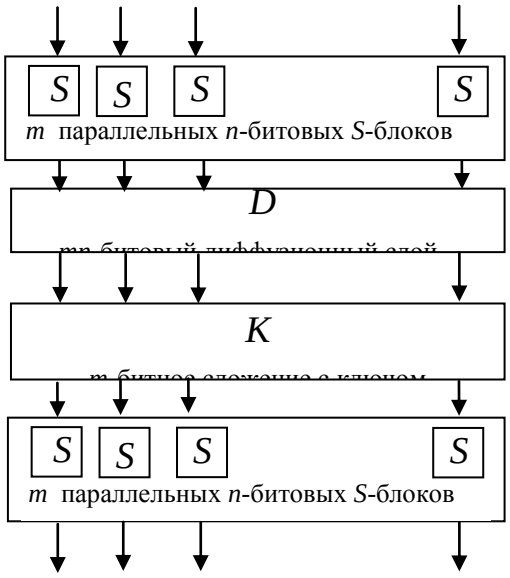


Рисунок 3.1 – *SDS* функция

В этой конструкции все S-блоки считаются одинаковыми, и их фундаментальные дифференциальные и линейные вероятности p и q (максимальные зна-чения вероятностей переходов таблиц XOR разностей и смещений таблиц линейных аппроксимаций) соответственно равны $p = DP^{s}_{\max}$, $q = DL^{s}_{\max}$.

Для описания дифференциальных и линейных свойств *SDS* преобразования используется понятие числа ветвлений, с помощью которого

определяется эффективность распространения активизации (лавинного изменения битовых значений) в процессе прохождения диффузионного слоя *D*. Само линейное преобразование при этом задается с помощью умножения входного блока данных на $mn \times mn$ матрицу линейного преобразования *D* с элементами из поля $GF(2)$.

Определение 1 (Число ветвлений). *Дифференциальное число ветвлений* $ß_D(D)$ *для диффузионного слоя D определяется как:*

$$ß_D(D) = \min_{\Delta x \neq 0} (Hw(\Delta x) + Hw(D\Delta x)),$$

где Δx *– входная разность.*

Отметим, что понятие Хеммингова веса $Hw(x)$ здесь применяется не для числа ненулевых битов, а сразу для числа *m* ненулевых *n*-битных ″характеров″ (*m*-сегментов из *n*-битных слов) mn - битного входного блока данных.

Аналогично определяется линейное число ветвлений $ß_L(D)$ для диффузионного слоя *D* как

$$ß_L(D) = \min_{Гу \neq 0} (Hw(D^* Гу) + Hw(Гу)),$$

где $Гу$ *– выходное масковое значение, а* D^* *– транспонированная матрица линейного преобразования D.*

Практически значения $ß_D(D)$ и $ß_L(D)$ являются нижней границей для числа активных S-блоков для двух смежных циклов дифференциальной и линейной характеристики соответственно.

Когда оба числа ветвлений имеют своим максимумом значения *m*+1, то диффузионный слой называется MDS, так как такой диффузионный слой может быть создан на основе сепарабельного кода с максимальным расстоянием.

Остается теперь изложить теорему Хонга, доказательство которой приведено и в [130]. Эта теорема достаточно полно отражает сущность концепции, о которой идет речь.

Теорема 1. *Если диффузионный слой D есть m×m матрица, чьи элементы принадлежат полю* $GF(2^n)$ *и если D есть* MDS *функция* (*порождающая матрица сепарабельного кода с максимальным расстоянием*) *для n-битного слова с числами ветвлений* $ß_D = ß_L = m + 1$, *то*

$$MADP^{SDS} \le p^m,\ MALHP^{SDS} \le q^m.$$

Для шифра Rijndael (128-битной версии), рассматриваемого как вложенная SPN структура (состоящая из *N* структур *SDS*) с числами ветвлений нижнего и верхнего уровня равными соответственно $n = m = 4$, для $p = q = 2^{-6}$ авторы рассматриваемой работы предлагают считающийся доказанным результат:

$$MADP^{NSDS} \le [illegible]^{4\times 4} = 2^{-96},$$

$$MALHP^{NSDS} \le [illegible]^{4\times 4} = 2^{-96}.$$

Близкую к изложенной можно найти точку зрения и во многих других работах [125-129 и др.], правда, результаты могут отличаться от представленных в значительных пределах.

Несмотря на то, что представленные оценки, получены на основе внешне строгих математических рассуждений, однако как показывает анализ, их скорее следует рассматривать как удачная подгонка, так как практические результаты по определению показателей стойкости шифров свидетельствуют о том, что приведенные показатели от свойств используемых в шифрах S-блоков (максимальных значений переходов таблиц XOR разностей и смещений таблиц линейных аппроксимаций S-блоков) практически не зависят. Тут нельзя не привести высказывание о том, что любая теория чего-нибудь стоит лишь тогда, когда её подтверждает практика!

Возможность практически проверить многие теоретические результаты удалось впервые на основе использования уменьшенных (малых) моделей шифров, являющихся основой развиваемого подхода. Именно с помощью экспериментов с уменьшенными моделями были получены основные

содержательные результаты, подтверждающие новую идеологию оценки показателей доказуемой стойкости шифров и обладающие высокой степенью доверия, не достижимой для всех известных до последнего времени методов и подходов.

Задачей этого раздела и является изложение результатов вычислительных экспериментов с малыми моделями шифров, позволяющими определить действительное состояние дел в этом важном направлении современной криптологии. Как уже было отмечено ранее, вместо *MADP* и *MALHP* в этом разделе будут вычисляться *AMDP* и *AMLHP*, как являющиеся более адекватными решаемым задачам.

Очевидно, что одним из важных этапов реализации предлагаемой методологии является разработка самих малых моделей шифров, с помощью которых получены все последующие результаты. Напомним кратко выполненную работу в этом направлении исследований.

3.1.2 О криптографической значимости S-блоков в современных блочных шифрах

В этом подразделе излагается содержание работы [209], специально подготовленной на эту тему. Здесь речь будет идти не о дифференциальных и линейных показателях малых моделей конкретных шифров (мы на них остановимся позднее), а в более общем плане о влиянии S-блоковых конструкций на дифференциальные и линейные свойства двух обобщённых классов шифров (на уровне малых моделей), которые отличаются эффективностью линейного циклового преобразования (шифров с сильными и шифров со слабыми цикловыми линейными преобразованиями). Мы хотим здесь ответить на вопрос, насколько актуальной является задача поиска S-блоков с особыми криптографическими свойствами?

1.2.1 Краткий анализ результатов исследования подстановочных конструкций в интересах криптографических применений

В работе [159], посвященной методам формирования S-блоковых конструкций случайного типа с улучшенными криптографическими показателями, достаточно детально проанализировано большое число публикаций в этом направлении. Воспользуемся выводами, которые сводятся к следующим утверждениям:

1. Существующие на сегодня подходы и методы конструирования S-блоков ориентированы в первую очередь на обеспечение минимальных значений максимумов таблиц XOR разностей DP_{max} и таблиц линейных аппроксимаций LP_{max}. И в этом направлении удалось добиться существенных успехов. Реализованы S-блоковые конструкции с предельными (теоретически минимально возможными) значениями показателей LP_{max} и DP_{max}.

2. Имеется основательно развитый аппарат оценки криптографических показателей (свойств) булевых функций, с помощью композиции которых можно описать преобразования осуществляемые S-блоками. Определены подходы и правила, с помощью которых результирующие криптографические показатели отдельных булевых функций, входящих в S-блок, можно пересчитать в показатели всего преобразования в целом (S-блока). И хотя в литературе уделяется очень большое внимание развитию и применению для оценки криптографических показателей S-блоков математического аппарата булевых функций, тем не менее, этот алгебраический подход для применяемых в шифрах конструкций S-блоков не является определяющим. Более того, использованные в современных шифрах S-блоковые конструкции обладают далеко не лучшими, а по ряду показателей и весьма низкими криптографическими свойствами входящих в них булевых функций.

3. Представленные результаты свидетельствуют о том, что хорошие S-блоки, как правило, можно отнести к множеству случайных подстановок и, по-видимому, проверку случайности можно было бы включить в процедуры отбора подстановок с хорошими криптографическими свойствами, однако порождать S-блоки порядка 256 и более с высокими показателями δ-

равномерности, как показывает анализ, является вычислительно очень сложной задачей (для этого требуется практически не реализуемые вычислительные мощности). Именно поэтому все реальные разработки по построению больших (байтовых) S-блоков основываются на методах, которые скорее можно назвать регулярными. Так в [159] отмечается, что более прогрессивным выглядит использование для построения S-блоков отдельных предложений, имеющихся в публикациях, в частности, предложений (соображений) К. Ниберг [124]. Именно они нашли дальнейшее развитие и практическое применение в конструкциях S-блоков, построенных в процессе создания многих новых блочных симметричных шифров (Rijndael, Camellia, Лабиринт, ADE и др.).

Мы отметим здесь работы [1, 39, 57, 210, 211 и др.], в которых обосновывается положение, состоящее в том, что показатели стойкости современных блочных симметричных шифров к атакам линейного и дифференциального криптоанализа от свойств S-блоков (за исключением вырожденных конструкций) не зависят.

Большинство работ этого направления связано с исследованием уменьшенных до 16-битного входа моделей больших шифров, в которых, как правило, используются 4-х битные (по входу и выходу) S-блоки. Именно использование уменьшенных моделей шифров стало основой предложенного нового подхода (новой идеологии) к оценке показателей стойкости итеративных шифров [1].

В отмеченных работах показатели стойкости шифров оцениваются как раз с помощью средних значений дифференциальной и линейной вероятностей (*AMDP* и *AMLHP*).

Заметим далее, что 4-х битные конструкции S-блоков используются и при конструировании современных блочных симметричных шифров. Достаточно назвать хотя бы шифр Serpent, занявший почётное второе место на конкурсе AES.

Конечно же, изучению криптографических показателей полубайтовых S-блоков посвящено немало работ. Мы хотим далее обратить внимание на две публикации, посвященные исследованию полубайтовых S-блоков, которые появились в последнее время.

В работе [212] представлено исчерпывающее исследование всех 16! биективных 4-х битных S-блоков. Отмечается, что в представленной ранее в 2007-ом году работе Leander-a и Poschmann-a была исчерпывающе проанализирована аффинная эквивалентность классов. В своей работе авторы представляют результаты исследований дальнейших свойств оптимальных классов S-блоковой линейной эквивалентности. В своём анализе они полагают, что два S-блока являются криптографическим эквивалентом, если они изоморфны вплоть до перестановки входных и выходных битов и XOR-а с константой на входе и выходе. Перечисляются все такие эквивалентные классы с указанием их дифференциальных и линейных свойств, и отмечается, что эти классы эквивалентны не только по дифференциальным и линейным границам, но и имеют эквивалентные алгебраические свойства число ветвлений и схемную сложность. Авторы в своей работе описывают “золотое” множество S-блоков, которые, как они считают, имеют идеальные криптографические свойства.

Во второй работе [213] рассматриваются квадратичные аппроксимации (булевых функций) специального вида и возможность применения их в нелинейном криптоанализе блочных шифров. Для четырехразрядных подстановок, рекомендованных для использования в S-блоках алгоритмов ГОСТ 28147-89, DES, и s^3DES, показано, что почти во всех случаях существуют более вероятные (по сравнению с линейными) квадратичные соотношения специального вида на входные и выходные биты этих подстановок. Отмечается, что среди большинства рассмотренных S-блоков, рекомендуемых для использования в шифрах, существуют линейные соотношения относительно битов шифртекста, что по существу может использоваться при решении систем линейных уравнений, возникающих при

анализе шифров. Нас, однако, в этой работе будут интересовать не слабости S-блоковых конструкций, а скорее их криптографические свойства в целом. В этом отношении можно заключить, что полубайтовые S-блоки являются изученными наиболее полно и глубоко.

Следующий вопрос, касающийся решения поставленной в работе задачи, связан с обоснованием методики выполнения исследований.

3.2 Методика исследования дифференциальных и линейных свойств шифров

Внимание этого раздела сосредотачивается именно на полубайтовых S-блоках, с использованием которых построены все уменьшенные модели шифров (кроме шифра ГОСТ), и которые, как уже было отмечено выше, являются изученными наиболее глубоко. Здесь мы отходим от общепринятых подходов к оценке криптографических показателей S-блоков и предлагаем *новый подход* к определению степени их криптографической пригодности. Этот подход строится на оценке числа циклов преобразования (шифрования), которые необходимы шифру для достижения стационарного состояния характерного для случайной подстановки. Это стационарное состояние будет определяться моментом достижения шифром устойчивого минимального значения максимума таблицы XOR разностей для всего шифра (полного дифференциала) и минимального устойчивого значения максимума смещения таблицы линейных аппроксимаций (линейной оболочки), совпадающих, как показали исследования, с соответствующими показателями случайной подстановки степени 2^{16}.

В центре внимания будут 16-битные по входу и выходу модели шифров. Сам момент прихода шифра к стационарному состоянию будет определяться числом циклов, начиная с которого среднее по ключам значение максимума таблицы XOR разностей становится, как будет видно из дальнейшего, меньше 20-ти (теоретическое значение максимума

дифференциальной таблицы случайной подстановки степени 2^{16} равно 19,5 (см. раздел 3, а также [43]). Для максимума смещения таблицы линейных аппроксимаций стационарное значение будет определяться моментом (числом циклов), начиная с которого измеренный максимум смещения таблицы линейных аппроксимаций становится меньше 900 (теоретическое значение максимума смещения таблицы линейных аппроксимаций случайной подстановки степени 2^{16} равно 750 (см. раздел 3, а также [44]), т.е. экспериментальное значение будет браться несколько больше теоретического).

Объясним выбор таких значений

Очевидно, что формируемые оценки *AMDP* и *AMLHP* (см. раздел 4) являются случайными и совсем не равны точно значениям, следующим из теоретических распределений свойственных случайной подстановке (соответственно 19,5 и 750).

Как же обоснованно определить момент прихода шифра к стационарному состоянию?

Ответ на поставленный вопрос можно получить, если воспользоваться методом доверительных интервалов, являющимся методом математической статистики, специально предназначенным для построения множества приближенных значений неизвестных параметров вероятностных распределений [171]. Мы уже освещали сущность этого подхода при изучении лавинных показателей шифра ГОСТ [214] и в подразделе 2 при оценке близости теоретических законов распределения переходов XOR таблиц случайных подстановок и смещений таблиц линейных аппроксимация случайных подстановок экспериментальным.

В соответствии с изложенным в [171] подходом, задаваясь доверительной вероятностью $P_c(\theta_1, \theta_2) = 0{,}999 \rightarrow \alpha = 0{,}001$, на основании таблицы распределения Стьюдента [171] для заданных значений α, и n = 30 (объём выборки ключей в наших опытах), а также значения дисперсии

распределения дифференциальных переходов XOR таблицы случайной подстановки в виде пуассоновского закона [171]:

$$\Pr[\Lambda(\Delta X, \Delta Y) = 2i] \approx Poisson\left(i; \frac{1}{2}\right) = \frac{e^{-\frac{1}{2}}}{i!2^i}$$

с параметром $\lambda = \frac{1}{2}$, получим:

$$\frac{t \cdot S}{\sqrt{n}} = \frac{3{,}646 \cdot \sqrt{0{,}5}}{\sqrt{30}} = 0{,}941,$$

и, следовательно, можно считать попавшими в доверительный интервал все значения максимумов числа переходов $\Lambda(\Delta X, \Delta Y)_{\max}$ входной разности ΔX в выходную разность ΔY, удовлетворяющие условиям

$$19 - 0{,}941 \le \Lambda(\Delta X, \Delta Y)_{\max} \le 19 + 0{,}941.$$

Мы в наших экспериментах попадание в доверительный интервал будем фиксировать с одной стороны в виде выполнения неравенства $\Lambda(\Delta X, \Delta Y)_{\max} \le 20$.

При оценке линейных свойств шифров (мини-версий) за основу будет взята аппроксимация закона смещений таблиц линейных аппроксимаций в виде нормального закона [178]:

$$\Pr[\Lambda(\alpha, \beta) = 2x] \approx Z\left(\frac{x}{2^{(n-4)/2}}\right).$$

Среднеквадратическое значение для этого закона распределения равно $2^{(n-4)/2}$ и для подстановки степени 2^{16} имеем $2^{(16-4)/2} = 2^6$. В этом случае алгоритм вычисления таблиц линейных аппроксимаций оказывается намного медленнее (для построения таблицы нужно выполнить 2^{48} операций). Поэтому далее будут представлены эксперименты, выполненные для одного ключа зашифрования (и уже имеющиеся результаты для большего числа ключей). При определении доверительного интервала в этом случае, за основу были взяты расчёты, выполненные для выборки из 10-ти ключей зашифрования. Для значений доверительной вероятности, использованных

ранее, значение параметра t таблицы распределения Стьюдента здесь равно $4{,}587$, что приводит к результату:

$$\frac{t \cdot S}{\sqrt{n}} = \frac{4{,}587 \cdot 64}{\sqrt{10}} = 92{,}83,$$

и, следовательно, доверительный интервал определяется граничными значениями:

$$750 - 93 \le \Lambda(\alpha, \beta)_{\max} \le 750 + 93.$$

Данные реальных измерений для малых шифров, как мы увидим, будут давать несколько завышенные средние значения максимума смещения по сравнению с теоретическим, поэтому отсчет попадания в доверительный интервал и в этом случае предлагается вести односторонней границей в виде проверки выполнения несколько завышенного неравенства $\Lambda(\alpha, \beta)_{\max} \le 900$.

Важно то, что эти показатели можно проверить (оценить) на основе применения малых моделей шифров, использование которых, как было отмечено выше, стало основой реализации новой методологии оценки показателей доказуемой стойкости БСШ [204].

В дальнейшем будет рассмотрено две модели (два типа) шифров. Первая модель будет демонстрировать (представлять) шифр со слабым линейным преобразованием (коэффициент ветвления в среднем равен 3), а вторая – с сильным линейным преобразованием (коэффициент ветвления 5).

В качестве универсальной модели шифров со слабым линейным преобразованием в работе выбран 16-битный шифр, предложенный в работе профессора Хеуса [215].

Это шифр с подстановочно-перестановочной структурой (SPN), представлен на Рис. 2.1 (заметим, что к шифрам со слабым линейным преобразованием мы относим и Фейстель-подобные шифры DES и ГОСТ).

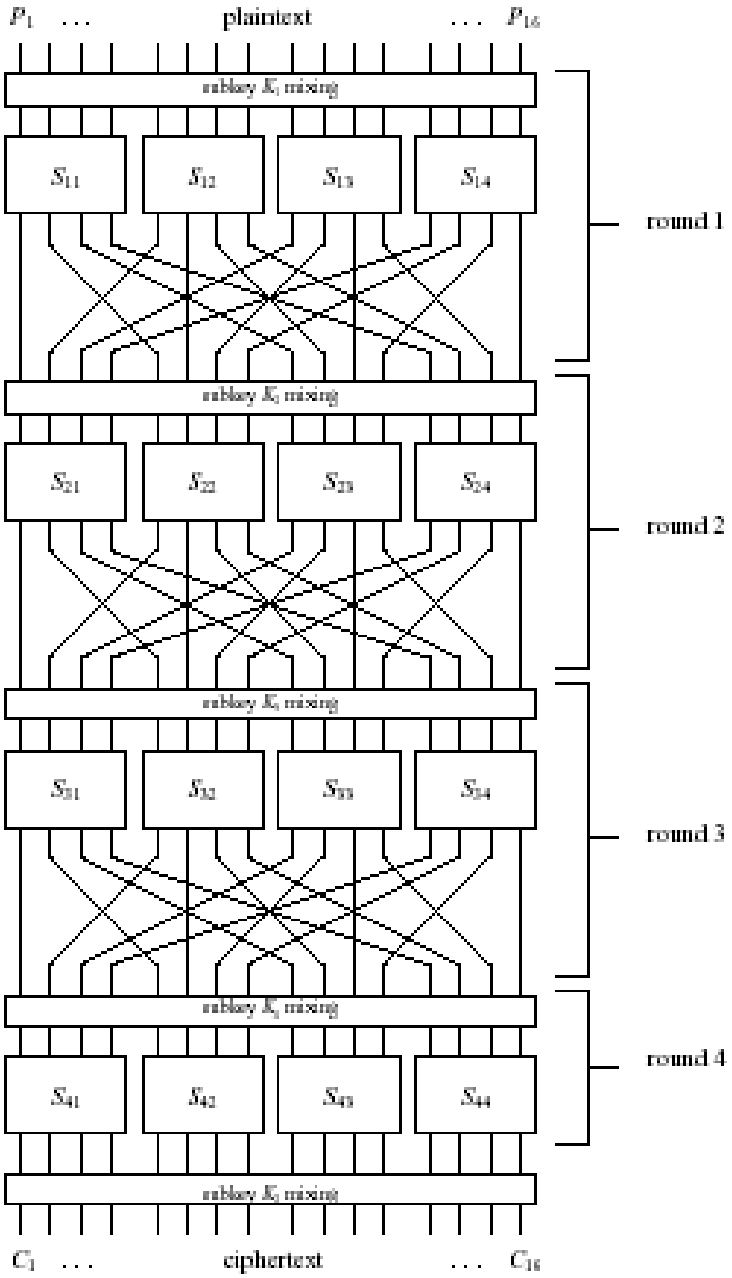


Рисунок 3.2 – Шифр на основе подстановочно-перестановочной (SPN) структуры

Выполняемые в этом шифре операции во многом аналогичны тем, которые используются в шифре DES. По этой же схеме построены и многие современные шифры, включая Rijndael.

Как следует из схемы Рис. 2.1, на вход алгоритма поступает 16-битный блок входного текста. Этот блок обрабатывается с помощью повторения четырех циклов, состоящих из простейших операций замены, перестановки битов и сложения с ключом.

В алгоритме в каждом отдельном цикле преобразований входной 16-битный блок данных делится на четыре подблока, каждый из которых поступает на соответствующие входы блоков замены (S-блоков, осуществляющих замену четырех входных битов на четыре выходных).

Линейное преобразование в каждом цикле осуществляет простую перестановку битов 4-х битных выходов блоков замены (слабое линейное преобразование). Оно представлено в виде перестановки, приведенной в табл. 2.1.

Таблица 3.1

Перестановка битов в шифре Хеуса

Вход	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Выход	0	4	8	C	1	5	9	D	2	6	A	e	3	7	B	F

Для сложения с ключом (подключом) используется простая операция побитного сложения по модулю 2 (XOR). Кроме цикловых подключей в алгоритме осуществляется дополнительное сложение битов ключа с выходными битами последнего цикла. Это традиционно делается для того, чтобы затруднить анализ алгоритма дешифрования. Обычно в используемых алгоритмах шифрования подключ для каждого цикла преобразований извлекается из главного ключа (мастер ключа). Во всех экспериментах биты подключей генерируются независимо и не связаны друг с другом (хотя, как показывают опыты, это не принципиально).

В качестве шифра с сильным линейным преобразованием будет выступать уменьшенная до 16-битного входа конструкция шифра Rijndael. В этом случае во всех экспериментах в цикловых преобразованиях шифра будут использоваться четыре одинаковых полубайтовых подстановки (S-блока), выходы которых (набор из четырех полубайтовых значений выходов четырех S-блоков $B = (b_0, b_1, b_2, b_3)$) будут обрабатываться с помощью операции MixColumns на весь текст. Результатом линейного преобразования в этом случае является 16-ти битный вектор $C = (c_0, c_1, c_2, c_3)$, определяемый с помощью матричного умножения. Здесь используется матрица МДР кода (кода с максимальным

кодовым расстоянием). Идеи построения таких матриц хорошо изложены в работе [51]. Математически это преобразование описывается в виде:

$$(c_0, c_1, c_2, c_3) = b_0, \quad b_1, \quad b_2, \quad b_3 \; * \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix},$$

где операция матричного умножения выполняется в поле $GF(2^4)$ (элементы матрицы − это элементы поля $GF(2^4)$). При такой конструкции линейного преобразования операции ShiftRows не требуется. Практически последние конструкции шифров можно получить их шифра Хеуса заменой его линейного преобразования матричным [192].

1.2.3 Результаты вычислительных экспериментов по использованию в шифрах разных конструкций S-блоков

Первая серия экспериментов была выполнена с использованием SPN шифра из работы Хеуса, представленного на Рис.2.1. Наши изменения основного предложения [183] заключались в том, что мы сделали в этом шифре переменное число цикловых преобразований (в разработке [183] реализовано 4-х цикловое преобразование), и самое главное применили в этом шифре подстановки различного типа, взятые из упомянутых выше работ [212, 213].

Таблица 5.2 иллюстрирует результаты экспериментов, выполненных с использованием в шифре Хеуса подстановок из списка, представленного в работе [213]. В него вошли подстановки из книги А.Г. Ростовцева и Е.Б. Маховенко [216], в которой приведена серия экстремальных четырехразрядных подстановок $S^1,\ldots,S^{10}$, рекомендованных для S-блоков стандарта ГОСТ 28147-89 (в таблице 2 эти подстановки представлены порядковыми номерами 1-10). В [213] отмечается, что из каждой такой подстановки путем умножения ее на аффинные подстановки можно получить целый класс экстремальных подстановок. Все они были выбраны так, говорят

авторы работы [213], чтобы максимально повысить стойкость шифра к методам линейного и дифференциального криптоанализа.

Таблица 3.2

Поцикловые значения максимумов полных дифференциалов (XOR таблиц) шифра Хеуса с различными наборами S-блоков, взятых из работы [213].

№ п/п	Подстановки из работы [213]	1	2	3	4	5	6	7	8	9	10
1	0,D,B,8,3,6,4,1,F,2,5,E,A,C,9,7	16384	4096	523,07	69,67	30,60	**19,20**	19,13	19,27	19,47	19,33
2	0,1,9,E,D,B,7,6,F,2,C,5,A,4,3,8	16384	4096	1828,80	383,73	114,53	32,93	20,40	**19,60**	18,87	19,13
3	0,1,D,B,9,E,6,7,C,5,8,3,F,2,4,A	16384	4096	1821,73	426,13	150,40	49,40	20,73	**19,13**	18,87	19,13
4	0,1,2,4,3,5,8,A,7,9,6,D,B,E,C,F	16384	4096	2671,47	1009,8	394,87	145,87	68,40	29,67	**20,00**	19,27
5	0,1,B,2,8,6,F,3,E,A,4,9,D,5,7,C	16384	4096	1172,87	351,67	112,33	42,80	21,20	**18,80**	19,27	19,33
6	0,1,B,2,8,3,F,6,E,A,4,9,D,5,7,C	16384	4096	2310,40	693,27	234,07	90,33	35,87	20,67	**19,33**	19,27
7	0,4,B,2,8,6,A,1,E,F,3,9,D,5,7,C	16384	4096	955,67	322,07	132,67	47,33	21,87	**19,20**	19,53	19,07
8	0,4,B,2,8,3,F,1,E,A,6,9,D,5,7,C	16384	4096	1495,47	397,40	126,47	46,40	21,87	**18,73**	19,20	18,87
9	0,B,F,9,1,5,6,8,3,A,4,C,E,D,7,2	16384	4096	1410,13	379,87	131,40	48,33	24,60	**18,73**	19,33	18,93
10	0,7,A,E,9,1,D,8,C,2,B,F,3,5,4,6	16384	4096	1985,87	686,33	186,33	61,80	23,73	**19,20**	19,20	19,40
11	4,A,9,2,D,8,0,E,6,B,1,C,7,F,5,3	16384	6144	2052,40	672,13	187,67	72,87	33,20	**19,23**	19,20	19,00
12	8,2,D,B,4,1,E,7,5,F,0,3,A,6,9,C	32768	9216	1298,33	328,53	68,47	29,80	**19,53**	19.13	19,07	18,87
13	A,5,3,F,C,9,0,6,1,2,8,4,B,E,7,D	32768	8192	1045,87	142,07	64,73	31,07	**18,87**	19,07	19,20	19,33
14	5,A,C,6,0,F,3,9,8,D,B,1,7,2,E,4	32768	16384	5043,20	1327,87	369,60	150,00	64,20	31,47	24,07	23,87
15	3,9,F,0,6,A,5,C,E,2,1,7,D,4,8,B	32768	8192	1338,67	184,67	43,73	**19,87**	19,47	19,33	18,00	19,00
16	F,0,A,9,3,5,4,E,8,B,1,7,6,C,D,2	32768	8192	2372,27	283,33	129,20	47,40	20,20	**19,00**	19,40	19,20
17	C,6,3,9,0,5,A,F,2,D,4,E,7,B,1,8	32768	8192	1716,80	258,33	85,60	35,80	**19.07**	1927	18,93	19,13
18	D,A,0,7,3,9,E,4,2,F,C,1,5,6,B,8	32768	16384	2080,00	415,47	117,80	42,53	**18,93**	19,07	19,33	19,33

Подстановка S^{11} (с порядковым номером 11 в таблице), отмечается в работе [213], взята из книги Б. Шнайера [5], в которой приведе семь четырёхразрядных подстановок, использовавшихся при шифровании

методом ГОСТ в приложении для ЦБ РФ, а также в однонаправленной хэш-функции ГОСТ.

В [213] рассмотрены также 32 подстановки в S-блоках модифицированного алгоритма s^3DES [217], которые считаются устойчивыми к методам дифференциального и линейного криптоанализа. Отмечается, что среди них только 7 подстановок (это подстановки S^{12},…, S^{18}) обладают нелинейностью $NL = 4$. Они представлены в таблице 2 соответствующими порядковыми номерами 12-18.

Как видно из представленных результатов при всех вариантах задействованных в шифрах S-блоков (кроме одного – 14-го) все шифры за девять циклов преобразований успевают стать случайными подстановками. Заметим здесь, что первые одиннадцать подстановок имеют значения показателей

δ-равномерности (максимальное значение переходов таблицы XOR разностей) равные минимально возможному значению равному 4-ём, а остальные имеют показатель δ-равномерности равный 8-ми. Видно, что подстановки с показателем δ-равномерности равный 8-ми ничем не уступают подстановкам с предельно малыми значениями показателя δ-равномерности.

Подстановка под номером 14 оказалась по дифференциальным показателям выпавшей из общего списка. Она тоже пришла к стационарному состоянию, но для неё асимптотическое значение оказалось равным 24-ём. Как показал анализ, эта подстановка, по-видимому, попала в список работы [213] случайно. Она по нашим оценкам относится к подстановкам вырожденного типа (имеет значение показателя нелинейности равное нулю).

Таблица 1.3 иллюстрирует результаты экспериментов, выполненных с использованием в шифре Рис.1.1 подстановок из работы [212] (подстановки шифра Serpent и золотые подстановки). В приложении этой работы представлены S-блоки и других шифров: Lucifer, Present, JH, ICEBERG, LUFFA, NOEKEON, HAMSI, Serpent, Hummingbird-1, Hummingbird-2, GOST

и DES. Мы, однако, ограничимся здесь только S-блоками шифра Serpent и золотыми подстановками.

Таблица 3.3

Поцикловые значения максимумов XOR таблиц шифра Хеуса с различными наборами S-блоков, взятыми из работы [212].

№ п/п	Подстановки шифра СЕРПЕНТ	1	2	3	4	5	6	7	8	9	10
1	3,8,f,1,a,6,5,b,e,d,4,2,7,0,9,c	16384	4096	460,67	70,60	32,07	**19,60**	18,93	19,20	19,33	19,00
2	f,c,2,7,9,0,5,a,1,b,e,8,6,d,3,4	16384	4096	467,33	70,00	30,27	**19,20**	18,93	19,07	19,13	19,27
3	8,6,7,9,3,c,a,f,d,1,e,4,0,b,5,2	16384	4096	631,47	78,73	35,80	**19,07**	19,13	19,07	18,80	18,80
4	0,f,b,8,c,9,6,3,d,1,2,4,a,7,5,e	16384	4096	502,80	67,20	29,47	**19,27**	19,07	19,33	19,00	19,
5	1,f,8,3,c,0,b,6,2,5,4,a,9,e,7,d	16384	4096	446,93	70,87	30,40	**18,80**	19,07	19,13	18,80	19,20
6	f,5,2,b,4,a,9,c,0,3,e,8,d,6,7,1	16384	4096	437,60	82,40	32,33	**19,07**	19,20	18,87	19,20	18,80
7	7,2,c,5,8,4,6,b,e,9,1,f,d,3,a,0	16384	4096	514,13	66,27	26,00	**18,80**	19,40	19,07	19,13	19,27
8	1,d,f,0,e,8,2,b,7,4,c,a,9,3,5,6	16384	4096	454,00	64,73	29,07	**19.07**	19,20	19,00	18,80	19,20
	Золотые подстановки	1	2	3	4	5	6	7	8	9	10
1	0,3,5,8,6,9,c,7,d,a,e,4,1,f,b,2	16384	4096	512,13	81,40	30,40	**19,20**	19,07	19,87	19,07	19,53
2	0,3,5,8,6,a,f,4,e,d,9,2,1,7,c,b	16384	4096	502,80	67,29	29,47	**19,20**	19,33	19,33	19,00	19,00
3	0,3,5,8,6,c,b,7,9,e,a,d,f,2,1,4	16384	4096	427,47	62,47	27,07	**19,13**	19,20	19,27	18,87	18,73
4	0,3,5,8,6,c,b,7,a,4,9,e,f,1,2,d	16384	4096	502,80	67,20	29,47	**19,27**	19,07	19,33	19.00	19,00

Как видно из результатов, представленных в табл. 1.3, подстановки шифра Серпент повторяют свойства золотых подстановок [212] (это подстановки одного и того же класса). Все подстановки в этом эксперименте обеспечивают переход к показателям случайной подстановки за шесть циклов (это наименьшее по всем экспериментам для шифра Хеуса число циклов для выхода по дифференциальным показателям к стационарному состоянию).

Заметим, что предельные показатели прихода к асимптотическому значению $\Lambda(\Delta X, \Delta Y)_{\max} \le 20$ демонстрируют и некоторые подстановки из таблицы 1.2, но в целом показатели подстановок из табл. 1.2 заметно уступают по динамическим характеристикам подстановок из табл. 1.3, т.е. для шифра Хеуса они являются далеко не лучшими.

В таблицах 1.4 и 1.5 демонстрируются уже результаты, полученные для таблиц линейных аппроксимаций шифра Хеуса с теми же конструкциями

Таблица 3.4

Поцикловые значения максимумов смещений линейных корпусов шифра Хеуса с различными наборами S-блоков, взятыми из работы [213].

№ п/п	Подстановки из [213]	1	2	3	4	5	6	7	8
1	0,D,B,8,3,6,4,1,F,2,5,E,A,C,9,7	16384	1792	800	830	834	**848**	822	810
2	0,1,9,E,D,B,7,6,F,2,C,5,A,4,3,8	16384	2048	1280	830	834	810	**798**	860
3	0,1,D,B,9,E,6,7,C,5,8,3,F,2,4,A	16384	2048	848	798	796	824	800	**822**
4	0,1,2,4,3,5,8,A,7,9,6,D,B,E,C,F	16384	2048	**832**	878	820	876	788	830
5	0,1,B,2,8,6,F,3,E,A,4,9,D,5,7,C	16384	2048	832	796	800	820	786	**814**
6	0,1,B,2,8,3,F,6,E,A,4,9,D,5,7,C	16384	2048	816	810	848	792	800	792
7	0,4,B,2,8,6,A,1,E,F,3,9,D,5,7,C	16384	2048	816	800	816	800	846	**786**
8	0,4,B,2,8,3,F,1,E,A,6,9,D,5,7,C	16384	2048	816	814	832	804	838	**820**
9	0,B,F,9,1,5,6,8,3,A,4,C,E,D,7,2	16384	1664	848	792	824	816	**870**	822
10	0,7,A,E,9,1,D,8,C,2,B,F,3,5,4,6	16384	2048	808	818	824	840	824	**804**
11	4,A,9,2,D,8,0,E,6,B,1,C,7,F,5,3	16384	2048	**848**	830	798	790	848	790
12	8,2,D,B,4,1,E,7,5,F,0,3,A,6,9,C	32768	8192	2048	818	828	828	**818**	816
13	A,5,3,F,C,9,0,6,1,2,8,4,B,E,7,D	32768	8192	2048	844	818	858	786	**824**
14	5,A,C,6,0,F,3,9,8,D,B,1,7,2,E,4	32768	32768	32768	32768	32768	32768	32768	32768
15	3,9,F,0,6,A,5,C,E,2,1,7,D,4,8,B	32768	8192	1512	794	792	880	**800**	**830**
16	F,0,A,9,3,5,4,E,8,B,1,7,6,C,D,2	32768	8192	2048	824	826	818	846	**826**
17	C,6,3,9,0,5,A,F,2,D,4,E,7,B,1,8	32768	8192	1280	840	800	838	832	**886**
18	D,A,0,7,3,9,E,4,2,F,C,1,5,6,B,8	32768	8192	2048	796	836	816	812	**830**

S-блоков, что и в предыдущих экспериментах (при использовании одного ключа зашифрования). Все подстановки, приведенные в таблицах, за исключением подстановки под номером 14, обладают максимально возможным значением показателя нелинейности $NL = 4$.

Таблица 3.5

Поцикловые значения максимумов смещений линейных корпусов шифра Хеуса с различными наборами S-блоков, взятыми из работы [212].

№ п/п	Подстановки шифра СЕРПЕНТ	1	2	3	4	5	6	7	8
1	3,8,f,1,a,6,5,b,e,d,4,2,7,0,9,c	16384	8192	4128	2032	1152	**792**	812	806
2	f,c,2,7,9,0,5,a,1,b,e,8,6,d,3,4	16384	8192	4352	2080	944	**814**	806	850
3	8,6,7,9,3,c,a,f,d,1,e,4,0,b,5,2	16384	8192	4128	2152	1154	**844**	822	808
4	0,f,b,8,c,9,6,3,d,1,2,4,a,7,5,e	16384	8192	4128	2032	**888**	796	800	818
5	1,f,8,3,c,0,b,6,2,5,4,a,9,e,7,d	16384	8192	4896	1892	942	**882**	830	858
6	f,5,2,b,4,a,9,c,0,3,e,8,d,6,7,1	16384	8192	4896	1712	1212	**782**	828	816
7	7,2,c,5,8,4,6,b,e,9,1,f,d,3,a,0	16384	8192	4896	1928	1112	**812**	842	868
8	1,d,f,0,e,8,2,b,7,4,c,a,9,3,5,6	16384	8192	3840	2064	956	**818**	812	830
	Золотые подстановки	1	2	3	4	5	6	7	8
1	0,3,5,8,6,9,c,7,d,a,e,4,1,f,b,2	16384	8192	5056	1856	956	**806**	798	792
2	0,3,5,8,6,a,f,4,e,d,9,2,1,7,c,b	16384	8192	3936	1952	818	908	**820**	812
3	0,3,5,8,6,c,b,7,9,e,a,d,f,2,1,4	16384	8192	3648	1744	**822**	830	772	802
4	0,3,5,8,6,c,b,7,a,4,9,e,f,1,2,d	16384	8192	4128	2176	**818**	818	824	862

И здесь представленные результаты во всех случаях (кроме 4-ой, 11-ой и 14-ой подстановок), уверенно (в пределах восьми циклов) демонстрируют при ход шифров с различными конструкциями S-блоков к показателям случайной подстановки. Заметим, что подстановка под номерами 14 и в этом

случае показывает практическую непригодность для построения шифрующего преобразования. Подстановки 4 и 11 тоже по нашим критериям для шифра Хеуса являются плохими (заметим, что 11-я подстановка использована в шифре ГОСТ). Мы на 14-й подстановке и других ″плохих″ подстановках детальнее остановимся позднее.

Дальнейшие результаты посвящены анализу дифференциальных и линейных свойств шифров (уменьшенных моделей), когда в них используется сильное линейное преобразование. В таблицах 1.6 и 1.7 приводятся результаты оценки дифференциальных и линейных показателей уменьшенных моделей шифра Rijndael с различными конструкциями S-блоков, повторяющими ранее рассмотренные наборы. Во всех случаях в шифрах baby-Rijndael реализовано отмеченное выше линейное преобразование MixColumns на весь текст (сильное линейное преобразование).

Таблица 3.6

Поцикловые значения максимумов полных дифференциалов (XOR таблиц) шифров baby-Rijndael с различными наборами S-блоков, взятыми из работ [112, 213].

Подстановки из работ [212,213]		Число циклов зашифрования					
		1	2	3	4	5	6
1	0,D,B,8,3,6,4,1,F,2,5,E,A,C,9,7	16384	128	**19,33**	19,13	19,53	19
2	0,1,9,E,D,B,7,6,F,2,C,5,A,4,3,8	16384	88	21,13	**19,2**	19,4	19
3	0,1,D,B,9,E,6,7,C,5,8,3,F,2,4,A	16384	128	**19,67**	19	19,67	19,33
4	0,1,2,4,3,5,8,A,7,9,6,D,B,E,C,F	16384	128	**18,73**	19	19,27	18,93
5	0,1,B,2,8,6,F,3,E,A,4,9,D,5,7,C	16384	128	**19,13**	18,73	19,2	19,13
6	0,1,B,2,8,3,F,6,E,A,4,9,D,5,7,C	16384	131,7	**19,33**	19,07	18,87	18,93
7	0,4,B,2,8,6,A,1,E,F,3,9,D,5,7,C	16384	80	**19,2**	19,13	19,33	18,87
8	0,4,B,2,8,3,F,1,E,A,6,9,D,5,7,	1638	128	**19,6**	19,0	19,2	19,1

	C	4			7		3
9	0,B,F,9,1,5,6,8,3,A,4,C,E,D,7,2	16384	128	**19,2**	19	19,4	19,4
10	0,7,A,E,9,1,D,8,C,2,B,F,3,5,4,6	16384	136	21,87	**19,07**	19,07	18,87
11	4,A,9,2,D,8,0,E,6,B,1,C,7,F,5,3	16384	222	20,13	**19,67**	19,2	19,47
12	8,2,D,B,4,1,E,7,5,F,0,3,A,6,9,C	24576	846	59,27	**19,27**	19,33	18,73
13	A,5,3,F,C,9,0,6,1,2,8,4,B,E,7,D	32768	1024	115,9	**19,00**	19,07	19,4
14	5,A,C,6,0,F,3,9,8,D,B,1,7,2,E,4	32768	2048	286	58,2	30,67	31
15	3,9,F,0,6,A,5,C,E,2,1,7,D,4,8,B	32768	576	42,6	**19,13**	19,13	19,2
16	F,0,A,9,3,5,4,E,8,B,1,7,6,C,D,2	32768	1024	120	**19,07**	18,73	19,13
17	C,6,3,9,0,5,A,F,2,D,4,E,7,B,1,8	32768	576	53,07	**19,13**	19,27	18,87
18	D,A,0,7,3,9,E,4,2,F,C,1,5,6,B,8	32768	768	80	**19,07**	19,07	19,47
подстановки шифра СЕРПЕНТ		1	2	3	4	5	6
1	3,8,F,1,A,6,5,B,E,D,4,2,7,0,9,C	16384	128	20,33	**19,00**	19,47	19,13
2	F,C,2,7,9,0,5,A,1,B,E,8,6,D,3,4	16384	96	**19,07**	19,07	19,67	19,33
3	8,6,7,9,3,C,A,F,D,1,E,4,0,B,5,2	16384	144	**19,67**	19,33	19,2	19,4
4	0,F,B,8,C,9,6,3,D,1,2,4,A,7,5,E	16384	96	**19,8**	19,27	19,13	19,13
5	1,F,8,3,C,0,B,6,2,5,4,A,9,E,7,D	16384	130,9	20,53	**19,2**	18,93	19,07
6	F,5,2,B,4,A,9,C,0,3,E,8,D,6,7,1	16384	130,1	**19,27**	19,13	19,2	19,13
7	7,2,C,5,8,4,6,B,E,9,1,F,D,3,A,0	16384	128	**19,27**	19	19,47	18,87
8	1,D,F,0,E,8,2,B,7,4,C,A,9,3,5,6	16384	128	22,2	**18,93**	18,73	18,8
Золотые подстановки		1	2	3	4	5	6
1	0,3,5,8,6,9,C,7,D,A,E,4,1,F,B	1638	134,	**19,0**	19,2	18,9	19,13

	,2	4	9	**0**	7	3	
2	0,3,5,8,6,A,F,4,E,D,9,2,1,7,C,B	16384	128	**19,80**	19,00	1947	18,93
3	0,3,5,8,6,C,B,7,9,E,A,D,F,2,1,4	16384	128	**19,53**	19,2	19,27	19
4	0,3,5,8,6,C,B,7,A,4,9,E,F,1,2,D	16384	132	**18,8**	19,33	19,07	19,13

По результатам табл. 1.8 и табл. 1.9 можно сделать вывод, что случайные S-блоки в своей массе демонстрирую показатели, которые оказываются ничуть не хуже показателей рассмотренных ранее S-блоков (и идеальных тоже).

Из представленных результатов следует, что практически независимо от конструкций использованных S-блоков все варианты шифров за три-четыре

Таблица 3.7

Поцикловые значения максимумов смещений линейных корпусов шифра baby-Rijndael с различными наборами S-блоков, взятыми из работ [212,213].

Подстановки		1	2	3	4	5	6	7
1	0,D,B,8,3,6,4,1,F,2,5,E,A,C,9,7	16384	1792	**800**	830	834	848	822
2	0,1,9,E,D,B,7,6,F,2,C,5,A,4,3,8	16384	16384	1280	**830**	834	810	798
3	0,1,D,B,9,E,6,7,C,5,8,3,F,2,4,A	16384	2048	944	**800**	822	832	810
4	0,1,2,4,3,5,8,A,7,9,6,D,B,E,C,F	16384	2048	**832**	878	820	876	788
5	0,1,B,2,8,6,F,3,E,A,4,9,D,5,7,C	16384	2048	**832**	796	800	820	786
6	0,1,B,2,8,3,F,6,E,A,4,9,D,5,7,C	16384	2048	**816**	810	848	792	800
7	0,4,B,2,8,6,A,1,E,F,3,9,D,5,7,C	16384	2048	**816**	800	816	800	846
8	0,4,B,2,8,3,F,1,E,A,6,9,D,5,7,C	16384	2048	**816**	814	832	804	838
9	0,B,F,9,1,5,6,8,3,A,4,C,E,D,7,2	16384	1664	**848**	792	824	816	870
10	0,7,A,E,9,1,D,8,C,2,B,F,3,5,4,6	16384	2048	**808**	818	814	840	824
11	4,A,9,2,D,8,0,E,6,B,1,C,7,F,5,3	16384	2048	**832**	830	798	790	848
12	8,2,D,B,4,1,E,7,5,F,0,3,A,6,9,C	32768	8192	2048	**822**	840	798	882
13	A,5,3,F,C,9,0,6,1,2,8,4,B,E,7,D	32768	8192	2048	**844**	818	858	786
14	5,A,C,6,0,F,3,9,8,D,B,1,7,2,E,4	32768	32768	32768	32768	32768	32768	32768
15	3,9,F,0,6,A,5,C,E,2,1,7,D,4,8,B	32768	8192	1512	**794**	792	880	800
16	F,0,A,9,3,5,4,E,8,B,1,7,6,C,D,2	32768	8192	2048	**824**	826	818	846
17	C,6,3,9,0,5,A,F,2,D,4,E,7,B,1,8	32768	8192	1280	**840**	800	838	832
18	D,A,0,7,3,9,E,4,2,F,C,1,5,6,B,8	32768	8192	2048	**796**	836	816	812
Подстановки шифра СЕРПЕНТ		1	2	3	4	5	6	7
1	3,8,F,1,A,6,5,B,E,D,4,2,7,0,9,C	16384	2048	**872**	878	808	862	844
2	F,C,2,7,9,0,5,A,1,B,E,8,6,D,3,4	16384	2048	**816**	792	786	882	874

3	8,6,7,9,3,C,A,F,D,1,E,4,0,B,5,2	16384	2048	**880**	820	816	844	860
4	0,F,B,8,C,9,6,3,D,1,2,4,A,7,5,E	16384	2048	**800**	844	856	844	850
5	1,F,8,3,C,0,B,6,2,5,4,A,9,E,7,D	16384	2048	**840**	864	838	798	778
6	F,5,2,B,4,A,9,C,0,3,E,8,D,6,7,1	16384	2048	**808**	846	820	822	832
7	7,2,C,5,8,4,6,B,E,9,1,F,D,3,A,0	16384	2048	**808**	866	830	826	828
8	1,D,F,0,E,8,2,B,7,4,C,A,9,3,5,6	16384	2048	**872**	820	826	792	804
Золотые подстановки								
1	0,3,5,8,6,9,C,7,D,A,E,4,1,F,B,2	16384	2048	**800**	814	844	786	806
2	0,3,5,8,6,A,F,4,E,D,9,2,1,7,C,B	16384	2048	**824**	794	804	868	836
3	0,3,5,8,6,C,B,7,9,E,A,D,F,2,1,4	16384	1792	**808**	812	846	782	822
4	0,3,5,8,6,C,B,7,A,4,9,E,F,1,2,D	16384	1792	**864**	824	786	794	826

цикла приходят к асимптотическим значению максимума полного дифференциала и максимума смещения линейного корпуса, совпадающим с соответствующими теоретическими значениями, полученным для случайной подстановки [43]. Сильное линейное преобразование практически сгладило отличия в результатах.

Наконец, в третьей серии экспериментов выполнены оценки дифференциальных и линейных свойств шифров при использовании в них случайно сгенерированных подстановок. Результаты этих экспериментов представлены в табл. 1.8 и 1.9 (табл. 1.8 иллюстрирует дифференциальные показатели, а табл. 1.9 линейные).

Таблица 3.8

Поцикловые значения максимумов полных дифференциалов (XOR таблиц) шифра baby-Rijndael со случайными S-блоками.

случайные подстановки		1	2	3	4	5	6
1	0,A,4,C,3,7,E,9,1,F,2,B,5,6,D,8	24576	335,9	25,27	**19,27**	18,93	19,27
2	B,5,A,2,7,D,8,E,4,3,1,F,6,C,9,0	32768	768	34,4	**19,07**	18,87	19,27
3	3,B,4,C,1,A,8,5,2,0,D,E,7,6,9,F	24576	355,2	21,93	**19**	19,33	19,2
4	3,6,C,7,0,D,5,A,B,1,2,4,9,8,F,E	24576	223,2	**19,53**	19,2	19	18,93
5	2,4,5,A,9,E,7,B,C,6,F,3,1,0,8,D	24576	223,1	**19,53**	19,33	19,33	19,27
6	C,A,E,2,0,9,4,8,5,1,6,B,7,D,F,3	24576	524,00	32,13	**19,33**	19,27	19,00

7	0,D,F,5,7,4,3,B,E,6,9,2,8,C,1,A	24576	190,40	20,93	**19,07**	19,07	19,4
8	7,F,E,B,1,2,0,D,5,C,4,8,A,3,6,9	24576	328,00	35,6	**19,33**	19,13	19,2
9	4,2,0,E,6,B,D,7,C,A,9,F,1,5,3,8	24576	216	**19,16**	19,2	19,33	19,47
10	6,1,7,F,C,4,5,D,0,E,8,2,A,3,B,9	24576	336	29,07	**19,2**	19,53	19,27

Таблица 39

Поцикловые значения максимумов смещений линейных корпусов шифра baby-Rijndael для случайно сгенерированных S-блоков

случайные подстановки		1	2	3	4	5	6
1	0,A,4,C,3,7,E,9,1,F,2,B,5,6,D,8	24576	5184	1000	**796**	814	824
2	B,5,A,2,7,D,8,E,4,3,1,F,6,C,9,0	24576	5248	1616	**828**	838	836
3	3,B,4,C,1,A,8,5,2,0,D,E,7,6,9,F	24576	3584	984	**816**	808	794
4	3,6,C,7,0,D,5,A,B,1,2,4,9,8,F,E	24576	3584	**816**	794	820	808
5	2,4,5,A,9,E,7,B,C,6,F,3,1,0,8,D	24576	3520	**856**	886	844	866
6	C,A,E,2,0,9,4,8,5,1,6,B,7,D,F,3	24576	5248	1096	**826**	804	822
7	0,D,F,5,7,4,3,B,E,6,9,2,8,C,1,A	24576	3584	928	**858**	816	810
8	7,F,E,B,1,2,0,D,5,C,4,8,A,3,6,9	24576	3520	**808**	874	850	842
9	4,2,0,E,6,B,D,7,C,A,9,F,1,5,3,8	16384	2048	**816**	800	784	880
10	6,1,7,F,C,4,5,D,0,E,8,2,A,3,B,9	24576	5248	1576	900	**814**	850

Заметим, что результаты табл. 1.8 получены при усреднении по 30-ти разным ключам, а табл. 1.9 получены для одного ключа. Ключи сгенерированы случайным образом.

Остаётся отметить, что к этим же результатам можно присоединить и многочисленные другие из ранее отмеченных публикаций [38,39,218 и др.], в которых обосновывается новая методология оценки стойкости блочных симметричных шифров к атакам дифференциального и линейного криптоанализа.

Результаты экспериментов, выполненные в настоящей работе, да и других работах показали, что свойства (различие) подстановок можно почувствовать (увидеть) только в шифрах с плохим (малоэффективным)

диффузионным слоем. Шифры с хорошим диффузионным слоем этой разницы просто не чувствуют! Сама разница, если она есть, проявляется в числе циклов, необходимых шифру для прихода к стационарному состоянию. Но эта разница в худшем случае достигает значения максимум трёх циклов.

Заметим, что к аналогичным выводам мы пришли и при исследовании больших шифров. Эксперименты с ними также полностью подтвердили исходную гипотезу о сходимости шифров с ростом числа циклов к показателям случайных подстановок соответствующей степени.

Несколько слов о так называемых вырожденных подстановочных конструкциях. Эксперименты показывают, что к вырожденным подстановкам следует отнести, прежде всего, подстановки с показателями нелинейности равными нулю (максимальное значение смещения таблицы ЛАТ равно 2^{n-1}). В табл. 1.10 и табл. 1.11 мы представляем поведение шифров с такими подстановками. В качестве примеров взяты: тождественная подстановка (единичная подстановка симметрической группы), зафиксированная в процессе экспериментов в таблицах 1 и 3, подстановка под номером 14, а также специально сгенерированная подстановка из работы [211], с показателем нелинейности также равным нулю).

Очевидно, что сами вырожденные подстановки легко определяются на основе результатов экспериментов. Как мы видим, подстановка из работы [211] оказалось плохой по дифференциальным показателям и для шифра Хейса и для шифра Rijndael (в последнем случае шифр выходит к стационарному значению на шестом цикле). Различное влияние показателя нелинейности, по-видимому, определяется числом линейных булевых функций в используемых S-блоках.

Таблица 3.10

Поцикловые значения максимумов полных дифференциалов (XOR таблиц) шифра Хеуса с вырожденными S-блоками.

	Тождественная подстановка	Значения максимумов ТР в зависимости от числа циклов					
1	1,2,3,4,5,6,7,8,9,1A,B,C,D,E,F ЛАТ – 8, ДТ – 16 (15)	1	2	3	4	5	6
		57617,07	50364,40	45675,60	40971,40	37338,80	39267,00
		Число циклов					
		7	8	9	10	11	12
		41487,80	43386,53	44803,13	46411,40	47075,40	47872,93
	Подстановка 14 из таблицы 2	Число циклов					
2	5,A,C,6,0,F,3,9,8,D,B,1,7,2,E,4 ЛАТ – 8, ДТ – 8 (12)	1	2	3	4	5	6
		32768	16384	5043,20	1327,87	369,60	151,07
		Число циклов					
		7	8	9	10	11	12
		61,53	32,60	24,20	23,87	23,93	24,13
	Подстановка 1 из работы [22]	Число циклов					
3	C,D,5,1,A,B,6,2,E,3,7,F,4,0,8,9 ЛАТ – 8, ДТ – 12 (3)	1	2	3	4	5	6
		49152,00	27648,00	15552,00	3616,00	1016,00	451,27
		Число циклов					
		7	8	9	10	11	12
		209,27	106,60	53,07	27,53	20,07	19,07

Таблица 3.11

Поцикловые значения максимумов смещений таблиц ЛАТ для шифра Хеуса с вырожденными S-блоками.

	Тождественная подстановка	Значения максимумов ЛАТ в зависимости от числа циклов					
1	0,1,2,3,4,5,6,7,8,9,1A,B,C,D,E,F	1	2	3	4	5	6
		32768	32768	32768	32768	32768	32768
		Число циклов					
		7	8	9	10	11	12
		32768	32768	32768	32768	32768	32768
	Подстановка 14 из таблицы 2	Число циклов					
2	5,A,C,6,0,F,3,9,8,D,B,1,7,2,E,4	1	2	3	4	5	6
		32768	32768	32768	32768	32768	32768
		Число циклов					
		7	8	9	10	11	12
		32768	32768	32768	32768	32768	32768
	Подстановка 1 из работы [22]	Число циклов					
3	C,D,5,1,A,B,6,2,E,3,7,F,4,0,8,9	1	2	3	4	5	6
		32768	24576	12288	5244	2044	1080
		Число циклов					
		7	8	9	10	11	12
		792	872	826	816	842	816

Аналогично можно сказать и о дифференциальных показателях. В табл. 1.12 приведены примеры подстановки с предельными (по максимуму) дифференциальными показателями. По примерам табл. 1.10 можно сделать

вывод, что плохими могут быть подстановки и со значением максимумов дифференциальных таблиц превышающих 8-10 (они, как правило, свойственны подстановкам с максимальными значениями показателя нелинейности). Мы на этом пока закончим обсуждение вырожденных подстановок. Более полное их исследование составляет предмет отдельной работы.

Таблица3.12

Поцикловые значения максимумов полных дифференциалов (XOR таблиц) шифра baby-Rijndael с вырожденными S-блоками.

	Подстановка с тремя переходами с вероятностью 1	Число циклов					
4	F,D,9,B,1,3,2,0,4,6,7,5,A,8,C,E ЛАТ – 8, ДТ – 16 (3)	1	2	3	4	5	6
		58112,00	16827,00	8192,00	8192,27	8192,47	8193,93
		Число циклов					
		7	8	9	10	11	12
		8195,87	8195,47	8197,47	8198,20	8198,47	8200,87
	Подстановка с одним переходом с вероятностью 1	Число циклов					
5	6,4,1,3,0,2,7,5,E,C,D,F,8,A,9,B ЛАТ – 8, ДТ – 16 (1)	1	2	3	4	5	6
		57617,07	10290,13	8192,00	8193,67	8192,47	8196,93
		Число циклов					
		7	8	9	10	11	12
		8198,67	8199,27	8200,27	8202,93	8204,13	8207,27

Здесь важно подчеркнуть, что вероятность попасть на вырожденную подстановку при их случайном формировании весьма мала. Так, вероятность получения полубайтовой подстановки с показателем нелинейности равным нулю, как показывают расчёты, близка к одной десятитысячной, а для генерации байтовой подстановки с таким же показателем нелинейности потребуется перебрать один миллиард подстановок. Вероятность получения полубайтовой подстановки с максимальным значением перехода равным 16 (вероятность перехода входной разности ΔX в выходную разность ΔY равна 1) определяется соотношением $P_{N_1} = \frac{N_1}{N} = \frac{2{,}33 \cdot 10^9}{2 \cdot 10^{13}} = 1{,}1 \cdot 10^{-4}$, т.е. тоже близка к одной десятитысячной.

Тем не менее, представленные примеры применения вырожденных и невырожденных подстановок ярко свидетельствуют, что S-блоки в шифрах играют весьма важную роль. Без подстановок невырожденного типа построить хорошего криптографического преобразования нельзя. Подстановки выполняют один из важных для шифра механизмов − механизм нелинейного перемешивания (перестановки) битов блоков данных, с помощью которого удаётся наиболее просто добиться эффекта хаотичности в их преобразовании.

Подводя итоги приведенным результатам, можно отметить следующие моменты.

Уже известно достаточно много конструкций S-блоков, разработанных и использованных в различных шифрах (S-блоки, строящиеся с использованием аппарата булевых функций, S-блоки, конструируемые на основе преобразований детерминированного типа, случайные подстановки, построенные с использованием различных критериев отбора, S-блоки, выбранные по определённым критериям путём полного перебора и другие конструкции). Как следует из представленных результатов, различие между S-блоками можно увидеть (почувствовать) лишь при использовании в цикловом преобразовании шифра слабого линейного преобразования (как, например, в шифре Хеуса или DES). Эта слабость проявляется в увеличенном числе циклов зашифрования, необходимых для достижения шифром стационарного состояния, под которым понимается момент, начиная с которого законы распределения переходов XOR таблиц и смещений таблиц линейных аппроксимаций начинают повторять соответствующие законы распределения вероятностей случайной подстановки.

Далее мы представляем результаты анализа динамических свойств шифра Хеуса для полубайтовых S-блоков (шифра со слабым линейным преобразованием), которые концентрируются вокруг следующих положений:

- наиболее эффективными оказываются S-блоки, названные в работе [212] идеальными. Для них минимальное число циклов, необходимых для выхода

шифра Хеуса на стационарный режим по дифференциальным показателям оказывается равным 6-ти (для линейных показателей − даже пяти);

- все другие конструкции S-блоков специального вида из работы [213] и др., а также S-блоки детерминированного и случайного типа, как правило, обладают динамическими показателями прихода к стационарному состоянию, изменяющимися в значительных пределах (от 6-ти до 9-ти и более циклов);

- S-блоки, конструкции которых использованы в современных шифрах, ориентированные на обеспечение минимальных значений показателей дельта-равномерности и максимально достижимых значений нелинейности (S-блоки шифров AES, Камелия, Лабиринт, ADE и др., построенные по идеям
К. Нюберг), с точки зрения динамических свойств обладают для шифра Хеуса далеко не лучшими характеристиками;

Общим выводом выполненных исследований можно отметить то, что хорошее линейное преобразование (преобразование с высоким значением коэффициента ветвления) нивелирует различие между S-блоками (не тривиального типа). Все известные конструкции S-блоков, используемых в шифрах, показывают практически одинаковые значения показателя эффективности (числа циклов выхода к стационарному состоянию, которые равны 3-4).

В целом же, если говорить об асимптотических значениях максимумов дифференциалов и линейных корпусов (получающиеся при полном наборе шифрующих преобразований), определяющих по современным меркам показатели их доказуемой стойкости, то для практически всех известных шифров они от свойств, применяемых в шифрах S-блоковых конструкций, не зависят. Причём, S-блоки с подходящими (высокими) криптографическими показателями легко могут быть получены (отобраны) на основе применения случайных подстановок. Этот факт приводит к важному для криптографии выводу, что заниматься поиском S-блоковых конструкций с улучшенными

криптографическими показателями для шифров с эффективным линейным преобразованием является не перспективной задачей. Это интенсивно развиваемое в криптографии направление *не имеет* сколько-нибудь существенных продолжений. Для шифров со слабым линейным преобразованием задача поиска (отбора) более совершенных S-блоковых конструкций сохраняет свою актуальность, однако нужно тогда смириться с тем, что шифры со слабым линейным преобразованием будут всегда существенно (до трёх и более циклов) уступать в динамике перехода к стационарному состоянию шифру с сильным линейным преобразованием. Более того, для шифров со слабым линейным преобразованием S-блоки могут оказаться такими, что шифр будет иметь затянутый период перехода к стационарному состоянию, как это получилось, например, для шифра DES. когда на 13-ти циклах удалось найти дифференциальную характеристику существенно более вероятную, чем это следует из теоретических расчётов для случайных подстановок. Заметим, однако, что и здесь сохраняет свою практическую ценность и целесообразность применение случайно взятых подстановок (заметим, что в [5] отмечается, что случайно взятые подстановки ухудшают показатели стойкости шифра DES; вопрос оценки свойств составных S-блоков, которые использованы в DES, требует отдельного изучения).

Следует особо подчеркнуть нашу позицию состоящую в том, что S-блоки являются всё же необходимым и существенным элементом эффективного шифрующего преобразования, выполняющим одну из главных функций процедуры зашифрования − нелинейного перепутывания битовых выходов, при этом основное значение имеет свойство хаотичности нелинейного преобразования, которое проявляется в том, что осуществляется массовое в значительной степени случайное изменение битовых позиций на его выходе. При последовательном выполнении нескольких таких преобразований практически независимо от конкретного характера исходного нелинейного преобразования (нетривиального типа) происходит

статистическое уравновешивание эффектов от воздействия каждого из битов входа, что и приводит к результирующему однородному (стационарному) распределению для каждого перехода входной разности ΔX в выходную разность ΔY.

Сами блоки нелинейных замен (S-блоки) влияют лишь на динамику перехода к стационарным состояниям, свойственным случайным подстановкам соответствующей степени. Таким образом, S-блоки представляются существенной компонентой современных итеративных шифров. Это один из наиболее простых механизмов введения нелинейности в криптографическое преобразование, хотя и реализовать нелинейное преобразование можно и без S-блоковых конструкций (как, например, в шифре ThreeFish). Но и в этом случае нелинейное преобразование можно интерпретировать как соответствующий S-блок. Устранение нелинейного преобразования (или его низкая эффективность) разрушает один из существенных механизмов случайного перемешивания, реализуемого шифром

Отдельно можно отметить важную роль нелинейных подстановочных преобразований в формировании механизма случайного перемешивания. Они (S-блоки) сами являются источником хаотической перестановки битов входного блока данных. И без введения случайной компоненты, задаваемой цикловыми подключами, произведение подстановочных (даже однотипных) преобразований приводит к результирующему подстановочному преобразованию случайного типа (законы распределения переходов XOR таблиц и смещений таблиц линейных аппроксимаций повторяют соответствующие законы распределений случайной подстановки) независимо от вида исходного подстановочного преобразования (не тривиального типа). Именно на основе этого механизма (можно сказать закона природы) осуществляется переход любого шифра с ростом числа циклов к стационарному состоянию, после достижения которого дальнейшее

наращивание числа циклов не приводит к изменению показателей его стойкости.

Ещё один из установленных в ходе экспериментов факт. Хорошие подстановки, в том числе и подстановки, относящиеся к идеальным, не лишены тождественных переходов (фиксированных точек). Более того, вполне пригодными для криптографических применений оказываются подстановки, имеющие до 6-ти фиксированных точек (в наших экспериментах).

Не представляет сомнения, что аналогичные выводы будут справедливы и для подстановок более высокой степени.

3.2.3 Сравнительный анализ дифференциальных свойств уменьшенных моделей шифров.

Теперь наша ближайшая задача убедиться в справедливости изложенных выше результатов и положений применительно к уменьшенным моделям современных шифров с полубайтовыми S-блоками, повторяющими по конструкции, S-блоки, использованные в оригинальных разработках (прототипах). Они были уже приведены в табл. 1. Сегодня уже проверено много уменьшенных конструкций современных шифров. Часть из полученных результатов представлена ниже.

В работе [38] изучена поцикловая зависимость максимумов полных дифференциалов (таблиц XOR разностей для всего многоциклового шифрующего преобразования) от ключевых значений (значений цикловых подключей) для всех шифров, представленных на украинский конкурс, а также для шифра мини-AES, и мы приведём здесь основные её результаты. Ещё раз здесь подчеркнём, что в процессе экспериментов оценивались поцикловые значения не средних значения максимумов дифференциальных вероятностей (*AMDP*), а поцикловые средние (по ключам) значения максимумов заполнений ячеек таблиц XOR разностей, которые связаны с *AMDP* очевидным соотношением (для 16-битных шифров), следующим из определения 7 раздела 4:

$$\overline{d_{\max}} = \underset{k}{ave}\, DP_{\max}^{f[k]} \cdot 2^{16} = ADMP^{f} \cdot 2^{16}.$$

В табл. 5.13 представляются сначала результаты оценки поцикловых значений максимумов полных дифференциалов (таблиц XOR разностей) для мини-версии шифра Калина. При выполнении экспериментов были построены дифференциальные таблицы с частичным и полным перебором ключей для числа циклов зашифрования от 1-го до 4-х с использованием S-блоков с минимально возможными значениями δ – равномерности (S-блоки BabyR).

Таблица 5.13

Результаты полного перебора ключей для мини-Калины

Число циклов	Число экспериментов	Абсолютные значения максимумов ($d_{\max}$)	Средние значения максимумов ($\overline{d_{\max}}$)
1	11279	5120	3295.16
2	48535	672	361.316
3	65536	28	19.2439
4	65536	26	19.1112

Как следует из представленных результатов среднее (по множеству ключей) значение максимума полных дифференциалов уже при трех циклах шифрования приходит к асимптотическому (установившемуся) значению, не превышающему числа $m + 4 = 20$ (значении дифференциалов попадают в доверительный интервал).

Одно из принципиальных наблюдений состоит в том, что практически среднее значение максимума полного дифференциала рассматриваемого шифрующего преобразования оказывается близким к максимуму максиморуму по всему множеству ключей зашифрования, т.е. оценку для среднего значения максимума XOR таблицы (полного дифференциала) для произвольного ключа зашифрования можно рассматривать как доказанное значение, позволяющее в соответствии с (4.14) определить стойкость шифра к атакам дифференциального криптоанализа (доказуемую безопасность).

При проведении дальнейших экспериментов мы во многом повторили (проверили) результаты исследований, выполненных для уменьшенных моделей в других работах [39,57,222,223 и др.]. Здесь они представлены в более обстоятельной форме (известные результаты выполнены для шифров AES и ADE для выборки 1000 ключей, в то время как остальные шифры проверены на отдельных ключах зашифрования).

В экспериментах работы [38], результаты которой представлены ниже, было построено для каждого шифра 100 дифференциальных таблиц с использованием нелинейных узлов замены из табл. 4.1 и сгенерированных дополнительно.

Определялись следующие показатели:

- абсолютные значения максимумов таблиц разностей мини-шифров по множеству из 100 ключей зашифрования.

- средние по множеству из 100 случайно выбранных ключей шифрования значения максимумов таблиц разностей мини-шифров;

При выполнении исследований этого этапа в шифрах Мини-AES, Мини-ADE и Мини-Лабиринт использовались в том числе полубайтовые S-блоки, повторяющие конструкции S-блоков больших версий шифров (с минимально возможными значениями $\delta-$равномерности, равными четырём).

В других шифрах со случайными S-блоками (Калина и Мухомор) были использованы два типа S-блоков: со значениями δ-равномерности, равными 4-м и случайно порожденные S-блоки, с большими значениями δ-равномерности. Кроме того, список уменьшенных моделей был дополнен 16-битным шифром из работы [215], повторяющим идеи построения SPN шифра, изложенные в работе Х. Фейстеля [59]. Результаты экспериментов первого направления исследований иллюстрирует таблица 5.14.

Таблица 5.14

Абсолютные значения максимумов таблиц разностей различных шифров

	Шифр Хейса	Мини-	Мини-	Мини-	Мини-	Мини-Калина

			AES	ADE	Лабиринт	Мухомор			
Число циклов	S-блок $\delta = 8$	S-блок $\delta = 4$	S-блок $\delta = 4$	S-блок $\delta = 4$	S-блок $\delta = 4$	S-блок $\delta = 8$	S-блок $\delta = 4$	S-блок $\delta = 8$	S-блок $\delta = 4$
1	32768	16384	16384	16384	-	65536	65536	7168	6144
2	12288	4096	4096	5120	-	16384	6144	1376	640
3	2490	488	352	1024	128	4608	3072	38	24
4	286	98	22	38	22	2304	224	22	22
5	92	46	22	24	24	156	60	22	22
6	36	22	20	22	24	36	20	24	24
7	22	22	-	-	26	-	-	-	-
8	22	22	-	-	22	-	-	-	-

Как следует из этой таблицы, все шифры дают результаты, хорошо согласующиеся с результатами, рассмотренными ранее для шифра Калина. Шифр Хейса (со слабым линейным преобразованием) приходит к устойчивому абсолютному значению максимума после шести-семи циклов, другие шифры (мини-AES, мини-Лабиринт и мини-Калина) выходят к установившемуся значению максимума гораздо быстрее. Опять подтверждается отмеченная выше возможность проводить оценку доказуемой стойкости шифра по частному значению максимума дифференциала по одному случайно взятому ключу.

Результаты экспериментов второго направления исследований иллюстрирует таблица 5.15. Здесь рассматривались те же шифра, что и в первой серии экспериментов.

Таблица 5.15

Средние значения максимумов таблиц разностей малых версий шифров

	Шифр Хейса		Мини-AES	Мини-ADE	Мини-Лабиринт	Мини-Мухомор		Мини-Калина	
Число циклов	S-блок $\delta = 8$	S-блок $\delta = 6$	S-блок $\delta = 4$	S-блок $\delta = 4$	S-блок $\delta = 4$	S-блок $\delta = 8$	S-блок $\delta = 4$	S-блок $\delta = 8$	S-блок $\delta = 4$
1	32768	16384	16384	16384	-	65536	65536	6082,56	3732,48
2	12288	4096	3036,16	3353,6	-	14187,5	5770,24	826,88	382,4
3	2326,8	439	274,24	307,2	37,5	2496,32	1802,24	24,8	19,36

	1								
4	216,803	56,964	19,326	20,54	19,04	542,72	125,53	19,04	19,14
5	65,38	26,18	19,02	19,08	19,24	46,28	29,7	19,14	19,2
6	24,108	19,108	18,812	19,24	19,04	19,48	18,88	19,14	19,36
7	19,021	19,086	-	-	19,14	-	-	-	-
8	19,16	19,1	-	-	19,24	-	-	-	-

В обеих таблицах для обозначения максимального значения перехода XOR таблицы подстановочного преобразования (S-блока) использовано обозначение, названное δ-равномерностью. Напомним его определение [219]

Определение. *Пусть F будет $n \times s$ S-блоком, где $n \geq s$. Пусть δ будет наибольшим значением в дифференциальной таблице S-блока* (*исключая первый вход в первую строку*)*, а именно*

$$\delta = \max_{\alpha \in V_n, \alpha \neq 0} \max_{\beta \in V_s} \left| \{x | F(x) \oplus F(x \oplus \alpha) = \beta\} \right|.$$

Тогда говорят, что S-блок F является дифференциально δ-равномерным (*плоским*).

Отметим также для 16-битного шифра Хейса в одном случае в качестве S-блоков (одинаковых) взят S-блок шифра MiniA (первая строка первого S-блока шифра DES). Во втором случае взят случайный S-блок. Он, однако, по дифференциальным характеристикам не уступает S-блоку BabyR.

Как уже отмечалось ранее, в шифре мини-Мухомор таблицы подстановок, участвующие в *SL*-преобразованиях сформированы случайным образом. Текущее состояние на входе функция усложнения M-8 представляется в виде двух полубайт, к каждому из которых применяется своя подстановка.

При реализации шифра мини-Калина также использованы S-блоки двух типов: с худшими дифференциальными свойствами (S-блоки шифра DES) и с высокими дифференциальными показателями (S-блоки шифра ADE).

Отметим также, что для шифра мини-Лабиринт начальное и конечное преобразования (*IT* и *FT*) считались как отдельные циклы преобразования. Поэтому в таблицах приведены данные, начиная с третьего цикла.

Как видно из приведенных результатов все рассмотренные шифры пришли после соответствующего числа циклов к установившемуся (асимптотическому) значению для данного порядка подстановок весьма близкому к теоретически обоснованному значению

$$d_{\max} = \Lambda_{m,2k^*} = 20.$$

(в таблице 6 ячейки, соответствующие выходу к асимптотическим значениям выделены серым цветом). Таблица 5 подтверждает, что вычисленные абсолютные (для выборки рассматриваемого объёма) значения максимумов полных дифференциалов оказываются действительно для всех рассмотренных шифров весьма близкими к соответствующим средним значениям максимумов полных дифференциалов. Разница между числом циклов, требуемых для выхода к асимптотическим значениям, для разных шифров составляет один – два цикла. Интересно отметить, что близким по рассматриваемым показателям к уменьшенным версиям современных шифров оказался и шифр Хейса с простейшей конструкцией линейного преобразования. Можно также подчеркнуть, что в рамках рассмотренных уменьшенных моделей шифры выходят на асимптотические значения максимумов полных дифференциалов в таком порядке: 1) Калина, 2) Лабиринт, 3) AES, 4) ADE, 5) Мухомор, 6) шифр Хейса.

Приведем в заключение для более полной информации распределение переходов XOR таблиц шифра мини-Мухомор в зависимости от числа циклов зашифрования (табл. 5.16.)

Аналогичные результаты получаются и для других версий уменьшенных шифров (табл. 5.17).

Таблица 5.16

Законы распределение переходов XOR таблиц шифра мини-Мухомор для 5-ти 10-ти циклов зашифрования

Число циклов 5 Key: 1	Число циклов 10 Key: 1
#0. 2604929979	#0. 2604884505
#2. 1302443900	#2. 1302455754
#4. 325634132	#4 325627300
6. 54278584	#6. 54284259
#8. 6783190	#8. 6781258
#10. 679866#	#10. 678321
#12. 56522	#12. 56731
#14. 4196	#14. 4086
#16. 267	#16. 245
#18. 19	#18. 16
#20. 4	#20. 2
#22. 1	
#24. 1	

Сопоставление результатов таблиц 5.16 и 5.17 с первой колонкой табл. 5.17 подтверждает, что дифференциальные свойства шифра Мухомор, а вместе с ним и других шифров асимптотически с высокой точностью повторяют дифференциальные свойства случайной подстановки соответствующей степени.

Аналогичные результаты можно найти в работах [183 и др.].

Таблица 5.17

Распределение парных разностей по числу ячеек для шифра Baby-Rijndael

Расчет	Эксперимент
#0. 2604929979	#0. 2604884505
#2. 1302484861	#2. 1302551726
#4. 325626184	#4. 325625709
#6. 54271858	#6. 54253870
#8. 6784085	#8. 6781574
#10. 678418	#10. 677785
#12 56535	#12 56793
#14. 4038	#14. 3974
#16. 252	#16. 272

#18. 14 #20. 1	#18. 17 #20. 0

Если теперь согласиться с правомерностью переноса свойств уменьшенных моделей шифров на их прототипы, то представленные результаты свидетельствуют, что дифференциальные свойства представленных на украинский конкурс шифров Калина, ADE, Мухомор и Лабиринт после 4-5 циклов зашифрования становятся близкими по свойствам случайным подстановкам. Можно сделать общий вывод о том, что современные БСШ (при полном наборе цикловых преобразований) являются случайными подстановками.

Свойства шифрующих преобразований современных блочных симметричных шифров (Rijndael-ю и других шифров, представленных на украинский конкурс, являются одним из проявлений свойств случайных подстановок, и в этом смысле шифр Rijendael и шифры, представленные на Украинский конкурс, являются эквивалентными (неразличимыми). Все они реализуют в соответствии с приведенным в работе утверждением наибольшую вероятность максимума полного дифференциала (для 128 битных версий) близкую к 2^{-120}. Этот результат следует считать обоснованным и теоретически и практически.

3.2.4 Результаты вычислительных экспериментов по определению полных дифференциалов уменьшенных моделей современных шифров для разных конструкций линейного преобразования.

В этом подразделе с помощью малых версий современных шифров демонстрируется независимость асимптотических значений максимумов полных дифференциалов от используемых в шифрах S-блоков (S-блоков с различными значениями параметра p, названного выше δ-равномерностью).

В данном случае приводятся результаты исследований дифференциальных свойств 16-битных моделей шифров Rijndael с различными типами линейных преобразований и простейшего шифра SPN структуры, предложенного еще Х. Фейстелем [60] Для таких размеров

входных блоков данных вычислительных ресурсов вполне достаточно, чтобы построить целиком таблицу XOR переходов (полных дифференциалов) сразу для всего шифра.

В табл. 5.18 представлены зависимости средних значений максимумов полных дифференциалов ($AMDP^f \cdot 2^n$) от числа циклов шифрования r алгорит ма Baby-Rijndael при использовании S-блоков с различными значениями $DP^s_{\max} = p$ и операцией MixColumns на весь текст (самого эффективного

Таблица 5.18

Значения максимумов полного дифференциала для различных S-блоков и количества циклов алгоритма Rijndael с операцией MixColumns на весь текст

Sbox / r	Sbox, Сл *p*4, F2	Sbox. *p*4 Лабир.	Sbox AES, *p*4	Sbox *p*6, F0	Sbox *p*6, F2	Sbox DES, *p*8	Sbox *p*8, F0	Sbox *p*12, F0
1	16384,00	16384,0	16384,0	24576,0	24576,0	32768,0	32768,0	49152,0
2	83,87	132,00	132,00	490,87	230,40	1152,00	1536,00	5184,00
3	20,73	**19,47**	**18,80**	25,53	35,27	70,87	139,13	146,13
4	**19,60**	18,73	19,00	**19,20**	**18,93**	**19,27**	23,93	**19,07**
5	19,13	19,47	19,47	18,93	19,40	19,00	23,87	19,00

варианта линейного преобразования: для 16-ти битного вектора B, представляющего собой набор из четырех полубайтовых значений выходов четырех

S-блоков $B = (b_0, b_1, b_2, b_3)$ [192]). Результатом линейного преобразования является 16-ти битный вектор $C = (c_0, c_1, c_2, c_3)$, определяемый с помощью матричного умножения:

$$(c_0, c_1, c_2, c_3) = b_0, \quad b_1, \quad b_2, \quad b_3 \ * \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix},$$

где операция матричного умножения выполняется в поле $GF(2^4)$ (элементы матрицы – это элементы поля $GF(2^4)$). В этом случае дополнительного преобразования (перестановки) ShiftRows не требуется.

В экспериментах для каждого шифра рассматривалась выборка из 30 различных ключей зашифрования. Сами варианты использованных S-блоков иллюстрирует табл. 5.19 (см. также [189]).

Как следует из представленных результатов во всех случаях, кроме S-блока Sbox *p*8 F0 (7-я колонка таблицы) независимо от максимального зна-

Таблица 5.19

Варианты использованных S-блоков с расшифровкой их описаний

Варианты использованных S-блоков	Расшифровка описания S-блоков
SboxAES *p*4 ⇒{A,4,3,B,8,E,2,C,5,7,6,F,0,1,9,D}; SboxDES *p*8 ⇒{E,4,D,1,2,F,B,8,3,A,6,C,5,9,0,7} (первая строка S-блока S_1 шифра DES); Sbox *p*6 F2 ⇒ {B,C,5,0,1,3,2,7,8,4,D,F,6,9,E,A}; Sbox *p*6 F0 ⇒ {4,6,F,B,E,7,5,D,9,C,1,0,3,8,A,2}; Sbox *p*12 F0 ⇒ {8,3,1,9,A,B,E,C,5,D,F,2,0,4,7,6}; Sbox *p*8 F0 ⇒ {C,9,4,6,8,E,D,5,3,F,B,0,A,2,1,7}; Sbox, Сл. *p*4 F2 ⇒ {D,E,B,5,4,2,1,F,0,9,6,A,7,C,8,3}; Sbox. *p*4 Лаб. ⇒	*p*X – X максимальное значение в дифференциальной таблице S-блока; FY – Y количество фиксированных точек ($S(x) = x$). Отсутствие FY в описании S-блока эквивалентно F0.

{B,8,6,4,A,0,D,2,C,5,1,E,3,F,9,7}.	

чения таблицы XOR разностей p S-блоков все шифры приходят к одному и тому же среднему значению максимума полного дифференциала, характерному для случайной подстановки соответствующей степени [43,44]. Можно также отметить, что S-блок из 7-ой колонки табл. 5.4 тоже приходит к асимптотическому значению на четвертом цикле, только оно оказывается несколько больше асимптотического значения для случайной подстановки (≈24). ″Хорошие″ S-блоки с минимальным значением максимума $DP_{\max}^{f}$ равным 4-ём имеют преимущество перед остальными S-блоками всего лишь один цикл. Поэтому связь дифференциальных показателей S-блоков $DP_{\max}^{f}$ с показателями стойкости к дифференциальному (и линейному) криптоанализу всего шифра *MADP* (*MALHP*), которую считают, что нашли авторы цитируемых работ, становится далеко не очевидной. Из других результатов наших исследований здесь приводятся данные по анализу дифференциальных показателей шифра проф. Хейса [210], построенного по идеям Х. Фейстеля, которые иллюстрирует табл. 5.20.

Каждая колонка таблицы соответствует использованию S-блоков (одинаковых) со свои ми значениями параметра p. Результаты и этого эксперимента свидетельствуют, что все варианты шифров (отличающихся S-блоками) приходят через определенное число циклов опять к установившемуся значению, характерному для случайной подстановки, правда, Sbox p8 F0 снова ″пошел своей дорогой″.

Таблица 5.20

Значения максимумов полного дифференциала для различных S-блоков и количества циклов алгоритма Хейса

Sbox / r	Sbox,Сл ***p*4,** F2	Sbox. ***p*4** Лабир.	Sbox AES, ***p*4**	Sbox, Сл, ***p*6,** F0	Sbox ***p*6**, F2	Sbox DES, ***p*8**	Sbox ***p*8,** F0	Sbox ***p*12,** F0
1	16384,0	16384,0	16384,0	24576,0	24576,0	32768,0	32768,0	49152,0
2	4096,00	4096,00	4096,00	6144,00	6144,00	12288,0	8192,0	15552,0
3	443,87	1616,00	2036,27	1920,00	2802,40	2303,33	3432,00	1587,20
4	54,60	362,87	596,00	601,20	649,33	222,27	1184,47	613,13

5	25,07	88,47	190,33	148,93	292,93	64,13	457,07	265,73
6	**19,00**	24,93	77,47	50,00	71,47	24,80	178,40	104,87
7	19,33	**19,53**	35,87	22,00	32,00	**18,80**	87,33	46,87
8	19,20	18,93	21,07	**19,07**	19,67	18,80	39,93	23,87
9	18,97	19,27	**19,27**	18,87	**18,93**	19,00	**24,60**	**19,13**
10	19,45	19,02	19,33	19,27	19,33	18,93	24,27	19,00

Далее приводятся результаты экспериментов по применению при построении различных версий шифра Baby - Rijndael других вариантов линейных преобразований. Так, в табл. 5.21 представлены поцикловые значения полных дифференциалов для SPN шифров с линейным преобразованием MixColumns и ShiftRows GF(2^4): для 16-ти битного вектора B, представляющего набор из четырех полубайтовых значений выходов четырех S-блоков $B = (b_0, b_1, b_2, b_3)$, результатом линейного преобразования является 16-ти битный вектор $C = (c_0, c_1, c_2, c_3)$, определяемый с помощью матричного умножения

$$\begin{pmatrix} c_0 & c_2 \\ c_1 & c_3 \end{pmatrix} = \begin{pmatrix} 3 & 2 \\ 2 & 3 \end{pmatrix} * \begin{pmatrix} b_0 & b_2 \\ b_1 & b_3 \end{pmatrix} \text{ GF}(2^4),$$

Таблица 5.21

SPN шифр с линейным преобразованием MixColumn и ShiftRows F(2^4)

Sbox / r	Sbox AES ***p*4**	Sbox. ***p*4** Лабир.	Sbox, Сл ***p*4** F2	Sbox, Сл ***p*6** F0	Sbox DES ***p*8**	Sbox ***p*8**F0	Sbox, Сл ***p*12** F0
1	16384,00	16384,00	16384,00	24576,00	32768,00	32768,00	49152,00
2	2952,53	3515,73	3072,00	4625,07	4744,53	8192,00	10240,00
3	282,67	318,93	373,33	612,27	748,80	1382,40	768,00
4	**19,33**	**19.27**	**19,20**	22,60	26,87	53,87	41,67
5	19,33	19,07	19,27	**18,73**	**19,00**	**24,13**	**18,80**
6	19,00	19,40	19,00	19,00	19,00	23,80	19.27

где операция матричного умножения выполняется в поле GF(2^4). В этом случае для дополнительного перемешивания результатов матричного произведения используется операция ShiftRows.

Применение менее эффективного линейного преобразования не изменяет картину. Опять через определенное число циклов (четыре или пять) независимо от варианта используемых подстановок наблюдается выход к установившемуся (асимптотическому) значению (Sbox *p*8F0 снова дает индивидуальный результат, повторяющий по характеру предыдущий).

В последнем из представленных здесь экспериментов взята уменьшенная модель шифра Rijndael с линейным преобразованием MixColumn и ShiftRows GF(2^8). В этом случае для 16-ти битного вектора B', представляющего набор из двух байтовых значений выходов четырех S-блоков $B' = (b'_0, b'_1)$ ($b'_0 = (b_0, b_1)$, $b'_1 = (b_3, b_4)$) результатом линейного преобразования является 16-ти битный вектор $C' = (c'_0, c'_1) = (c_0, c_1, c_2, c_3)$, определяемый с помощью матричного умножения:

$$\begin{pmatrix} c'_0 \\ c'_1 \end{pmatrix} = \begin{pmatrix} 01 & 03 \\ 03 & 01 \end{pmatrix} * \begin{pmatrix} b'_0 \\ b'_1 \end{pmatrix} \mathrm{GF}(2^8),$$

где операция матричного умножения выполняется в поле GF(2^8) (элементы матрицы – это элементы поля GF(2^8)). В этом случае для дополнительного перемешивания (перестановки) результатов матричного произведения используется также операция StiftRows (в данном случае байты c_0 и c_1 просто меняются местами). Табл. 5.22 иллюстрирует результаты этого эксперимента.

Интересно отметить, что S-блок Sbox*p*8 F0 в этом случае тоже показал асимптотическое значение, соответствующее случайной подстановке.

Приведенные результаты свидетельствуют, что концепция оценки доказуемой стойкости, развиваемая в приведенных во введении и многих других

работах, ошибочна! Во всех рассмотренных примерах показатели стойкости шифров не зависят от подстановок, использованных при их построении, а

зависят только от битового размера входа в шифр. Подстановки влияют лишь на динамику

Таблица 5.22

Значения максимумов полного дифференциала для SPN шифра с линейным преобразованием MixColumn и ShiftRows GF(2^8) и S-блоками с различными показателями δ-равномерности (параметра ***p***)

Sbox / *r*	SboxAES ***p*4**	Sbox. ***p*4** Лабир.	Sbox, Сл. ***p*6**	Sbox, Сл. ***p*6**	SboxDES ***p*8**	Sbox ***p*8**F0	Sbox, Сл ***p*12** F0
1	16384,00	16384,00	24576,00	24576,00	32768,00	32768,00	49152,00
2	4983,47	1024,00	5102,93	4522,67	3635,20	6144,00	3072,00
3	647,47	32,73	530,13	833,07	542,93	301,47	166,73
4	42,40	**19,33**	34,00	66,53	22,13	25,00	**19,07**
5	20,67	18,67	**19,60**	40,93	**19,07**	**19,00**	18,87
6	**19,20**	19,00	19,33	**19,27**	19,27	19,27	19,00
7	19,13	18,00	19,20	19,20	19,00	19.13	19,13

перехода к асимптотическому значению. Этот результат, следующий из большого числа экспериментов со случайными подстановками и малыми моделями шифров, будет сохраняться и для больших версий шифров (и не только для шифров с SPN структурой!).

3.2.5 Результаты вычислительных экспериментов по определению максимумов смещений линейных корпусов уменьшенных моделей современных шифров

В этом подразделе продолжается изложение результатов исследований малых моделей шифров, и представляются свидетельства тому, что и по линейным показателям большие шифры при полном числе циклов шифрования становятся случайными подстановками.

До недавнего времени работы в этом направлении сдерживались существенно увеличивающимся объёмом вычислений, требующимся при построении линейной аппроксимационной таблицы (ЛАТ). Даже для малых

версий шифров построение ЛАТ при использовании традиционных методов увеличивало сложность вычислений в 2^{16} раза по сравнению с построением дифференциальной таблицы (для вычисления которой нужно было просчитать значения заполнений 2^{32} её ячеек).

Дело существенно продвинулось вперед в связи с найденным в Интернете описанием быстрого алгоритма просчета ЛАТ [220]. В нашей работе [39] также приводится его описание. Вычислительная сложность быстрого алгоритма для битового входа подстановки из n бит в n битный выход оценивается значением $O(2n)$ (вычисление производится поколоночно за время O($2n$) для одного элемента таблицы).

Далее опять за основу берётся вычисление показателя стойкости не в виде максимального значения вероятности смещения таблицы линейных аппроксимаций, а как и в предыдущем случае определяется действительное значение максимума заполнения по всем ячейкам линейной таблицы, а точнее среднее значение по множеству ключей зашифрования, использованных в эксперименте.

В соответствии с (**)

$$\overline{l_{\max}} = \sqrt{2^{n-1} \cdot \underset{k}{ave} LP^{f}_{\max}(Гx \to Гy) AMLHP^{f}} = \sqrt{ALHMP \cdot 2^{n-1}}.$$

Здесь $\overline{l_{\max}}$ обозначает средне (по множеству ключей зашифрования использованных в эксперименте) значение максимумов смещений таблиц линейных аппроксимаций.

Далее приводятся результаты применения для построения ЛАТ быстрого алгоритма, с помощью которого удалось более полно исследовать линейные свойства всех интересующих нас шифров [220]. Они представлены в табл. 5.23.

В процессе выполнения первой серии экспериментов в каждом случае были выполнены расчеты для множества из 100 случайно сгенерированных ключей шифрования и выполнено последующее усреднение полученных результатов по этому множеству.

Как следует из представленных результатов, с большим уровнем доверия

подтверждается выдвинутое в качестве гипотезы положение, в соответствии с которым блочные симметричные шифры после определенного небольшого числа начальных циклов шифрования по своим линейным свойствам повторяют показатели случайных подстановок соответствующего порядка. Все шифры после

Таблица 5.23

Математические ожидания максимальных смещений линейных корпусов ($\overline{l_{\max}}$) уменьшенных версий шифра Mini-AES и шифров, представленных на украинский конкурс вместе со значениями среднеквадратических отклонений

Шифр / *r*	Mini-AES	Mini-ADE	Mini-Labyrinth	Mini-Kalina	Mini-Muhomor
1	16384	16384	-	9349,15±237,02	16384±0
2	9164,8±120,7	9093,10±94,37	-	3545,8±83,93	11818±1677,37
3	3658,2±65,8	3509,8±62,37	1069,8±38,41	830,55±83,93	5900±804,556
4	827,24±7,25	828,56±7,58	826,36±6,6	820,14±6,96	1611±369,464
5	821,34±6,16	820,52±5,48	820,8±5,44	823,28±4	820±42,5809
6	821,68±7	819,92±5,81	822,88±7,25	825,04±6,34	812±21,3198

четырёх циклов шифрования приходят к установившемуся значению максимума линейного корпуса, характерному для случайных подстановок

соответствующей степени (шифр Мухомор поле пяти). Отличие от расчетного значения (750) оказывается меньшим 25%.

Еще одна серия экспериментов была проведена для определения абсолютных значений максимумов линейных корпусов для выборки из 100-а ключей зашифрования. Результаты этих экспериментов обобщены в табл. 5.24.

Таблица 5.24

Абсолютные значения максимумов линейных корпусов мини версий шифров, представленных на украинский конкурс

Шифр / *r*	Mini-AES	Mini-ADE	Мини-Лабиринт	Мини-Калина	Мини-Мухомор
1	16384	16384	-	12288	16384
2	10240	12288	-	4480	12420
3	4352	4352	1444	936	6912
4	940	932	914	912	1860
5	900	896	902	900	1364
6	922	880	906	912	836

По всем шифрам можно сделать вывод, что абсолютные значения максимумов линейных корпусов отличаются от средних значений максимумов линейных корпусов менее чем на 15 %. Это означает, что в качестве максимальных значений смещений при оценке стойкости шифра можно использовать текущие всем шифрам можно сделать вывод, что абсолютные значения максимумов линейных корпусов отличаются от средних значений максимумов линейных корпусов менее чем на 15 %. Это означает, что в качестве максимальных значений смещений при оценке стойкости шифра можно использовать текущие значения измерений этого показателя. Шифр Мини-Мухомор приходит к максимальным значениям смещения на 1-2 цикла позже. Напомним, что в работе [180], посвященной описанию разработанной уменьшенной модели этого шифра отмечалось, что не все операции шифра поддаются масштабированию, и при построении уменьшенной модели операция SL преобразования (в большой версии четыре

байтовых S-блока с последующим МДР линейным преобразованием) была заменена полубайтовой подстановкой. Может здесь дело и не в этом, а в самой конструкции преобразования. В общем, здесь требуется дополнительная проверка этого результата на большом шифре

Была выполнена также серия экспериментов для рассмотренных шифров, когда в них используются S-блоки с различными показателями нелинейности. Отобранные конструкции S-блоков представлены в табл. 5.25.

Таблица 5.25

Полубайтовые подстановки (S-блоки) с различными показателями нелинейности

нелинейность	S-блок
0	S_1 = {12, 13, 5, 1, 10, 11, 6, 2, 14, 3, 7, 15, 4, 0, 8, 9}
2	S_2 = {5, 0, 13, 6, 4, 8, 2, 3, 9, 1, 15, 10, 12, 14, 7, 11}
2	S_3 = {10, 4, 5, 8, 2, 15, 7, 0, 14, 9, 11, 12, 6, 13, 1, 3}
4	S_4 = {2, 5, 0, 9, 3, 14, 4, 1, 10, 11, 8, 15, 7, 12 ,6, 13}
4	S_5 = {10, 4, 3, 11, 8, 14, 2, 12, 5, 7, 6, 15, 0, 1, 9 ,13}

Представленные в таблице подстановки были использованы в качестве S-блоков во всех исследуемых малых моделях шифров (в каждом шифре использовались S-блоки с одинаковыми показателями нелинейности).

В процессе вычислительных экспериментов определялись поцикловые средние значения максимумов смещений линейных корпусов ($\sqrt{ALHMP \cdot 2^{n-1}}$) уменьшенных моделей трех шифров из представленных на украинский конкурс (шифры Лабиринт, Мухомор и Калина) и уменьшенной модели шифра Rijndael.

Для каждого шифра с фиксированным числом циклов шифрования выполнялось построение линейных корпусов (таблиц линейных аппроксимаций) для 30 различных ключей зашифрования, сгенерированных случайным образом, определялись максимальные значения смещения для каждой из таблиц, а затем результаты усреднялись. Строилась зависимость средних значений максимумов смещений линейных корпусов от числа циклов зашифрования.

В табл. 5.26-5.29 представлены поцикловые значения средних значений максимумов смещений таблиц линейных аппроксимаций (линейных корпусов) для рассмотренных в работе шифров.

Таблица 5.26

Математическое ожидание максимальных смещений линейных корпусов шифра мини-Лабиринт со значениями среднеквадратического отклонения для соответствующих S-блоков

Число циклов шифрования *r*	Показатели нелинейности S-блоков, использованных в шифре				
	0	2	0	2	4
1	10751,6±1934	4385,6±1166	3200±494	4300±1774	3178±777
2	1910,8±460	1182,2±162	1219,6±312	1043,60±252	980±193
3	1032,8±190	839,2±34	864,4±55	831,6±27	825,4±14
4	810±15	813,6±13	822,4±17	811,4±11	825,6±23
5	816,2±17	827,8±16	833,4±25	809±15	817,2±11
6	826,4±24	832±32	814,8±24	828,2±35	824±21
7	817,8±18	812±13	811±17	837,4±22	823,4±30
8	835,4±16	813,6±19	822,6±23	815,4±20	833,6±35
9	819,4±23	812,6±14	815,8±26	820,4±15	824,8±24
10	820,6±20	815,4±22	814,4±20	808±21	819±17

Как следует из представленных результатов во всех случаях, независимо от значения показателя нелинейности S-блоков все шифры приходят к одному и тому же среднему значению максимума смещения линейного корпуса, ха-

Таблица 5.27

Математическое ожидание максимальных смещений линейных корпусов шифра мини-Калина со значениями среднеквадратического отклонения для соответствующих S-блоков

Число циклов шифрова	Показатели нелинейности S-блоков, использованных в шифре				
	0	2	0	2	4

ния *r*					
1	21845,3±5565	18090,6±1043	16497,7±2222	11491,5±1825	9671,1±867
2	9557,33±1231	7288,8±758	5432,88±693	3968±307	3370,6±301
3	2823,55±436	1661,22±168	1394,66±219	862,66±50	836,8±15
4	1307,77±216	822,55±30	815,33±17	826,44±23	832,2±21
5	818,88±21	796±20	818,66±11	811,77±12	838,6±21
6	824,22±15	808,66±29	840,44±28	816,88±13	835,5±33
7	810±18	811±30	820,88±32	824,44±23	821,5±22
8	819,77±21	808,88±29	834,66±43	829,33±16	827,3±18
9	808±20	810,66±11	811,33±15	827,77±22	813,3±21
10	836,44±39	821,55±26	816±11	817,33±16	834±28

Таблица 5.28

Математическое ожидание максимальных смещений линейных корпусов шифра мини-Мухомор со значениями среднеквадратического отклонения для соответствующих S-блоков

Число циклов шифрования *r*	Показатели нелинейности S-блоков, использованных в шифре				
	0	2	0	2	4
1	32768±0	32768±0	32768±0	32768±0	32768±0
2	16896±2101	15616±1511	14364,4±1372	10951,1±1112	12839,3±1031
3	7082,6±1020	6553,3±663	6599,1±678	6200,8±841	6400±697
4	2190,4±364	2022,6±381	2052,4±395	1663,1±222	1797,6±347
5	1035,7±33	828,8±48	857,5±40	866±47	837,8±47
6	828,6±25	823,1±24	820±34	818,6±15	815,6±24
7	837,5±26	821,7±22	824,4±24	821,7±20	817,2±20
8	849,5±33	810,2±12	830,8±21	834±19	815,8±15
9	810,2±13	815,7±25	825,7±19	806,6±13	815,5±15
10	808,6±8	819,3±21	836,6±27	813,5±18	810±17

рактерному (весьма близкому) для случайной подстановки соответствующей степени [43]. Видно по всем рассмотренным шифрам, что для перехода к асимптотическому значению смещения требуется 4-5 циклов. Видно, что использо-

Таблица 5.29

Математическое ожидание максимальных смещений линейных корпусов шифра мини-Rijndael со значениями среднеквадратического отклонения для соответствующих S-блоков

Число циклов шифрования *r*	Показатели нелинейности S-блоков, использованных в шифре				
	0	2	0	2	4
1	32768±0	24576±0	32768±0	24576±0	16384±0
2	24576±0	7808±0	10368±0	3584±0	2048±0
3	12288±0	2527,1±54	2304±0	1128±107	851,5±48
4	5233,1±58	891,7±58	905,7±29	829,3±22	814,2±14
5	2040,2±103	822,6±24	831,3±29	817,7±11	828,6±20
6	1077,7±35	825,3±19	832,2±21	820,8±23	827,7±31
7	839,7±34	815,7±16	826,4±21	822,6±27	822,2±28
8	821,7±11	810,8±21	822,8±20	819,3±26	803,7±12
9	830±26	811,1±15	821,7±34	830,4±27	810,4±18
10	811,3±18	834,4±29	806,6±11	808,4±14	823,1±27

вание S-блоков с высокими показателями нелинейности даёт выигрыш в динамике выхода к асимптотическому значению в пределах одного цикла.

Здесь можно повторить вывод, сделанный в работе [39] по отношению к дифференциальным показателям уменьшенных моделей шифров, перенеся теперь его на линейные показатели: если согласиться с правомерностью переноса свойств уменьшенных моделей шифров на их прототипы, то изложенные в работе результаты свидетельствуют о том, что линейные свойства представленных на украинский конкурс шифров – Калина, ADE, Мухомор и Лабиринт после 4-5 циклов шифрования повторяют с большой точностью свойства случайных подстановок. Можно сделать общий вывод о

том, что современные БСШ (при полном наборе цикловых преобразований) действительно являются случайными подстановками.

Из полученных данных был сделан важный вывод о том, что и в этом случае все шифры (их уменьшенные модели) после определенного (небольшого числа циклов принимают одно и то же (″асимптотическое″) значение максимума линейного корпуса, причем эти значения совпадают с соответствующими значениями максимумов смещений таблиц линейных аппроксимаций случайных подстановок соответствующей степени [44].

Самый главный результат изучения уменьшенных моделей состоит в том, что общепринятая точка зрения, разрабатываемая во многих работах и заключающаяся в том, что линейные и дифференциальные свойства шифров непосредственно связаны со свойствами S-блоков, используемых при их построении, оказалась не верной или не совсем верной [57]. На самом деле результирующие (т.е. получающиеся при использовании полного набора цикловых преобразований) показатели стойкости шифров определяются практически только размером битового входа в шифр.

Второй важный вывод, следующий из представленных результатов, сводится к тому, что показатели стойкости больших (полных реализаций) шифров к атакам дифференциального и линейного криптоанализа (таких шифров, как Rijndael и многих других известных шифров, а также шифров Лабиринт, Калина, Мухомор, ADE, представленных на украинский конкурс по выбору национального стандарта шифрования) могут быть получены расчетным путем [57].

Несмотря на очевидные соображения, связанные с тем, что случайные показатели будут только улучшаться с ростом степени подстановок, все же оставались скептики, которые требовали более веских доказательств положения о том, что большие шифры, как и их малые модели с увеличением числа циклов приобретают свойства случайной подстановки.

Сегодня уже проведен цикл исследований и в этом направлении. Нами уже получены результаты, свидетельствующие о том, что законы

распределения переходов таблиц XOR разностей и смещений таблиц линейных аппроксимаций больших шифров повторяют законы распределения вероятностей случайных подстановок соответствующей степени [38 и др.]. Результаты исследований в этом направлении будут представлены далее в отдельном разделе.

Этот важный результат, приведенный и в первом разделе работы, позволяет легко определить интересующие нас показатели. Для этого достаточно воспользоваться упрощенными расчетными соотношениями, приведенными в подразделе 4.11 работы. Здесь далее приводятся итоговые результаты для 128-битного шифра Rijndael.

Для среднего значения максимумов дифференциальной вероятности (максимума полного дифференциала) 128-битного шифра получаем с использованием приближённой аппроксимации результат: $AMDP^R = \frac{128+3}{2^{128}}$.

Заметим, что оценка, полученная автором работы [43] для шифра Rijndael $MADP^R \leq 2^{-96}$, отличается от приведенной в 2^{25} раза. Она действительно может рассматриваться как грубая оценка сверху, но это скорее случайность (не лишенная интуитивных предчувствий), чем строго доказанный результат. По методике [43] для шифров с подстановками, имеющими большие значения максимумов p и q, результаты должны заметно измениться в худшую сторону, чего на самом деле не происходит.

Среднее значение максимального смещения таблицы линейных аппроксимаций (максимальное значение линейного корпуса (оболочки)) для шифров с n-битным размером входного блока данных может быть оценено приближённым соотношением $\left(\frac{3}{2}\right)^n$. Для среднего значения максимума вероятности линейного корпуса 128-битного шифра приходим к результату

$AMLHP^R \leq \left(\frac{\left(\frac{3}{2}\right)^{128}}{2^{127}} \right)^2 = 2^{-104}$ (реальное значение максимума смещения таблицы линейных аппроксимаций, как показано выше, лежит в границах $2^{-120} \leq LD^f_{\max} \leq 2^{-104}$ (см. раздел 4)), а более точно близко к 2^{-118}.

Представленными результатами исследований подтверждено, что свойства шифрующих преобразований современных блочных симметричных шифров (Rijndael-я и других шифров, представленных на украинский конкурс), являются одним из проявлений свойств случайных подстановок, и в этом смысле шифр Rijendael и шифры, представленные на Украинский конкурс, являются эквивалентными (одинаковыми по стойкости). Все они реализуют в соответствии с приведенным в работе утверждением и расчетным соотношением из нашей работы [21] наибольшую вероятность максимума линейного корпуса (для 128 битных версий) близкую к 2^{-118}. Этот результат следует считать обоснованным и теоретически и практически.

В отношении адекватности моделей малых шифров. Здесь представлено несколько моделей шифра baby-Rijndael, отличающиеся конструкциями основного (принципиального) преобразования шифра Rijndael − линейного преобразования. Была поставлена задача выбора наиболее подходящей (более полно повторяющей свойства оригинальной версии) конструкции модели. Результаты, однако, свидетельствуют о том, что различные в данном случае варианты реализации МДР преобразования приводят к шифрам с весьма близкими дифференциальными показателями. Полагается, что эта инвариантность к точному повторению в модели отдельных операций прототипа будет сохраняться и по отношению к другим показателям.

Ну а в отношении сохранения в большом шифре свойств уменьшенной модели можно лишь отметить, что при нашем подходе говорится о сохранении свойств и показателей случайной подстановки. Представляется

очевидным, что при увеличении битового размера входа в шифр (с увеличением степени подстановки) ее соответствие асимптотическим законам распределения вероятностей (по инверсиям, возрастаниям и циклам), а также законам распределения вероятностей переходов таблиц полных дифференциалов и смещений таблиц линейных корпусов будет только повышаться. Более обстоятельному обоснованию справедливости повторения свойств случайных подстановок большими шифрами будет посвящен отдельный раздел работы.

Общим важным итогом раздела является обоснование новой идеологии оценки стойкости блочных симметричных шифров к атакам линейного и дифференциального криптоанализа, позволяющей в отличие от известных подходов получить не оценочные (граничные), а точные значения максимумов полных дифференциалов и смещений линейных корпусов больших шифров.

4 РАЗРАБОТКА НОВОЙ МЕТОДОЛОГИИ ОЦЕНКИ СТОЙКОСТИ СИММЕТРИЧНІХ КРИПТОПРЕОБРАЗОВАНИЙ

4.1. Сущность и общая характеристика предлагаемого подхода

Излагаемые далее соображения и результаты строятся исходя из развиваемого нового подхода в теории и методах криптоанализа, ориентированного с одной стороны на использование при определении ожидаемых результатов стойкости больших шифров результатов анализа уменьшенных их версий, а с другой, − уточненной на основе изучения свойств и показателей случайных подстановок и уменьшенных моделей шифров, рассматриваемых как подстановочные преобразования, концепции (новой идеологии) определения показателей стойкости БСШ к атакам дифференциального и линейного криптоанализа.

В соответствии с развиваемым подходом, для преодоления трудностей анализа полномасштабных моделей (алгоритмов) шифрования было предложено пойти по пути разработки и исследования уменьшенных моделей прототипов [41,42,56,57 и др.], для которых имеющихся вычислительных ресурсов оказывается уже вполне достаточно даже для решения задач переборного типа. Наши проработки [179-187 и др.] показали, что большое число хорошо известных алгоритмов шифрования допускают ″масштабирование″. Удается построить уменьшенные модели, которые сохраняют все свойства своих прототипов и позволяют решить большой объём исследований по анализу и сравнению показателей стойкости малых, а в конечном итоге, и больших версий шифров.

Самый главный результат изучения уменьшенных моделей состоит в том, что общепринятая точка зрения, разрабатываемая во многих работах и состоящая в том, что линейные и дифференциальные свойства шифров непосредственно связаны со свойствами S-блоков, используемых при их построении, оказалась не верной. На самом деле результирующие (т.е. получающиеся при использовании полного набора цикловых

преобразований) показатели стойкости шифров определяются практически только размером битового входа в шифр [42]. Далее это положение будет изложено более основательно.

Второй важный вывод, следующий из выполненных исследований, приводится к тому, что показатели стойкости больших (полных реализаций) шифров к атакам дифференциального и линейного криптоанализа, таких шифров, как Rijndael и многих других известных шифров, а также шифров Лабиринт, Калина, Мухомор, ADE [48-51], представленных на украинский конкурс по выбору национального стандарта шифрования, могут быть получены расчетным путем [42,57].

Сама сущность новой идеологии оценки стойкости блочных симметричных шифров к атакам дифференциального и линейного криптоанализа в концентрированном виде сформулирована следующим образом [42,57]:

Все современные блочные шифры[1] *через определенное число циклов независимо от используемых в шифрах S-блоков* (*конечно, здесь речь идет не о вырожденных их конструкциях*) *приобретают свойства случайных подстановок, т.е. по комбинаторным показателям* (*числу инверсий, возрастаний и циклов*)*, а также по законам распределения переходов таблиц XOR разностей* (*полных дифференциалов*) *и законам распределения смещений таблиц линейных аппроксимаций* (*линейных корпусов*) *повторяют соответствующие показатели случайных подстановок. В результате значения максимумов полных дифференциалов и линейных корпусов могут быть определены расчетным путем из формул для законов распределения вероятностей переходов XOR таблиц и смещений таблиц линейных аппроксимаций случайных подстановок соответствующей степени.*

При этом, проверка показателей случайности больших шифров может быть выполнена на основе разработки и последующего анализа

[1] Здесь шифр DES не попадает в число современных шифров.

показателей случайности уменьшенных моделей, допускающих проведение вычислительных экспериментов в приемлемые (*реальные*) *сроки.*

Малые модели шифров, повторяющие своих прототипов, позволяют оценить не только средние значения максимумов таблиц дифференциальных вероятностей (AMDP), и средних значений максимумов линейных вероятностей (AMLP) для ограниченного множества ключей, но и решить задачу определения (*проверки*) *абсолютного значения максимума по полному множеству ключей*[2].

Практически в соответствии с предлагаемой методологией процесс оценки доказуемой стойкости шифра к атакам дифференциального или линейного криптоанализа состоит из следующих этапов:

1. Строится малая (16-ти битная) модель шифра;

2. Выполняется проверка соответствия оговоренных показателей случайности малой модели шифра соответствующим показателям случайной подстановки (проверяется соответствие комбинаторных показателей шифра с полным числом циклов свойствам случайной подстановки, и определяется для полного набора циклов шифра распределение переходов таблиц XOR разностей (полных дифференциалов) и смещений таблиц линейных аппроксимаций (линейных корпусов) на одном случайно взятом ключе зашифрования). Делается проверочный просчёт дифференциальных и линейных показателей на втором ключе (или большем числе ключей) зашифрования;

3. Дополнительно может быть поставлена задача определения абсолютных максимумов таблиц XOR разностей и таблиц линейных аппроксимаций при использовании всех ключей зашифрования (поиск слабых ключей);

4. В случае, если малая модель шифра на полном наборе циклов укладывается в границы случайной подстановки степени 2^{16}, в качестве показателей доказуемой безопасности большого шифра берутся значения

[2] Показатели *AMDP*, *AMLP* (*AMLHP*) будут определены позже.

максимумов таблиц XOR разностей и максимумов смещений таблиц линейных аппроксимаций случайной подстановки соответствующей степени, которые легко пересчитываются в значения максимальных вероятностей (либо берутся соответствующие результаты из заранее заготовленных таблиц или заранее выполненных расчётов по определению *AMDP* и *AMLHP*).

5. Полученные значения максимальных вероятностей (*AMDP* и *AMLHP*) берутся в качестве показателей доказуемой безопасности (стойкости) шифра.

Далее может быть поставлена задача оценки эффективности рассматриваемого решения.

Для решения этой задачи необходимо определить поцикловые характеристики (распределения) полных дифференциалов и линейных корпусов. Далее определяются конкретные значения чисел циклов зашифрования, необходимых для достижения шифром асимптотических значений максимумов полных дифференциалов и максимумов линейных корпусов. Остаётся выполнить сравнение полученного результата с эталонным или альтернативным решением, что позволяет принять решение в пользу рассматриваемого шифра или эталона (альтернативного предложения).

Материал, излагаемый далее, посвящён теоретическому и экспериментальному обоснованию изложенного подхода.

4.2 Разработка уменьшенных моделей блочных симметричных шифров. Особенности реализации.

Как показали наши исследования, многие современные шифры допускают масштабирование [40, 180-187 и др.]. В частности это легко удается сделать для шифра Rijndael. Кстати, в интернете можно найти варианты уменьшенных моделей этого шифра [188 и др.], предлагаемые авторами для учебных целей. Так вот, уменьшенные модели шифров нами

используются в первую очередь не для учебных целей, а для выполнения исследований показателей стойкости их больших прототипов.

Были разработаны малые модели большого числа современных шифров. Среди них такие шифры как Rijndael (AES), все шифры, представленные на украинский конкурс: Калина, Мухомор, Лабиринт, ADE, а также целый набор других шифров: DES, ГОСТ 28147-89, IDEA, FOX, Anubis, Camellia и ряд других.

Практически во всех случаях удалось построить уменьшенные модели, которые хорошо сохранили все свойства своих прототипов и позволили решить многие задачи анализа и сравнения показателей стойкости соответствующих больших версий. Правда, позднее стало ясно, что можно получить нужные результаты, используя и большие версии шифров (сами прототипы). Но об этом позже. Возникающий сразу вопрос об адекватности малых моделей шифров своим прототипам будет рассмотрен отдельно.

Мы приведем здесь краткие описания части из разработанных уменьшенных моделей шифров, которые станут предметом исследований настоящего раздела. Во всех представленных далее разработках за основу взят размер битового входа в шифры равный 16-ти битам. Это максимально возможный размер, позволяющий вычислительно реализовать большинство из известных к сегодняшнему времени атак (методов) криптоанализа.

Шифр Mini-AES. Описание уменьшенной модели этого шифра можно найти в Интернете [188], которая представляет собой уменьшенную 4-х цикловую версию шифра AES [84]. Для получения 16-битной модели 128-битный AES нужно уменьшить в 8-мь раз. В модели реализованы все операции прототипа, только при масштабировании размер S-блоков взят равным 4-ём битам (восьмикратное уменьшение байтового S-блока приводит к однобитному вырожденному преобразованию) и соответственно число S-блоков в уменьшенной модели взято равным 4-ём. Линейное преобразование (операции MixColomn и ShiftRow), которое в оригинальной разработке осуществляет перестановку (сдвиг) строк и перемешивание столбцов, причем

перемешивание столбцов в оригинальной разработке осуществлено на основе умножения выходов четверок смежных байтовых S-блоков на порождающую матрицу МДР[3] кода над полем $\mathbf{GF}(2^8)$, в уменьшенной версии заменено на умножение выходов двух смежных полубайтовых S-блоков на уменьшенную в два раза матрицу МДР кода над полем $\mathbf{GF}(2^4)$.

В шифре Mini-AES в качестве S-блоков использована одна и та же таблица подстановок, значения которой задаются первой строкой первого S-блока шифра DES. В наших разработках [189] рассмотрены варианты построения уменьшенных (полубайтовых) S-блоков и по идеологии, использованной разработчиками AES [31].

Наконец, в наших разработках реализованы программы, позволяющие менять число циклов шифрования от 1-го до 8-ми. Схема разворачивания ключей повторяет идеи, реализованные в прототипе.

Шифр Mini-ADE. Алгоритм ADE был разработан на основе шифра AES [63]. Разработчики постарались обеспечить повышение стойкости шифра AES к алгебраическим атакам, для чего ввели в шифрующее преобразование механизмы динамического управления промежуточными преобразованиями. С этой целью в структуру цикловых преобразований (не затрагивая принципиальной основы использованных в AES решений), включающих в AES операции рассеивания, сдвига и нелинейной побайтовой замены, введены ключезависимые параметры, позволяющие реализовать динамическое изменение результатов каждой из операций в зависимости от текущего значения ключевых бит.

В частности, в шифре ADE используются изменяемые таблицы блоков замены, формируемые с помощью дополнительно введенного параметра $\gamma \in \mathbf{GF}(2^8)$, который определяется битами расширенного мастер-ключа. Идея этого преобразования реализована и в шифре Baby-ADE [184], только оно отмасштабировано соответственно размеру 16-битного состояния.

---

[3] В англоязычных публикациях матрица МДР обозначается MDS (см. предыдущий подраздел).

В результате в качестве S-блоков (операции SubByte) выступает изменяемая матрица подстановок, которая строится с помощью вычисления муль-типликативно обратного элемента $(a \cdot \gamma)^{-1} \in \mathbf{GF}(2^4)$ с последующим выполнением аффинного преобразования

$$b = \beta^T = M(a \cdot \gamma)^{-1} + \beta .$$

Здесь уже $a = a_0, a_1, a_2, a_3$ и $b = b_0, b_1, b_2, b_3$ – четырехбитные векторы (полубайты матрицы состояний), M – квадратная невырожденная матрица 4×4:

$$M = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix},$$

β – 4-х битный вектор ($\beta \in \mathbf{GF}(2^4)$):

$$\beta^T = (1 \quad 0 \quad 1 \quad 0).$$

Каждое значение выхода подстановки $b = b_0, b_1, b_2, b_3$ зависит как от входного состояния $a = a_0, a_1, a_2, a_3$, так и от случайного вектора $\gamma = \gamma_0, \gamma_1, \gamma_2, \gamma_3$, который задается значением циклового ключа. В результате осуществляется криптографическое преобразование данных, при котором происходит динамическое изменение S-блоков (блоков нелинейных замен), с сохранением показателей их нелинейности.

Управляемыми с помощью битов подключа, как отмечено выше, сделаны и операции MixColumn и ShiftRow (см. [184]).

Шифр мини-Калина. Этот алгоритм шифрования также повторяет принципиальные решения, использованные при построении основной версии предложения [48], и практически является результатом масштабирования оригинальной разработки к размерам входного блока и ключа равным 16-ти битам [187]. Сам шифр Калина является Rijendael-подобным шифром. Его отличия от шифра Rijendael состоят, во-первых, в использовании разных S-

блоков, которые построены на основе отбора из множества случайных подстановок и поэтому не допускают простого алгебраического описания, как это удается сделать для S-блоков шифра Rijendael.

Во-вторых, в отличие от Rijendael-я в шифре Калина (его первой версии) цикловый подключ вводится не однообразным сложением состояний шифра с ключевыми состояниями по mod2, а используется попеременное (через цикл) сложение с подключами по модулю два (XORRoundKey) для нечетных циклов и сложение с подключами по модулю 2^{32} (Add32RoundKey) для четных циклов. Наконец, в шифре Калина использована отличная от Rijendael-я схема разворачивания ключей.

Все эти особенности учтены при построении малой версии шифра, т.е. в малой модели практически полностью повторяются решения по построению функций ShiftRows и MixColumns шифра Baby Rijendael, при этом, как и в шифре Baby Rijendael, слой нелинейного преобразования реализован с помощью четырех S-блоков 16-го порядка, а вместо функции Add32RoundKey использовано сложение с битами подключа по модулю 2^4.

Для получения цикловых подключей из исходного мастер-ключа используется процедура разворачивания ключей, повторяющая в своей уменьшенной версии оригинальную разработку [48].

Для 10-ти цикловой мини версии требуется 12 подключей, каждый длиной 4×4 бита (размер подключа совпадает с размером открытого/шифро-ванного текста.

Шифр мини Мухомор В большом шифре «Калина» [см. 48] используется 8 различных подстановок “байт-в-байт”, причем для байтов одной строки текущего состояния шифра используется одна и та же подстановка. В описании шифра готовые таблицы подстановок приведены в приложении и об их построении ничего не сказано. Для шифра «Мухомор» [49] указывается, что таблицы подстановок совпадают с первыми 4-мя S-блоками алгоритма «Калина». Поэтому при построении уменьшенных моделей этих шифров предложено использовать набор малых S-блоков (с

разными или одинаковыми) параметрами γ шифра baby-ADE**,** или малые S-блоки шифра Fox [36], обладающие, как утверждают авторы разработки, высокими дифференциальными и линейными характеристиками переходов. Отметим, что в исследованиях, представленных в этой работе, они не применялись**.**

Отметим теперь, что при построении уменьшенной версии шифра Мухомор пришлось столкнуться с ситуацией, когда шифр содержит преобразование, не допускающее прямого масштабирования. К такой операции относится *SL*-преобразование. В эту операцию оригинальной конструкции входят слой нелинейных преобразований, реализуемый с помощью 4-х байтовых S-блоков, и последующее МДР преобразование, осуществляющее матричное умножение байтовых выходов 4-х S-блоков (над полем $GF(2^8)$) на квадратную матрицу, размера 4×4 (по существу аналогичное преобразование выполняется в шифре Rijndael с помощью операции MixColumns, только там при умножении используется другой полином). При масштабировании этой операции к 16-ти битной модели она получается 4-х битной (в оригинале она 32-битная). В результате было принято решение заменить эту операцию подстановочной, в каком-то смысле эквивалентной по эффективности. В работе [190] была выполнена оценка дифференциальных свойств (закона распределения переходов таблицы XOR разностей) *SL*-преобразования 16-битной версии, повторяющей оригинальную. В нашем эксперименте для построения *SL*-преобразования были использованы четыре S-блока уменьшенной версии шифра Baby-Rijndael [188]. Результаты вычислительного эксперимента по выполнению операции MixColumns над четырьмя такими S-блоками (одинаковыми) привели к выводу, что для полубайтовых S-блоков выбрать соответствующий разумно обоснованный эквивалент не удается (по крайней мере, с теми наборами S-блоков, которые использовались в эксперименте).

Поэтому было принято решение в качестве *SL*-преобразований в малой версии шифра использовать просто S-блоки случайного типа.

Все остальные операции по масштабированию прототипа, в том числе и операция разворачивания мастер-ключа, затруднений не вызвали.

Шифр baby-Лабиринт. Как и все другие алгоритмы, шифр Лабиринт [50] построен по итеративной схеме, т.е. его основу составляет цикловое преобразование, которое повторяется заданное число раз. Каждый цикл состоит из двух абсолютно идентичных итераций, осуществляющих преобразование (зашифрование) двух полублоков, на которые разбивается блок данных на входе каждого цикла. Учитывая, однако, что для обновления обоих полублоков, составляющих один блок, в предлагаемом решении требуется, как минимум, две итерации, автор в своем описании алгоритма отделил понятие цикла от понятия итерации (у него цикл состоит из двух итераций).

Кроме повторяющегося циклового преобразования процедура зашифрования включает также начальное (IT) и конечное (FT) преобразования. Свойство инволютивности шифра достигается за счёт применения классической конструкции полублоковой цепи Фейстеля.

Все эти решения естественно сохранены в уменьшенной модели алгоритма Лабиринт [181]. Фейстель подобную структуру процедуры зашифрования с учетом уменьшенного размера входного блока данных иллюстрирует рис.4.1.

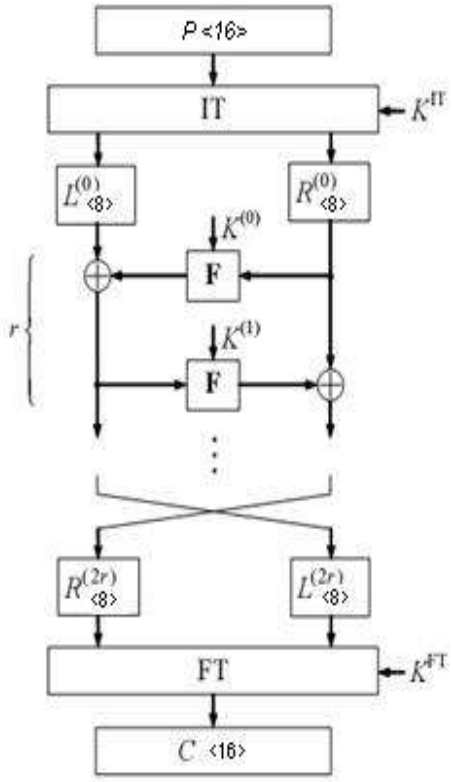


Рис. 4.1. Структура процедуры шифрования

Число циклов в уменьшенной версии может меняться от 1 до 16.

Поскольку шифр построен по достаточно оригинальной схеме, мы приведем более детальное его описание.

Процедура зашифрования EF состоит из трёх этапов:

Исходный блок данных P<16> (длиной 16 бит, что подчеркивается приставкой <16> к символу исходного блока данных P) обрабатывается начальным (IT) преобразованием на ключе K^{IT} (здесь и далее мы по возможности будем сохранять символику и обозначения, использованные автором в оригинальной разработке).

Результат 1-го этапа разбивается на два полублока длиной по 8 бит каждый: левый $L^0_{\langle 8\rangle}$ («старший») и правый $R^0_{\langle 8\rangle}$ («младший»). Полученная пара полублоков преобразуется на 8 итерациях (4 циклах) цепи Фейстеля. Все итерации полностью идентичны и построены на базе общего нелинейного преобразования − F-функции, управляемой ключами итерации $K^{(i)}$, $i = 1, 2, , , , , 2r$;

На третьем этапе два полублока $L_{\langle 8\rangle}^{2r}$ и $R_{\langle 8\rangle}^{0}$, полученные в результате 4-х циклового итеративного преобразования, меняются местами и обрабатываются конечным (FT) преобразованием на ключе K^{FT}. Полученный после FT преобразования двоичный вектор C<16> (длиной 16 бит) является зашифрованным блоком («криптограммой»).

На каждой итерации F-функция использует 8-ми битный подключ $K^{(i)}$. Длина ключей начального (K^{IT}) и конечного (K^{FT}) преобразований составляет по 16 бит каждый.

F-функция. В уменьшенной модели повторена конструкция F-функции большого шифра, только теперь операции выполняются не над байтами а, над двумя полубайтами, а сложение полубайтов (двух) циклового ключа с полублоками из полубайтов входного блока данных выполняется теперь уже по модулю 2^8. Кроме того, фиксированная перестановка P (как и в большом шифре для значения длительности блока данных 128 бит) в уменьшенной версии шифра не применяется. В итоге, F-функция, использующая в процессе преобразований подключ из 8 бит, может быть представлена схемой Рис.4.2, повторяющей структуру F-функции большого шифра.

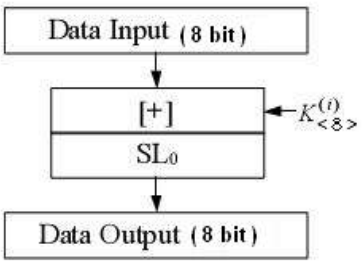


Рис. 4.2. 8-битная F-функция

Начальное и конечное преобразования. Начальное и конечное преобразования представляют по сути два дополнительных цикловых преобразования шифра.

Конструкция SL_0-преобразования будет рассмотрена далее отдельно.**Начальное и конечное преобразования**. Начальное и конечное преобразования представляют по сути два дополнительных цикловых преобразования шифра.

Начальное IT преобразование Начальное IT преобразование уменьшенной версии шифра «Лабиринт» (рис. 4.3), повторяя по структуре оригинальную разработку, включает сложение теперь уже по модулю 2^8 полублоков входного блока данных (по 8 бит) с 8-битными рабочими подключами (левый полублок ключа (8 бит) суммируется с левым полублоком входного блока

1. .....
2. ..... P

ис. 4.3. Начальное преобразование IT

данных, а правый полублок ключа (8 бит) соответственно с правым полублоком входного блока данных). На следующем шаге 16 результирующих бит разбиваются на блоки по 4-е бита, которые подаются на нелинейные преобразования S-блоки. Результат 16 бит снова разбивается на два полублока, над которыми выполняются циклические сдвиги: левый полублок сдвигается на 2 бита влево, а правый соответственно на два бита вправо, и в заключение над полученным 16-ти битным блоком выполняется операция инволютивного линейного смешивания IMix.

В уменьшенной версии шифра операция IMix, повторяя идею оригинального алгоритма, реализует сложение по модулю два 8-ми битных половинок входного блока данных, и после циклического сдвига результата

влево на 5 бит, его сложение по модулю два с полублоками входного блока данных поступающими на вход преобразования IMix.

Конечное FT преобразование (рис.4) выполняет те же операции что и начальное, но только в обратном порядке.

3..... Как и в большом шифре, после выполнения последней итерации цепи Фейстеля, левый и правый полублоки меняются местами, и только после этого результирующий блок поступает на вход FT-преобразования.

Преобразование SL. SL-преобразование определяет фиксированное биективное отображение слов. Это преобразование представляет собой объединение нелинейного преобразования байтов (с помощью байтовых S-блоков), с последующим линейным «смешиванием» результатов нелинейного преобразования.

В уменьшенной модели операции выполняются над полубайтами. Поэтому сначала над каждыми двумя полубайтами, объединенными в полублок, выполняется нелинейное преобразования с помощью уменьшенных (полубайтовых) S-блоков, над выходными полубайтами которых выполняется MBN-преобразование (линейное смешивание). Это преобразование осуществляет биективное линейное отображение слов, составляющих полублок (для уменьшенной модели полублок имеет размерность байта).

В уменьшенной версии шифра полностью сохранена идея построения MBN-преобразования. Оно реализовано в виде умножения квадратной матрицы размерностью 2×2 полубайта, образованной циклическим MBN-кодом, справа на вектор-столбец длиной 1 байт (два полубайта, соответствующих слову-аргументу). Элементы матрицы и элементы векторов аргумента − результата, интерпретируются как элементы поля $GF(2^4)$ (полубайты), образованного выбранным неприводимым (над полем GF(2)) полиномом 4-й степени $f_{MBN}(x) = x^4 + x^3 + 1$.

В данной разработке мини-версии БСШ «Лабиринт» использован следующий полином второй степени, порождающий MBN-код:

$$g(x) = 11x + 01.$$

Как отмечает сам автор разработки в [50], S-блок шифра «Лабиринт» выбран из множества так называемых предельно-нелинейных биективных преобразований, в основе которых лежит конструкция Ниберг-Динга, т.е. преобразование, аффинно-эквивалентное функции вычисления обратного элемента в поле. В малой версии шифра Лабиринт эта конструкция повторена для полубайтовых подстановок. Детальное описание этой конструкции можно найти в [189]. Можно только отметить, что практически свойства S-блоков шифра Лабиринт оказывается близкими к свойствам S-блоков шифра Rijndael.

В малой версии шифра предусмотрена и процедура разворачивания мастер-ключа. Она отмасштабирована под длину исходного (пользовательского) ключа 16 бит и полностью в уменьшенной версии повторяет авторскую разработку.

Шифр baby-Rijndael. Еще одну **у**меньшенную до нужных размеров модель шифра AES (baby-Rijndael) мы нашли в Интернете [191].

Если в ранее рассмотренном предложении [188] предлагалось в качестве элементов нелинейной замены использовать первую строку подстановки

1-го S-блока шифра DES, то достоинством этой версии является то, что, как показал анализ, автор статьи, повторил при построении S-блоков идеи, использованные Дайменом и Риджменом (это S-блок BabyR в таблице 4.1).

С другой стороны линейное преобразование в этом шифре, хотя и использует умножение на матрицу, оно отличается по реализации от линейного преобразования шифра Rijndasel.

Мы здесь приведем описание baby-Rijndael., так как оно совсем небольшое.

Шифрование начинается с того, что входной 16-битный блок данных в полубайтовом представлении $\gamma_0\gamma_1\gamma_2\gamma_3$ преобразуется в матрицу состояния

$$\begin{bmatrix} \gamma_0 & \gamma_2 \\ \gamma_1 & \gamma_3 \end{bmatrix}.$$

Общая структура процедуры зашифрования представляется в виде:

$$E(a) = r_4 \circ r_3 \circ r_2 \circ r_1(a \oplus k_0).$$

В этом выражении a обозначает состояние k_0, k_1, k_2, k_3, k_4 − цикловые подключи и

$$r_i(a) = \text{[illegible]}(S(a)) \oplus k_i,$$

за исключением того, что в последнем цикле умножение на матрицу t отсутствует. В конце зашифрования состояние преобразуется в 16-битный блок в том же самом порядке, как это было сделано в вначале при загрузке.

Описание индивидуальных компонент шифра:

• S операция является табличным преобразованием замены каждой шестнадцатеричной цифры состояния другой шестнадцатеричной цифрой исходного множества:

$$\begin{bmatrix} \gamma_0 & \gamma_2 \\ \gamma_1 & \gamma_3 \end{bmatrix} \to \begin{bmatrix} s(\gamma_0) & s(\gamma_2) \\ s(\gamma_1) & s(\gamma_3) \end{bmatrix},$$

где S функция задается таблицей (строкой в таблице 1)

• Операция [illegible] − это обмен входов во второй строке состояния:

$$\begin{bmatrix} \gamma_0 & \gamma_2 \\ \gamma_1 & \gamma_3 \end{bmatrix} \overset{[illegible]}{\to} \begin{bmatrix} \gamma_0 & \gamma_2 \\ \gamma_3 & \gamma_1 \end{bmatrix}.$$

• Матрица t линейного преобразования является 8×8 битовой матрицей вида:

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Для этого преобразования состояние рассматривается как 8×8 битовая матрица. Состояние умножается слева, используя матричное умножение на матрицу t по модулю 2: $a \to t \cdot a$. Отмечается, что верхняя левая 4×4 подматрица матрицы t равна нижней правой подматрице. И подобно этому верхняя правая и нижняя левая подматрицы равны.

• В начале шифра и в конце каждого цикла состояние побитно складывается по модулю два с цикловыми подключами. Они задаются 2×2 массивами шестнадцатибитных цифр подобных состоянию. Колонки цикловых подключей определяются рекурсивно по следующему правилу:

$$w_0 = \begin{pmatrix} k_0 \\ k_1 \end{pmatrix} \quad w_1 = \begin{pmatrix} k_2 \\ k_3 \end{pmatrix}$$

$$w_{2i} = w_{2i-2} \oplus S(\text{reverse}(w_{2i-1})) \oplus r_i$$

$$w_{2i+1} = w_{2i-1} \oplus w_{2i}$$

для $i = 1, 2, 3, 4$.

Константы определяются правилом $r_i = \begin{pmatrix} 2^{i-1} \\ 0 \end{pmatrix}$, а обратные (reverse) функции меняют местами два входа в колонке, S функция та же самая, что и ранее. Все сложения выполняются побитно по модулю два. Наконец, для $i = 0, 1, 2, 3, 4$ цикловые подключи k_i − матрица, колонками которой являются w_{2i} и w_{2i+1}. Можно заключить, что еще одной особенностью этой уменьшенной модели является процедура линейного преобразования отличающаяся от примененной в шифре Rijndael – в уменьшенной модели шифра baby-Rijndael умножение на матрицу осуществляется сразу для

четырех (всех) полубайтовых выходов S-блоков, причем сама матрица имеет битовые значения, и операция умножения выполняется по модулю 2 [191]. Следует заметить, что при выполнении экспериментов для этой модели будут использованы различные варианты линейного преобразования, в том числе и более точно повторяющие идею построения линейного преобразования шифра Rijndael.

Можно в заключение напомнить работу [189], в которой рассмотрены конструкции и свойства полубайтовых S-блоков шифров mini-AES, baby-Rijndael, baby-ADE, mini-FOX, baby-Лабиринт, мини-Калина и мини-Мухомор, повторяющие принципы, использованные в оригинальных разработках, и сделан вывод, что рассмотренные уменьшенные модели S-блоков разных шифров обладают практически одинаковыми (близкими) криптографическими свойствами.

Примеры таких S-блоков, использованных при построении уменьшенных моделей рассматриваемых шифров, иллюстрирует табл. 4.1, заимствованная из [189]).

Таблица 4.1

S-блоки для уменьшенных версий шифров.

Шифр	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
MiniA	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
BabyR	10	4	3	11	8	14	2	12	5	7	6	15	0	1	9	13
ADE6	10	2	0	6	15	1	12	4	14	11	7	13	9	5	3	8
ADE7	10	3	8	2	5	6	0	9	11	4	12	14	15	7	13	1
ADE3	10	11	2	14	0	13	6	7	15	5	1	9	12	8	4	3
Лабир.1	9	5	14	7	3	12	13	4	2	1	8	15	10	0	6	11
Лабир.2	9	4	12	5	7	1	11	6	10	8	3	0	14	15	2	13
FOX1	2	5	1	9	14	10	12	8	6	4	7	15	13	11	0	3

4.3 Комбинаторные показатели случайности малых моделей шифров.

Продемонстрируем сначала свойства малых шифров как случайных подстановок по комбинаторным показателям: числу инверсий, возрастаний и циклов. Напомним кратко методики выполнения вычислительных экспериментов в этом случае.

4.3.1. Методики выполнения исследований комбинаторных показателей малых моделей шифров.

Изложим сначала последовательно сущность выполняемых операций при анализе каждого из комбинаторных показателей.

Методика построения закона распределения числа циклов. В этом случае для каждого ключа зашифрования из полного множества ключей производится подсчет числа циклов подстановки, соответствующей каждому ключу. Поиск циклов и подсчет их числа производиться следующим образом. Создаётся массив всех возможных вариантов зашифрованных текстов. Затем начиная с нулевого значения элемента верхней строки подстановки (нулевого для зашифрования значения входного блока данных), выполняется зашифрование с последующим выполнением функции зашифрования к блокам данных, получающихся в результате зашифрования на текущем шаге. Значения результатов зашифрования фиксируются в исходном (запомненном) массиве путем исключения из него (или маркирования) появившихся в результате зашифрований значений. Эта операция выполняется до тех пор, пока в результате зашифрования не появится блок данных с нулевым значением (начальный блок данных серии последовательных зашифрований). После этого выбирается наименьший по значению из оставшихся элементов массива (не попавший в предыдущий цикл), и строится новый цикл, начинающийся от нового значения, с запоминанием (″вычеркиванием″ или маркированием) пройденных (состоявшихся) зашифрованных значений. И эта процедура повторяется до тех пор, пока не будут пройдены (″вычеркнуты″) все элементы исходного массива. Параллельно с поиском циклов выполняется подсчет их числа с помощью соответствующего накапливающего счетчика. Для последующего

построения закона распределения вероятностей для числа циклов используются дополнительный набор счетчиков, в которых подсчитывается число счетчиков, имеющих одинаковое заполнение, т.е. фиксирующих число подстановок (ключей зашифрования) с одним и тем же числом циклов.

Методика построения закона распределения числа иверсий. В этом случае для каждого ключа зашифрования из полного множества всех ключей путем последовательных зашифрований (на одном и том же ключе) создаётся массив всех возможных вариантов зашифрованных блоков данных, начиная с зашифрования нулевого значения блока данных, и так последовательно до последнего значения входного блока данных (равного $2^{16}-1$). Тем самым в массиве данных запоминается вторая строка нормализованного представления подстановки (шифрующего преобразования). Затем на основе последовательного просмотра и сравнения значений элементов массива с текущим, выбранным для анализа, выполняется подсчет числа инверсий, соответствующего рассматриваемому элементу массива (числа превышений значения текущего рассматриваемого элемента множества значений просмотренных элементов, стоящих в массиве правее рассматриваемого). Результат подсчета числа инверсий для каждого из последовательно выбранных элементов массива нижней строки подстановки отсылается в накапливающий счетчик, фиксирующий общее число инверсий соответствующее текущей подстановке. Естественно, что потребуется использовать 2^{16} накапливающих счетчиков, чтобы затем построить, по данным счетчиков закон распределения вероятностей для числа инверсий.

Методика построения закона распределения числа возрастаний. И в этом случае создаётся массив всех возможных вариантов зашифрованных текстов (строится вторая строка нормализованного представления подстановки). Для подсчета числа возрастаний для каждого элемента массива производится подсчет числа элементов массива, имеющих следующий элемент массива больший по значению текущего. Число превышений

(возрастаний), полученных для каждой подстановки фиксируется соответствующим накапливающим счетчиком.

4.3.2. Исследование циклических свойств малых моделей шифров.

Напомним работу [176], посвящённую изучению циклических свойств блочных симметричных шифров и последующие работы [181, 186]. В указанных работах были определены (вычислены) законы распределения инверсий, возрастаний и циклов для малых версий шифров ГОСТ, DES, Rijndael и сделан вывод о том, что полученные в результате обработки экспериментальных данных результаты с достаточно высокой точностью повторяют известные из комбинаторики асимптотические нормальные законы распределения вероятностей свойственные случайным подстановкам. В частности, установлено совпадение числовых характеристик (средних значений и дисперсий инверсий, возрастаний и циклов) с соответствующими характеристиками, полученными из расчётных соотношений для асимптотических законов (полагается, что результаты экспериментов с ещё более высокой точностью будут выполняться и для больших версий шифров). Вместе с тем, в этих работах не удалось подтвердить совпадения соответствующих эмпирических законов распределения вероятностей с асимптотическими законами.

Сегодня стало ясно, что сравнение экспериментально полученных законов необходимо проводить не с асимптотическими распределениями, а с законами распределения инверсий, возрастаний и циклов случайных подстановок соответствующей степени. Некоторые результаты исследований в этом направлении вместе с уже известными представлены ниже.

Сначала в табл. 4.2 приводятся результаты оценки циклических свойств ряда уменьшенных моделей шифров, представленных на Украинский конкурс по выбору национального стандарта шифрования, вместе с показателями шифра Rijndael − победителя конкурса AES, заимствованные из работы [176].

Далее в таблице 4.3 представляются расчёты средних значений и дисперсий распределений числа циклов для всех рассмотренных шифров.

Таблица 4.2

Сравнение циклических свойств уменьшенных моделей БСШ

Количество циклов	Количество подстановок (ключей зашифрования)						
	Baby-Мухомор (4 цикла)	Baby-Rijndael (10 циклов)	Baby ADE	mini-Kalina (4 цикла)	mini-Kalina (10 циклов)	Baby «Лабиринт» (2 цикла)	Baby «Лабиринт» (8 цикл.)
2	28	18	32	2	27	16	25
4	480	499	521	549	507	517	559
6	3169	3255	3322	3167	3234	3246	3129
8	9353	9436	9415	9528	9482	9306	9447
10	15391	15429	15366	15256	15317	15397	15320
12	16232	15963	16124	15988	16054	16008	16085
14	11432	11580	11400	11760	11524	11566	11642
16	6068	5956	5952	5935	6052	6120	6066
18	2391	2411	2397	2384	2407	2401	2277
20	716	774	778	733	706	708	775
22	201	174	188	166	186	198	173
24	32	35	34	39	34	44	33
26	11	6	5	7	5	6	5
28	2	0	1	0	1	0	0
30	0	0	1	0	0	0	0

Таблица 4.3

Характеристики законов распределения числа циклов для шифров, рассмотренных в таблице 1

Числовая характеристика	Baby-Rijndael (10 циклов)	Baby ADE	mini-Kalina (10 циклов)	Baby-Мухомор (4	Baby-Лабирин (8 цикл.)

				цикла)	
Среднее значение	11,66	11,65	11,66	11,67	11,66
Дисперсия	10,0489	10,1124	10,0489	9,9856	9,9856

Самый главный результат нашей работы – это выполнение проверки соответствия законов распределения чисел циклов для малых версий шифров, соответствующему экспериментально полученному закону распределения случайной подстановки степени 2^{16}.

В процессе вычислительных экспериментов для множества из 10000 случайно сгенерированных подстановок степени 2^{16} был построен закон распределения подстановок для числа циклов. Результаты этого эксперимента иллюстрирует таблица 4.4. В правой (в предпоследней) колонке этой таблицы для сравнения мы привели значения вероятностей для числа подстановок (ключей зашифрования) с фиксированными значениями числа циклов мини версии шифра Baby-Rijndael. В самой правой колонке представлены расчётные значения для разностей интегральных функций распределения числа циклов случайной подстановки степени 2^{16} и числа циклов шифра Baby-Rijndael, которые используются в критерии Колмогорова [176].

Таблица 4.4

Закон распределения числа циклов для чётных подстановок степени 2^{16}

Длина цикла k	Число подстановок ξ_n	Расчётное значение вероятности P_k	Baby-Rijndael (10 циклов)	$\lvert F_R(x_k) - F_{СП}(x_k) \rvert$
2	1	0,0002	0,00027	0,00007
4	42	0,0086	0,00761	0,00092
6	248	0,0496	0,04967	0,00085
8	694	0,1388	0,14398	0,00433

10	1222	0,2444	0,23543	0,00464
12	1192	0,2384	0,24358	0,00054
14	893	0,1786	0,1767	0,00136
16	460	0,092	0,09088	0,00248
18	189	0,0378	0,03679	0,00349
20	54	0,0108	0,01181	0,00248
22	13	0,0026	0,00266	0,00242
24	2	0,0004	0,00053	0,00229
26	1	0,0002	0,00009	0,0027

Из таблицы следует, что наибольшее значение разности $D_n = |F_R(x_k) - F_{CП}(x_k)|$ равно 0,00464. Для $\alpha = 0{,}05$ из таблицы распределения Колмогорова-Смирнова [171] находим $Q(\lambda_0) = 1 - \alpha = 1 - 0{,}5 = 0{,}95 \rightarrow \lambda_0 = 1{,}36$. Тогда для $n = 26$ выходит $\frac{\lambda_0}{\sqrt{n}} = \frac{1{,}36}{256} = 0{,}00531$. Следовательно, $D_n < \frac{\lambda_0}{\sqrt{n}}$. Гипотеза о том, что интересующая нас случайная величина распределена по закону, свойственному случайной подстановки, подтверждается.

4.3.3. Исследований законов распределения инверсий и возрастаний уменьшенной модели шифра «Калина»

В этом разделе мы рассмотрим только шифр mini-Калина. Мы здесь пошли по пути графической интерпретации результатов и не стали выполнять проверку соответствия эмпирических законов теоретическим, так как их близость не вызвала сомнений.

Результаты построения законов распределения возрастаний и инверсий, выполненные в соответствии с изложенными выше методиками, иллюстрируют рис.4.4 и рис.4.5.

Рис.4.4 Закон распределения числа инверсий для шифра мини-Калина

Подобные законы распределения вероятностей могут быть построены и для других шифров. Например, на Рис. 4.6 приведен закон распределения инверсий для уменьшенной модели шифра baby-ГОСТ 28147-89 [176].

Представленные результаты свидетельствуют, что законы распределения числа циклов, возрастаний и инверсий мини-версий шифров являются весьма

Рис.4.5. Закон распределения числа возрастаний для шифра мини-Калина

близкими к нормальным, т.е. повторяют свойства, характерные для случайных подстановок [165]. Для определения числовых значений этих распределений можно пользоваться формулами соответствующих асимптотических законов распределения случайных подстановок.

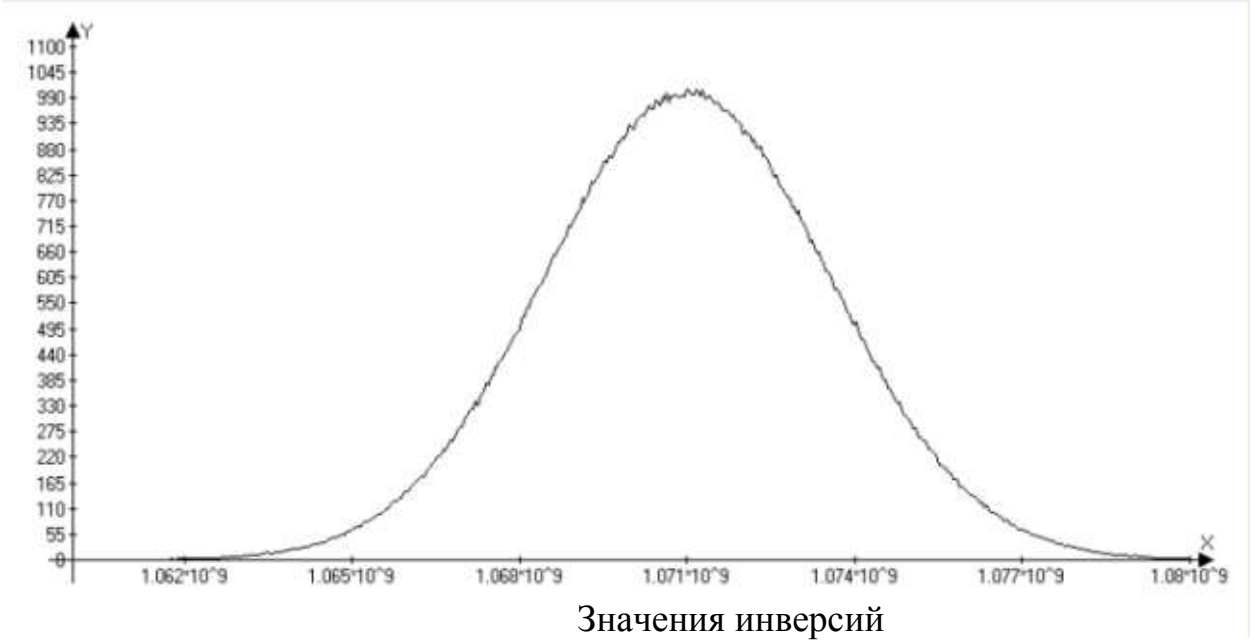


Рис. 4.6. Закон распределения числа инверсий для шифра baby-ГОСТ.

Далее будут рассмотрены дифференциальные и линейные показатели малых моделей шифров, представленных на украинском конкурсе вместе с малыми моделями других шифров. Но сначала напомним понятия и определения дифференциального и линейного криптоанализа и определим нашу точку зрения к формированию показателей доказуемой стойкости.

4.4. Понятийный аппарат линейного и дифференциального криптоанализа. Новые показатели доказуемой стойкости.

Уместно будет напомнить основной понятийный аппарат линейного и дифференциального криптоанализа, которым мы воспользуемся в дальнейшем. Следуя работе [126], приведём ряд определений.

Определение 1. (Дифференциальная и Линейная вероятности): *Дифференциальная вероятность* DP^f *и линейная вероятность* LP^f *соответственно для ключезависимой функции* f *с* n*-битным входом* x *и* n*-битным выходом* y $(x, y \in GF(2^n))$ *есть*

$$DP^f(\Delta x \to \Delta y) = \frac{\#\{x \in GF(2)^n \mid f(x) \oplus f(x \oplus \Delta x) = \Delta y}{2^n}; \quad (4.1)$$

$$LP^f(Гx \to Гy) = \left(\frac{\#\{x \in GF(2)^n \mid x \cdot Гx) = f(x) \cdot Гy}{2^{n-1}} - 1 \right)^2, \quad (4,2)$$

где Δx и Δy являются входной и выходной разностями, а $Гx$ і $Гy$ – это входная и выходная маски; $x \cdot Гx$ обозначает результат скалярного произведения x на $Гx$, $f(x) \cdot Гy$ – результат скалярного произведения $f(x)$ на $Гy$.

Определение 2. ($DP^f_{\max}$ и $LP^f_{\max}$): *Максимальное значение дифференциальной и линейной вероятностей для ключезависимой функции f определяются соответственно как*

$$DP^f_{\max} = \max_{\Delta x \neq 0, \Delta y} DP^f(\Delta x \to \Delta y), \quad (4.3)$$

$$LP^f_{\max} = \max_{Гx,\ Гx \neq 0,} LP^{f[k]}(Гx \to Гy). \quad (4.4)$$

В общем случае, ключезависимая функция f является сильной, если значения $DP^f_{\max}$ и $DL^f_{\max}$ функции f являются достаточно малыми [126].

Нас в дальнейшем и будут интересовать значения $DP^f_{\max}$ и $DL^f_{\max}$ для случаев, когда в качестве функции f выступают цикловые преобразования и последовательности цикловых преобразований итеративных шифров (ключезависимые функции), а также подстановочные преобразования (неключезависимые функции).

Пусть π – подстановочная таблица с n-битными входами и n-битными выходами. В [127] доказана лемма 2 (в обозначениях работы [126] S заменено на π):

Лемма 2. *Для любого преобразования π: $Z_2^n \to Z_2^n$:*

$$\sum_{\Delta y \in Y} DP^{\pi}(\Delta x \to \Delta y) = 1, \quad (4.5)$$

$$\sum_{\Gamma x \in X} LP^{\pi}(\Gamma x \to \Gamma y) = 1. \tag{4.6}$$

И, более того, если π *– подстановка, то*

$$\sum_{\Delta x \in X} DP^{\pi}(\Delta x \to \Delta y) = 1, \tag{4.7}$$

$$\sum_{\Gamma y \in Y} LP^{\pi}(\Gamma x \to \Gamma y) = 1. \tag{4.8}$$

Эти результаты представляются достаточно очевидными, исходя из определений (4.1) и (4.2) примененных к подстановкам (неключезависимым преобразованиям). Они являются отражением известных фактов, заключающихся в том, что суммы ячеек таблицы XOR разностей и суммы квадратов ячеек таблиц линейных аппроксимаций подстановок по строкам и по столбцам равны 2^n и 2^{2n} соответственно, где n – битовый размер входа и выхода подстановки степени 2^n. Далее нас будут интересовать случайные свойства шифров (шифрующих преобразований). Речь будет идти об обосновании новой идеологии.

Напомним также выражения для средних вероятностей *ADP*, *ALHP*, *MADP* і *MALHP* ключезависимой функции $f = f[k](x)$ с n-битным входом x и n-битным выходом, параметризованной ключом k, которые используются во многих публикациях по обоснованию показателей стойкости блочных шифров.

Определение 3. *Среднее значение дифференциальной вероятности* (*ADP*) *функции* $f[k](x)$ *есть*

$$ADP^{f} = \underset{k}{ave}\, DP^{f[k]}(\Delta x \to \Delta y).$$

Определение 4. *Среднее значение вероятности линейного корпуса* (*ALHP*) *функции* $f = f[k](x)$ *есть*

$$ALHP^{f} = \underset{k}{ave}\, LP^{f[k]}(\Gamma x \to \Gamma y).$$

Определение 5: *Максимум среднего значения дифференциальной вероятности* (*MADP*) *и максимум среднего значения вероятности линейного корпуса* (*MALHP*) *функции* $f = f[k](x)$ *есть*

$$MADP^f = \max_{\Delta x \neq 0, \Delta y} ADP^f(\Delta x \to \Delta y). \quad (4.9)$$

$$MALHP^f = \max_{\Gamma x,\, \Gamma y \neq 0} ALHP^f(\Gamma x \to \Gamma y). \quad (4.10)$$

Заметим теперь, что приведенные здесь определения *MADP* и *MALHP*, повсеместно используемые в публикациях, на наш взгляд не являются адекватными задаче оценке потенциальных характеристик стойкости шифра к атакам дифференциального и линейного криптоанализа (они характеризуют лишь максимумы средних значений таких вероятностей, вычисленных для некоторого фиксированного перехода $\Delta x \to \Delta y$ или некоторого фиксированного сочетания масок $\Gamma x \to \Gamma y$).

Изложим здесь сущность нашего предложения (метода), которое основывается на использовании для оценки стойкости БСШ к атакам дифференциального и линейного криптоанализа не $MADP^f$ (максимума средней дифференциальной вероятности) для некоторого фиксированного перехода входной разности Δx в выходную разность Δy, а среднее (по ключам) значение максимумов дифференциальных вероятностей (*AMDP*) ключезависимой функции $f[k](x)$ (для линейного криптоанализа соответственно предлагается пользоваться не *MALHP* а средним значением максимумов линейных вероятностей (*AMLHP*)) [190].

Дело в том, что на наш взгляд при вычислении $MADP^f$ ($MALHP^f$) теряется смысл, который вкладывается в понятие максимальной дифференциальной (линейной) вероятности. По логике стойкость шифра определяется наименьшей сложностью атаки, которая соответствует переходу с наибольшей вероятностью. В случае же $MADP^f$ при усреднении учитываются значения фиксированного перехода на всех ключах, среди которых присутствуют все возможные значения переходов. Нам кажется это

не логичным. Предлагается оценивать стойкость шифра, как это и требуется по логике, максимальными значениями вероятностей переходов (дифференциальных и линейных). Здесь можно выполнять и усреднение этих максимальных вероятностей по множеству ключей, однако, как показывают результаты экспериментов значения максимальных вероятностей, оцениваемых по нашему методу, практически от значений ключей не зависят. Кроме того, при нашем подходе получаемые результаты полностью согласуются и с дифференциальными (если идти дальше, то и линейными) свойствами случайных подстановок.

Наконец, при вычислении $MADP^f$ сначала определяется целиком дифференциальная таблица для каждого ключа зашифрования (нужно запоминать всё множества таблиц), потом выполняется усреднение значений однотипных переходов по множеству ключей с последующим выбором из множества усредненных значений максимального (по вероятности). В тоже время при вычислении среднего (при усреднении по множеству ключей) значения максимумов дифференциальных вероятностей (*AMDP*) ключезависимой функции $f[k](x)$, необходимо запоминать только значения максимумов для каждой из дифференциальных таблиц. Последующее усреднение этих максимумов в вычислительном отношении оказывается намного более удобным (нет надобности запоминать целиком все таблицы, а достаточно только определять и запоминать их максимальные значения).

Более адекватное, на наш взгляд, определение максимума дифференциальной вероятности (показателя *AMDP*) можно представить в таком виде.

Определение 6 (*AMDP*). *Среднее* (*по множеству из* 2^h *ключей*) *значение максимальных дифференциальных вероятностей ключезависимой функции* $f[k](x)$ *есть*

$$ADMP^f = \underset{k}{ave}\, DP_{\max}^{f[k]} = \frac{1}{2^h}\sum_{k=1}^{2^h} DP_{\max}^{f[k]}(\Delta x \to \Delta y), \qquad (4.11)$$

где 2^h *– мощность использованного множества ключей зашифрования*.

Очевидно неравенство:

$$MADP^f \le AMDP^f,$$

причём отличие результатов, полученных для рассматриваемых вариантов усреднения, оказывается близким к 25% (на уровне AES суперблока), зато, как уже отмечалось выше, в последнем случае обеспечиваются значительные вычислительные преимущества и, самое главное, здесь получается более адекватная оценка максимума полного дифференциала шифра.

Определение 7 (*AMPLH*). *Среднее (по ключам) значение максимальных вероятностей линейных корпусов функции* $f[k](x)$ *есть*

$$AMLHP^f = \underset{k}{ave}\, LP_{\max}^f(Гx \to Гy) = \frac{1}{2^h}\sum_{k=1}^{2^h} LP_{\max}^{f[k]}. \qquad (4.12)$$

В обоих случаях 2^h – мощность множества ключей зашифрования, которые использованы при вычислениях.

Метод предусматривает нахождение максимумов таблиц дифференциальных разностей (линейных аппроксимаций) при использовании каждого ключа. Далее необходимо запомнить максимальное значение каждой таблицы дифференциальных разностей (линейных аппроксимаций), после чего найти среднее значение максимумов этих таблиц по множеству ключей.

Очевидно неравенство: *MALHP* < *AMLHP*.

Кроме большей адекватности оценок, которые формируются (значения оценок совпадают с соответствующими дифференциальными и линейными показателями случайных подстановок и характеризуют максимально достижимые значения дифференциальных и линейных вероятностей), в последнем случае обеспечиваются и значительные вычислительные преимущества (нет необходимости запоминать целиком все таблицы, а достаточно только определять и запоминать их максимальные значения). При этом обеспечивается уменьшение объёма необходимой для вычислений

памяти в 2^{2n} раз, где n – размер битового входа в шифр. Именно с использованием этих двух показателей выполнено обоснование новой идеологии оценки доказуемой стойкости БСШ в наших работах [1,41,42, 56,57,81 и др.].

4.5. Шифрующие преобразования как случайные подстановки.

Самый важный вывод работ [43] и [44] состоит в том, что приведенные в разделе 2 законы распределения переходов XOR таблиц и смещений таблиц случайных подстановок выполняются для шифрующих преобразований всех современных блочных симметричных шифров рассматриваемых как подстановочные преобразования.

Здесь следует обратить внимание на то, что в отличие от случайных подстановок шифры, реализуя ограниченное число 2^n ключезависимых подстановок в рассмотренном отношении, ведут себя более предсказуемо, чем это ожидалось. Они практически независимо от используемого ключа шифрования через определённое число циклов зашифрования становятся совершенными случайными подстановками, т.е. принадлежат ограниченному (но всё равно очень большому множеству подстановок), которые рассматривались в предыдущем подразделе.

Само по себе отдельное шифрующее преобразование (отдельный цикл) не является случайной подстановкой, так как для него не выполняются законы распределения вероятностей (3.2) и (3.8). Оно не укладывается в рамки случайных подстановок и по инверсиям и по возрастаниям и по циклам (хотя бы потому, что имеются множества входов в преобразование, которые влияют не на все значения выходов). Однако при реализации механизмов перемешивания (линейных преобразований), используемых в каждом цикле, последовательность шифрующих преобразований приобретает свойства случайной подстановки (к чему как раз и стремятся все разработчики шифров). Этот, казалось бы, тривиальный вывод остался не

замеченным разработчиками шифров и криптоаналитиками при формировании оценок показателей стойкости шифров к атакам дифференциального и линейного криптоанализа (они не могли правильно интерпретировать результаты, так как были связаны полномасштабными версиями шифров, не поддающимися вычислительным экспериментам). Как уже отмечалось выше, во всех известных работах показатели многоцикловых преобразований (стойкость к атакам дифференциального и линейного криптоанализа) непосредственно связывались и связываются с соответствующими показателями S-блоковых конструкций, используемых в качестве нелинейных преобразований в каждой цикловой функции.

Наша позиция состоит в том, что итоговые (асимптотические) показатели стойкости (максимум полного дифференциала таблицы XOR разностей последовательности шифрующих преобразований, также как и максимумы линейных аппроксимационных таблиц этих же преобразований, зависят только от числа циклов шифрующего преобразования (шифра) и размера его битового входа.

Этот вывод зафиксирован в виде утверждения [42].

Утверждение 3. *Для каждого блочного симметричного шифра* (*из числа известных итеративных БСШ*) *существует вполне определенное число циклов, после которого шифр приобретает свойства случайной подстановки. Дальнейшее наращивание числа циклов не влияет на итоговые дифференциальные и линейные свойства шифра. Это значение является одним и тем же для всех шифрующих преобразований с одинаковым битовым размером входа.*

Заметим, что приведенное утверждение в первой части представляется в известном смысле достаточно очевидным (в том смысле, что каждый реальный шифр строится так, чтобы набор его цикловых преобразований в той или иной мере обладал свойствами случайной подстановки), при нашем подходе это свойство определяется как промежуточный результат,

переходящий в асимптотическое значение одинаковое для всех шифров (с одинаковым битовым размером входа), и к тому же поддающееся расчету.

Дифференциальные и линейные свойства по принятой системе взглядов оцениваются значениями $DP_{\max}^{f}$ и $DL_{\max}^{f}$ или соответствующими максимумами средних дифференциальных и линейных вероятностей *MADP* и *MALP* (или *MALHP*) [136]. Сами используемые значения *MADP* и *MALP* являются заметно отличающимися от действительных значений максимальных вероятностей. Они характеризуют не максимальные значения дифференциальных и линейных вероятностей, а максимумы средних значений, что также представляется не совсем правильным. Мы в монографии будем пользоваться вместо *MADP* и *MALP* введенными выше показателями – *AMDP* и *AMLP* (*AMLHP*), которые, как мы покажем, являются более адекватными решаемой задаче.

4.6. Приближение Истинно Случайным Шифром

Ниже приводится краткий обзор работ, по своим идеям приближающихся к концепции, излагаемой в этой монографии.

Вспомним введенную ранее во втором разделе в рассмотрение модель случайного шифра: для данного блочного размера N, случайный шифр (или идеальный шифр) является параметризованным с помощью ключа семейством всех взаимно однозначных отображений из $\{0, 1\}^N \to \{0, 1\}^N$ таким, что каждое отображение реализуется точно одним ключом [192]. В [63] отмечается, что для обычных размеров блоков, случайный шифр не может быть практически реализован, так как это требует ключ астрономической длины (примерно $N \times 2^N$ бит). И в большинстве публикаций случайный шифр рассматривается с теоретических позиций и, как правило, считается идеальной моделью блочного шифра [193].

Уже можно назвать не одну работу, в которой обсуждалась связь между SPN шифрами и истинно случайным шифром. Так, Чен и Таварес в [193] дали статистическое доказательство того, что распределение входов в

таблицу XOR SPN с фиксированным ключом с увеличением числа циклов приближается к соответствующему распределению для случайного шифра.

В ещё одной работе Келихера, Мейера и Тавареса [117] показывается, что если подстановочные компоненты (S-блоки) SPN шифра выбраны случайно, то ожидаемое значение любого *ELP* входа сходится к соответствующему значению истинно случайного шифра как только число циклов зашифрования увеличивается. Это, утверждают авторы, дает количественную поддержку утверждению, что SPN структуры являются практическими приближениями случайных шифров. По результатам экспериментов они выдвинули гипотезу, что для SPN шифров с широким классом линейных преобразований для произвольных масок входа и выхода $a,b \in \{0,1\}^N \setminus \mathbf{0}$ имеет место предел $\lim_{T\to\infty} E_{SPN}[ELP(a,b)] = \frac{1}{2^N - 1}$.

Естественным является представление идеального шифра для любого ключа зашифрования в виде случайной Булевой функции. В [178] для закона распределения переходов случайной Булевой функции, например, доказывается теорема, определяющая вероятность дифференциала для такой функции в виде:

Теорема 1. Для случайной с n битным входом и m-битным выходом векторной булевой функции число $N(a,b)$ пар входов с входной разностью a, которые имеют выходную разность b, является случайным значением, имеющим биномиальное распределение:

$$\Pr(N(a,b) = 2i) = (2^{-m})^i (1 - 2^{-m})^{2^{n-1} - i} \cdot \binom{2^{n-1}}{i}. \qquad (4.13)$$

Приводится простое доказательство, которое здесь повторяется.

Доказательство: Случайная Булева функция отображает 2^n различных входных значений v в независимые выходные значения $\alpha(v)$ и, следовательно, это отображение разности пар $\{v,u\}$ в независимые выходные разности. Для данного значения (a,b) пара с разностью a есть случайный

эксперимент, для которого успешным исходом считается выходная разность b. Число успехов имеет биномиальное распределение (4.13).

Для $n \geq 5$ и $n - m$ малого это распределение достаточно точно аппроксимируется Пуассоновским распределением [178].

$$\Pr(N(a,b) = 2i) \approx e^{-2^{n-m-1}} \cdot \frac{2^{(n-m-1)i}}{i!} = Poisson(i; 2^{n-m-1}).$$

Наконец, для случайного векторного Булева преобразования с $n = m$ имеем:

$$\Pr(N(a,b) = 2i) \approx Poisson\left(i; \frac{1}{2}\right) = \frac{e^{-\frac{1}{2}}}{i!2^i}.$$

Из приведенных в третьем разделе результатов следует, это распределение Пуассона повторяет распределение (3.8), полученное для случайных подстановок.

Можно отметить и ещё одно направление исследований. Хейс и Таварес рассмотрели *лавинные* свойства SPN-ов [194], рассматривая числа битовых изменений в шифртексте при изменении одного бита текста после каждого цикла. На основе этой модели, ожидаемое число изменившихся бит в шифртекте после R циклов, как, оказалось, сходится к значению $N/2$ − значению для истинно случайного шифра – как только R увеличивается. Сходимость довольно быстрая: на 64-битных SPN-ах с 8×8 S-блоками и перестановками Кама и Дэвида [195], значение из модели почти точно равно N/2 для $R \geq 5$. Юсеф [196] продолжил анализ, рассматривая различные линейные преобразования, – во всех случаях наблюдалась та же сходимость.

Здесь, можно также отметить и работу [5], в которой отмечается, что для шифра DES глубина лавинного эффекта равна 5-ти циклам, а этот же показатель для шифра ГОСТ 28147-89 соответствует 9-ти циклам.

Хейс и Таварес включили определенные предположения в свои модели, в первую очередь использование "идеализированных" S-блоков с

равномерной таблицей XOR (таких S-блоков не существует[4]). Тем не менее, одинаковые (идентичные) результаты можно получить, рассматривая проблему с другой точки зрения, которая не требует каких-либо предположений: значения, полученные в [5] равны ожидаемому числу битовых изменений в зашифрованном тексте для однобитного изменения открытого текста, где считалось, что все ключи независимы и при *всех выборах SPN S-блоков*, каждый S-блок выбирается независимо и равномерно из множества всех биективных $N \times N$ S-блоков.

В наших работах [38,39 и др.] как раз показывается, что значения максимумов полных дифференциалов и смещений таблиц линейных аппроксимаций для шифров с фиксированными ключами повторяют соответствующие значения максимумов для случайных подстановок, т.е. современные шифры можно считать случайными шифрами (повторяющими свойства случайных шифров) независимо от свойств ключевого материала.

Отдельно следует остановиться на работе [126]. В этой работе также рассматриваются SPN-ы со случайно выбранными S-боксами. Автор рассматривает линейные аппроксимации над $T \geq 2$ основными SPN циклами. Пусть

$a, b \in \{0, 1\}\mathrm{N} \setminus 0$ входная и выходная маски, соответственно, для этих основных циклов, M является числом S-боксов в отдельном цикле. В работе приводится два выражения для значения $ELP^{[1\ldots T]}(a,b)$ для SPN-ов с фиксированными S-блоками: первое следует из определения 2.6.2 [126], а второе из линейной оболочки Ниберг'а (теорема 3.2.6) [126]. Оба выражения, по всей видимости, трудно вычислить точно. Однако, рассматривая SPN-ы, в

[4] Поскольку значения ячеек XOR таблицы четные, и заполнения в каждой строке дают сумму 2^n, "наиболее равномерная" XOR таблица содержит 2^{n-1} нулей и $2^{n-1}-1$ двоек в любой строке индексированной ненулевым входным различием.

которых
S-блоки выбраны независимо и равномерно из множества всех $2^N!$ биективных $N \times N$ S-блоков, пишет автор цитируемой работы, можно получить формулу для *ожидаемого значения* ELP, где математическое ожидание вычисляется над всеми SPN-ами порожденными этой случайной выборкой S-блоков. В работе ищется ответ на вопрос: если SPN-блоки выбраны случайно и независимо и ключи выбраны случайно, то чему равно ожидаемое значение $LP^{[1 \dots T]}(a,b)$? Автор предлагает формулу и результаты вычислений по ней для некоторых SPN-схем с практическими размерами блоков. Результаты расчётов и наблюдений показывают, что полученные значения сходятся к соответствующему значению для случайных шифров с ростом T. Это, заключает Келихер, дает количественную поддержку утверждению, что SPN структура является практической аппроксимацией (приближением) к случайному шифру. Он считает, что эта сходимость имеет место для любого SPN шифра, у которого S-блоки и линейные преобразования удовлетворяют некоторым простым условиям. Таким образом, в рассмотренной работе автор тоже приходит к выводу о сходимости с ростом числа циклов свойств SPN шифров к случайному шифру, правда он останавливается на шифрах со случайными S-блоками.

4.7. Приближения Марковскими шифрами. Марковские шифры и марковские процессы.

Следует также напомнить концепцию Марковских шифров [74]. Теория Марковских шифров, как отмечается в [77] инспирировала многие исследования итеративных блочных шифров. Её можно рассматривать как первый подход к разработке блочных шифров, устойчивых к дифференциальному криптоанализу. Марковские шифры являются наиболее близкими к случайным шифрам.

Марковский шифр является (итеративным) шифром [77], для которого средняя вероятность распространения вероятности через один цикл не

зависит от циклового текстового входа. Для таких шифров предположение независимости цикловых подключей позволяет вычислить среднюю дифференциальную вероятность как произведение вероятностей индивидуальных циклов. Здесь средняя дифференциальная вероятность берётся над всеми цикловыми подключами, рассматриваемыми как независимые значения. Наш анализ показал, что известные подходы к заданию и описанию Марковских шифров являются не совсем аккуратными. Более того, ряд из представленных утверждений, связанных с Марковскими шифрами, представляются не корректными. Поэтому мы в этом разделе представим наши материалы по более строгому обоснованию определений Марковских шифров и уточнению ряда принципиальных моментов связанных с ними [197].

4.7.1. Обзор публикаций по Марковским шифрам.

Напомним сначала положения, относящиеся к Марковским шифрам, которые приведены в немногочисленных публикаций в этом направлении. Основополагающей здесь, по-видимому, следует считать совместную работу Лэя, Мэсси и Марфу [8] 1991 года. В этой работе рассматривается итеративный *r*-цикловый шифр представленный авторами в виде Рис. 4.7, на котором приведены необходимые нам обозначения.

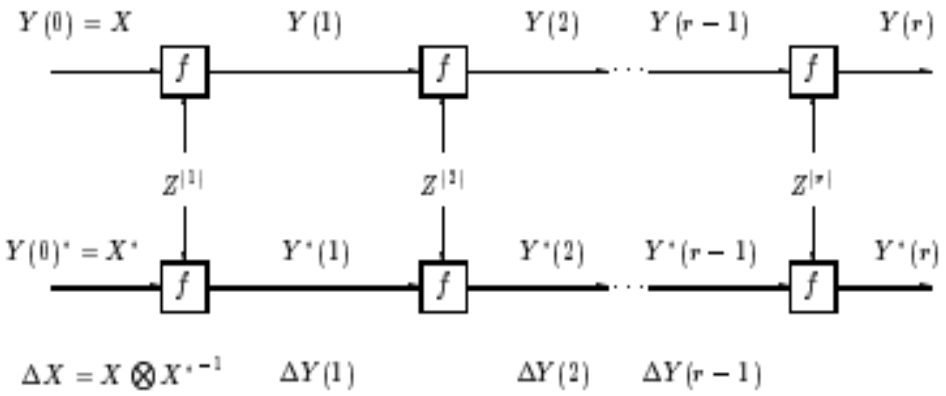


Рис.4.7. Шифрование пары плайнтектов в *r*-цикловом итеративном шифре

Приводится такое определение Марковского шифра.

Определение. *Итеративный шифр с цикловой функцией* $Y = f\{X, Z\}$ *является Марковским шифром, если имеется групповая операция* $\otimes$, *определяющая дифференциал такая, что для всех значений* $\alpha\,(\alpha \neq 0)$ *и* $\beta\,(\beta \neq 0)$ *условная вероятность*

$$P(\Delta Y = \beta \mid \Delta X = \alpha,\ X = \gamma)$$

является независимой от γ, *когда подключ* Z *является равномерно случайным.*

Далее приводится названная решающей теорема 2, которая объясняет, как указывают авторы, терминологию ″Марковский шифр″.

Теорема 2. *Если* r*-цикловый итеративный шифр является Марковским шифром и* r *цикловых ключей являются независимыми и равномерно распределёнными* (*случайными*), *то последовательность разностей* $\Delta X = \Delta Y(0),\ \Delta Y(1), \ldots, \Delta Y(r)$ *является однородной Марковской цепью. Более того, эта Марковская цепь является стационарной, если разности* ΔX *являются равномерно распределёнными над ненулевыми элементами группы.*

В этой работе шифр с операцией, определяющей разности, рассматривается как группа, ΔX является входной разностью, а $\Delta Y(0), \Delta Y(1), \ldots, \Delta Y(r)$ − поцикловые выходные разности.

В качестве примера Марковского шифра приводится шифр DES. Отмечается, что для Марковского шифра с независимыми и равномерно распределёнными (случайными) цикловыми подключами вероятность r-цикловой дифференциальной характеристики определяется уравнением Чепмена-Колмогорова для Марковской цепи:

$$P\left[\Delta Y(1) = \beta_1, \Delta Y(2) = \beta_2, \ldots, \Delta Y(r) = \beta_r, \Delta X = \beta_0\right] = \prod_{i=1}^{r} P\left[\Delta Y(1) = \beta_1 \mid \Delta X = \beta_{i-1}\right].$$

Из этого следует, что вероятность r-циклового дифференциала (β_{01}, β_r) есть

$$P\{\Delta Y(r)=\beta_r \mid \Delta X=\beta_0\}=\sum_{\beta_1}\sum_{\beta_2}\cdots\sum_{\beta r-1}\prod_{i=1}^{r}P\{\Delta Y(1)=\beta_1 \mid \Delta X=\beta_{i-1}\},$$

где суммы рассматриваются над всеми возможными значениями разностей между различными элементами, т.е. над всеми элементами группы, исключая нейтральный элемент *e*.

Заметим здесь, что в теореме 2 и последующих разъяснениях говорится о Марковских шифрах с независимыми и равномерно распределёнными (случайными) цикловыми подключами. Как станет понятно из дальнейшего, само понятие Марковского шифра включает отмеченные выше свойства цикловых подключей, так что специальное оговаривание для Марковского шифра независимости и случайности цикловых подключей является, конечно, лишним, т.е. аккуратнее было бы говорить от том, что итеративный шифр является Марковским, если *цикловые подключи являются независимыми и равномерно распределёнными.*

Полезно будет напомнить здесь также гипотезу статистической эквивалентности, представленную авторами в рассматриваемой работе:

Гипотеза статистической эквивалентности: *Для* $(r-1)$*-циклового дифференциала* (α,β)

$$P\{\Delta Y(r-1)=\beta \mid \Delta X=\alpha)\approx P\{\Delta Y(r-1)=\beta \mid \Delta X=\alpha, Z^{(1)}=\omega_1,\ldots,Z^{(r-1)}=\omega_{r-1}\}$$

почти для всех подключевых значений $(\omega_1,\ldots,\omega_{r-1})$.

Эта гипотеза в работе используется для обоснования условий уязвимости итеративного шифра к атакам дифференциального криптоанализа. Нам она понадобится для обоснования другого факта. Мы далее покажем, что эта гипотеза выполняется для всех и шифров, достигших стационарного состояния.

Приведём также теорему 3 этой работы:

Теорема 3. *Для Марковского шифра блочной длины m с независимыми и равномерно распределёнными цикловыми подключами, если полубесконечная Марковская цепь* $\Delta X=\Delta Y(0),\Delta Y(1),\ldots$ *имеет ″равномерно-*

устойчивое вероятностное" распределение, т.е. существует вероятностный вектор $(p_1, p_2, \ldots, p_M)$ *такой, что для всех* α_i $\lim_{r\to\infty} P\{\Delta Y(r) = \alpha_j \mid \Delta X = \alpha_i\} = p_j$, *то это равномерно-устойчивое распределение должно быть равномерным* $(1/M, 1/M, \ldots, 1/M)$, *т.е.* $\lim_{r\to\infty} P\{\Delta Y(r) = \alpha_j \mid \Delta X = \alpha_i\} = \frac{1}{2^m - 1}$ *для каждого дифференциала* (α, β), *так что каждый дифференциал является, грубо говоря, равновероятным для достаточно большого числа циклов. Если мы предполагаем дополнительно, что для этого Марковского шифра выполняется гипотеза статистической эквивалентности, то для почти всех подключей этот шифр безопасный против атаки дифференциального криптоанализа после достаточного числа циклов.*

Мы далее покажем, что первая часть утверждения этой теоремы не соответствует реальному состоянию дел, не говоря уже о том, что гипотеза статистической эквивалентности выполняется для Марковских шифров безусловно.

Приведём также выжимки из другой работы [77], в которой затрагиваются Марковские шифры. Это работа авторов L. Keliher-a, H. Meijer-a и S. Tavares-a. Мы далее привязываемся к обозначениям именно этой работы.

В [77] *R*-цикловый шифр определяется аналитически как отображение $\varepsilon : \{0,1\}^N \to \{0,1\}^N$, для которого цикл r задается функцией $\mathbf{y} = \varepsilon_r\{\mathbf{x}; \mathbf{k}^r\}$; $\mathbf{x} : \{0,1\}^N$ является цикловым входом, $\mathbf{k}^r : \{0,1\}^N$ является подключом *r*-того цикла. Тогда, отмечается в этой работе, при применении для сложения с ключом групповой операции XOR ($\oplus$) над $\{0,1\}^N$ *R*-цикловый шифр ε является Марковским шифром, если для $1 \le r \le R$ и любых $\mathbf{x}, \Delta\mathbf{x}, \Delta\mathbf{y} \in \{0,1\}^N$,

$$\begin{aligned} &\mathrm{Prob}_{\mathbf{K}}\{\varepsilon_r(\mathbf{x}; \mathbf{K}) \oplus \varepsilon_r(\mathbf{x} \oplus \Delta\mathbf{x}; \mathbf{K}) = \Delta\mathbf{y}\} = \\ &= \mathrm{Prob}_{\mathbf{X}, \mathbf{K}}\{\varepsilon_r(\mathbf{X}; \mathbf{K}) \oplus \varepsilon_r(\mathbf{x} \oplus \Delta\mathbf{x}; \mathbf{K}) = \Delta\mathbf{y}\}, \end{aligned} \tag{4.14}$$

где **X** и **K** независимые случайные значения, равномерно распределенные над $\{0,1\}^N$ и K множество всех независимых ключей соответственно (т.е. здесь по существу повторяется представленное выше определение).

Соотношение (4.14), отмечают авторы, определяет вероятность для ключа, который фиксированное входное различие преобразует в фиксированное выходное различие, не зависящие от циклового входа.

Легко показать, отмечается также в этой работе, что SPN шифры с фиксированными S-блоками являются Марковскими шифрами.

Из представленных материалов следует, что все подходы к заданию (описанию) Марковских шифров связываются с уравнениями для дифференциалов, что, как мы покажем далее, является не совсем аккуратным. Кроме того, ряд из представленных утверждений, как видно из замечаний, представленных по тексту, представляются не корректными. Мы в этом разделе работы поставили задачу более строгого обоснования Марковских шифров и уточнения ряда принципиальных моментов, связанных с ними.

В этом подразделе сначала представляется уточнённый подход к определению Марковских шифров, основывающийся на стохастических уравнениях Марковских процессов.

4.7.2. Уточнённый подход к определению Марковских шифров

В качестве введения мы здесь напомним небольшую работу [209], подготовленную ещё в 1978 году. В этой работе излагаются некоторые важные свойства дискретных и одновременно Марковских процессов, не освещённых в достаточной степени в литературе.

Мы сначала приведём здесь некоторые сведения из работы, подготовленной ещё в 1978 году, в которой излагаются некоторые важные свойства дискретных и одновременно Марковских процессов, не освещённых в достаточной степени в литературе.

В работе вводится понятие Марковского дискретного процесса *k*-того порядка. Рассматривается выборка процесса $y(t_i)$, $i = 1, 2\ldots, n$, заданного в виде последовательности $y_1, y_2, \ldots, y_n$ выборочных (или средних за элементарный интервал дискретности) значений исходного непрерывного процесса $y(t)$, заданного на некотором конечном интервале времени (T_1, T_2). Отмечается, что наиболее полной статистической характеристикой этого процесса является многомерный закон распределения вероятностей $P(y_1, y_2, \ldots, y_n)$ совокупности его выборочных значений. Определяется понятие Марковского процесса *k*-того порядка.

Определение. *Марковским процессом k-того порядка называется процесс, условный закон распределения вероятностей выборочных значений которого для каждого значения выборки* y_l, *относительно предыдущих значений* $y_{l-1}, y_{l-2}, \ldots, y_1$ *при любом* $l > k$ *зависит только от k предшествующих значений, т.е.*

$$P(y_l /, y_{l-1}, y_{l-2}, \ldots, y_1) = P(y_l /, y_{l-1}, y_{l-2}, \ldots, y_{l-k}).$$

Показано, что Марковский и одновременно нормальный процесс математически описывается стохастическим дифференциальным уравнением соответствующего порядка, дискретным аналогом которого является линейное неоднородное разностное уравнение со случайной правой частью [198].

$$y_l + \sum_{p=1}^{k} a_p y_{l-p} = \eta_l \text{ , для } l > k,\ a_p = a_p^{(l)}, \qquad (4.15)$$

Отметим, что при $l \le k$ имеем начальные условия

$$y_l + \sum_{p=0}^{l-1} a_p^{(l)} y_{l-p} = \eta_l \text{ , } \ a_0^{(l)} = 0.$$

В (4.15) $\eta_l -$ отсчёт (выборочное значение) случайного δ-коррелирован-ного процесса с нулевым математическим ожиданием и фиксированной дисперсией.

Нас далее будут интересовать сначала Марковские процессы первого порядка ($k = 1$), для которых

$$P(y_l /, y_{l-1}, y_{l-2}, \ldots, y_1) = P(y_l /, y_{l-1}). \quad (4.16)$$

В [197] показано, что простейший Марковский процесс первого порядка (экспоненциально коррелированный нормальный процесс) описывается стохастическим разностным уравнением:

$$y_l = -e^h y_{l-1} + \eta_l, \quad (4.17)$$

где $h = \frac{T_0}{\tau_k}$ (T_0 – интервал дискретности, τ_k – время корреляции процесса). Ему соответствует в непрерывном времени стохастическое дифференциальное уравнение первого порядка, но нас будет интересовать именно его представление, связывающее текущее значение дискретного процесса y_l с предыдущим отсчётным значением y_{l-1}.

Прежде чем идти дальше, полезно будет обобщить приведенные выше сведения следующим образом: *для Марковского процесса первого порядка соседние отсчётные значения процесса* (*два соседних значения*) *связаны между собой случайной компонентой, для Марковского процесса второго порядка три смежных отсчётных значения связаны между собой случайной компонентой, наконец, для Марковского процесса* k*-того порядка выборка из* $k + 1$*-го соседних отсчётных значений процесса связаны между собой случайной компонентой.*

Приведенные сведения и будут той основой, на которой мы будем строить определения для Марковских шифров.

Первое положение, которое мы хотим здесь обосновать, состоит в том, что практически любой современный шифр является шифром, формирующим в результате выполнения процедуры зашифрования Марковский процесс.

Мы здесь предлагаем свою, как нам кажется более строгую (последовательную), точку зрения к определению Марковских шифров.

Рассмотрим более детально SPN шифр, представленный на рис.2.1 раздела 2.

Если выделить в рассмотренном шифре сложение с цикловым подключом в отдельное преобразование (это можно сделать практически в любом SPN шифре), то одноцикловое преобразование такого шифра всегда можно представить в виде результата выполнения над входом в цикловую функцию биективного преобразования F (прохождение через S-блоки и линейное преобразование) и последующего сложения результата преобразования со случайной компонентой, определяемой цикловым подключом. т.е. блок данных на выходе цикловой функции будет иметь вид:

$$\mathbf{y} = \varepsilon_r\{\mathbf{x}; \mathbf{k}^{r+1}\} = F_r\{\mathbf{x}\} + \mathbf{k}^{r+1}. \qquad (4.18)$$

В результате мы приходим к аналогу уравнения (4.17), особенностью которого является то, что в уравнении (4.18) над цикловым входом $\mathbf{x}:\{0,1\}^N$ выполняется преобразование не линейного типа, как это сделано в уравнении (4.17), а нелинейное биективное преобразование, которое осуществляется в поле $\mathbf{x}, \mathbf{y} \in \{0,1\}^N$. Но это всё равно получается уравнение связывающее предыдущее значение входа $\mathbf{x}$ с текущим $\mathbf{y}$ (это уже новый вход в очередной цикл) с помощью случайной компоненты $\mathbf{k}^{r+1}$. Если полагать, что ключи для циклов выбираются равновероятно и независимо, то, конечно, это уравнение Марковского процесса (теперь, конечно же, отличающегося от нормального). В итоге предлагается определение Марковского шифра в виде:

Определение 1. *Марковским шифром* (*первого порядка*) *является* (*называется*) *шифр, для которого соотношение для выхода цикловой функции с её входом для любого значения входа и выхода* $\mathbf{x}, \mathbf{y} \in \{0,1\}^N$ *определяется нелинейным уравнением* (4.18), *в котором случайная компонента в правой части является ключевым значением* $\mathbf{k}^{r+1} \in \{0,1\}^N$, *выбранным независимо и равновероятно из всего множества возможных ключей.*

Аналогично можно ввести и определение Марковского шифра *k*-того порядка:

Определение 1′. *Марковским шифром* (*k-того порядка*) *называется шифр, для которого соседние значения выходов* k *цикловых функций и значение входа в первую цикловую функцию их этого набора связаны между собой нелинейно случайной компонентой* (*нелинейное уравнение, связывающее* k + 1 *соседних значений на выходах цикловых функций, содержит случайную компоненту*).

В этом случае мы имеем в виду уравнение, которое отличается от (4.15) нелинейной связью переменных, входящих в него.

Мы здесь привели определение Марковского шифра *k*-того порядка, так как нам потребуется в рамках этой работы и Марковские шифры второго порядка.

Установим (определим) теперь связь приведенных определений с определениями, известными из литературы.

Для входа в цикловую функцию $\mathbf{x}' = \mathbf{x} \oplus \Delta\mathbf{x}$, с учётом (4.16) имеем:

$$\mathbf{y}' = \varepsilon_r\{\mathbf{x} \oplus \Delta\mathbf{x}; \mathbf{k}^{r+1}\} = F_r\{\mathbf{x}'\} + \mathbf{k}^{r+1}.$$

В результате для дифференциалов (разностей) циклового преобразования $\mathbf{y} \oplus \mathbf{y}' = \Delta\mathbf{y}_r$, $\mathbf{x} \oplus \mathbf{x}' = \Delta\mathbf{x} = \Delta\mathbf{y}_{r-1}$ для одного и того же ключевого значения $\mathbf{k}^{r+1}$ приходим к уравнению

$$\Delta\mathbf{y}_r = F_r\{\mathbf{x}\} \oplus F_r\{\mathbf{x}'\} = F_r^*\{\Delta\mathbf{x}\} = F_r^*\{\Delta\mathbf{y}_{r-1}\}, \qquad (4.19)$$

В (4.19) F_r^* – функция циклового преобразования разностей.

В рассмотренных выше и других публикациях [8,75,79 и др.] понятие Марковского шифра связывают именно с уравнением для дифференциалов.

Ещё один пример. В [74] Марковским назван шифр, у которого уравнение шифрования на одном цикле удовлетворяет условию: вероятность дифференциала не зависит от выбора открытых текстов. Тогда, если подключи циклов между собой независимы, отмечается в [74], то

последовательность разностей после каждого цикла образует Марковскую цепь, где последующее состояние определяется только предыдущим.

Конечно, это и приведенные выше понятия согласуются с отмеченным определением Марковского процесса первого порядка для дифференциалов.

Но мы хотим сейчас привлечь внимание к имеющимся в литературе подходам к делению шифров на Марковские и немарковские. Нам представляется, что пропущенная многими авторами связь, выражаемая в виде уравнения, которое оперирует не с дифференциалами, а с соседними значениями одноцикловых переходов шифра, привела к не совсем аккуратной интерпретации свойств некоторых криптографических преобразований.

Так в [199] шифр ГОСТ 28147-89 относится к немарковским. Напомним, что в режиме простой замены шифра ГОСТ-а [200] 64-битный блок открытого текста (сообщения) разбивается на две части по 32 бита каждая (правая половина блока далее обозначена A_0, а левая B_0). Осуществляется 32 однотипных цикла преобразования, структура которых в каждом из циклов описывается выражениями

$$A_i = f(A_{i-1} + \bar{K}_j) \oplus B_{i-1}, \tag{4.20}$$

$$B_i = A_{i-1}, \tag{4.21}$$

причем для $i = \overline{1,24}$ берется $j = (i-1) \bmod 8$, для $i = \overline{25,31}$ соответственно $j = 32 - i$, и для последнего цикла

$$A_{32} = A_{31},$$

$$B_{32} = f(A_{31} + \bar{K}_0) \oplus B_{31},$$

где i – номер итерации; символом $+$ обозначена операция сложения по модулю 2^{32}.

Очевидно, что с учётом (4.21) соотношение (4.20) можно переписать в виде:

$$A_i = f(A_{i-1} + \bar{K}_j) \oplus A_{i-2}.$$

Легко убедиться, что по нашему второму определению мы пришли к уравнению Марковского процесса второго порядка (если считать цикловые подключи независимыми), и, следовательно, шифр ГОСТ 28147-89 тоже является Марковским. Представляется, что реально существующая корреляция подключей шифра существенно не изменит картину.

Марковским шифром второго порядка является также и шифр DES, для которого уравнения зашифрования ($R-$ правый полублок, $L-$ левый полублок) имеют вид [99]:

$$R_i = f(R_{i-1}, K_i) \oplus R_{i-2},$$

$$L_i = R_{i-1}.$$

Для $R'_i = f(R'_{i-1}, K_i) \oplus R'_{i-2}$, где $R'_i = \Delta R \oplus R_i$ имеем:

$$\Delta R_i = f^*(\Delta R_{i-1}) \oplus \Delta R_{i-2}. \qquad (4.22)$$

В работе [74] шифр DES причисляется к Марковским шифрам. На самом же деле, это справедливо лишь в том случае, если речь идёт о дифференциальной характеристике, использованной Э. Бихамом и А. Шамиром [99] при построении предложенной ими атаки на шифр, вероятность которой (характеристики) действительно приводится к произведению вероятностей цикловых переходов, характерному для частных дифференциалов Марковских шифров первого порядка. Напомним, что при построении своей атаки они воспользовались трёхблочными характеристиками обнуляющего типа (d = 1960 0000), для которых в (4.22) надо положить

$$\Delta R_{i-2} = \Delta R_{i-4} = \ldots = \Delta R_{i-2k} = 0,$$

при этом, естественно, что $f^*(\Delta R_{i-2}) = f^*(\Delta R_{i-4}) = \ldots = f^*(\Delta R_{i-2k}) = 0$.

Для характеристик обнуляющего типа соответственно $\Delta R_{i-1} = \Delta R_{i-3} = \ldots = \Delta R_{i-2k-1} = d$, в то время как $f^*(\Delta R_{i-1}) = f^*(\Delta R_{i-3}) = \ldots = f^*(\Delta R_{i-2k-1}) = 0$.

В результате:

$$\Delta R_i = f^*(\Delta R_{i-1}) = 0 \text{ с вероятностью } p,$$

$$\Delta R_{i-1} = f^* \left(\Delta R_{i-2} \oplus \Delta R_{i-3} \right) \rightarrow \Delta R_{i-1} = \Delta R_{i-3} = d \text{ с вероятностью 1,}$$

$$\Delta R_{i-2} = f^* \left(\Delta R_{i-3} \right) = 0 \text{ с вероятностью } p,$$

$$\Delta R_{i-3} = \Delta R_{i-5} = d \text{, с вероятностью 1 . . . ,}$$

т.е. в этом случае мы действительно приходим скорее не к уравнениям дифференциалов Марковского шифра первого порядка, а к результирующей вероятности дифференциальной характеристики выражаемой в виде произведения вероятностей однотипных цикловых переходов (рассматривается частная дифференциальная характеристика), причём половина из них происходят без снижения вероятности (с вероятностью единица).

Но самое интересное, так это то, что и Марковские шифры первого порядка и Марковские шифры второго порядка приходят к одному и тому же стационарному состоянию: таблицы полных дифференциалов и линейных корпусов шифров асимптотически повторяющему законы распределения вероятностей XOR переходов и смещений таблиц линейных аппроксимаций случайных подстановок соответствующей степени. Но об этом чуть позже.

Возвращаясь к теореме 2, мы здесь хотим обратить внимание на присутствующее в её формулировке требование независимости дифференциалов от выбора открытых текстов (ΔX являются равномерно распределёнными над ненулевыми элементами группы) и утверждение в теореме о стационарности Марковской цепи.

Так вот, если рассматривать теорию Марковских процессов [197 и др.], то любой случайный процесс имеет своё начало и конец. Например, простейший нормальный Марковский процесс (4.17) не сразу приобретает показатели стационарного распределения. Начальное его значения следует рассматривать как нестационарное. Для Марковского процесса *k*-того порядка (см. начальные условия для уравнения (4.15)) будет уже переходный процесс из *k* смежных значений (для нормального процесса).

Так и в шифрах. На основе многочисленных результатов проведенных экспериментов мы здесь утверждаем, что большинство современных шифров (использующих для введения циклового подключа не только групповую операцию XOR) являются Марковскими (текущее значение шифртекста является функцией конечного числа предыдущих значений шифртекстов). Для каждого такого шифра существует определённое (небольшое) число начальных циклов, после которого законы распределения переходов XOR таблиц (дифференциалов) и смещений таблиц линейных аппроксимаций шифра приходят к установившемуся (стационарному) значению. На первых шагах шифрования Марковскому шифру присущ переходный период, который вполне согласуется с положениями теории Марковских процессов. Шифр приходит к стационарному процессу (состоянию) асимптотически.

И ещё в отношении равномерности распределения входных разностей ΔX над ненулевыми элементами группы, отмеченными в теореме 2. Дело в том, что уравнение (4.15) и соответствующее ему уравнение (4.17) определяют Марковский процесс при любом значении входа $\mathbf{x} \in \{0,1\}^N$. Множество равновероятных значений входов необходимо лишь для формирования закона распределения переходов XOR таблицы шифра, повторяющего распределение дифференциалов случайной подстановки, которое устанавливается после переходного периода.

Остаётся отметить, что после публикации линейного криптоанализа, теория Марковских шифров была распространена на сопротивляемость линейному криптоанализу, что привело к аналогичным выводам для линейных приближений (корпусов) шифров [58]. Для нас в этом нет ничего нового. Наши исследования [39, 183 и др.] и в этом случае свидетельствуют, что линейные показатели шифров при росте числа циклов приходят к соответствующим показателям случайных подстановок (Марковский шифр приходит к стационарному состоянию и по этому показателю). Напомним здесь, что равенства, используемые при построении линейных аппроксимаций шифров, определяют связь соседних значений входов и

выходов цикловых функций (прошедших соответствующие маски) через случайную компоненту (сумму ключевых бит).

Здесь мы хотим остановиться, хотя у нас есть ещё претензии к использованию матриц переходных вероятностей при оценке показателей стойкости шифрующих преобразований (Марковских шифров), да и в целом к развиваемой во многих работах самой методике оценки стойкости БСШ к атакам дифференциального и линейного криптоанализа. Мы на них остановимся в следующем разделе.

4.8. К теоретическому обоснованию сходимости шифров к свойствам случайной подстановки

Нас в дальнейшем будет интересовать именно момент (число циклов), начиная с которого шифрующее преобразование становится случайной подстановкой. Именно в этом направлении и будет строиться доказательство (обоснование) представленного выше утверждения 3.

Продемонстрируем справедливость этого утверждения на примере рассмотрения дифференциальных показателей итеративного шифра. В качестве такого показателя будет выступать максимальное значение полного дифференциала, которое устанавливается после выполнения шифром некоторого начального числа циклов зашифрования.

Доказательство этого утверждения (скорее не доказательство, а объяснение его правомерности) начнём с конца, т.е. предположим, что БСШ имеет некоторое определенное число циклов, после которых шифр становится случайной подстановкой, т.е. обладает законом распределения вероятностей переходов разностей (3.2).

Покажем, что дальнейшее наращивание числа циклов не влияет на итоговые дифференциальные свойства этого шифра.

Важно сразу отметить, что особенностью случайной подстановки, удовлетворяющей критерию 4, является то, что речь идёт не о фиксированном распределении переходов наборов разностей $\Delta x \to \Delta y$

(закрепленным распределением значений входов (ячеек) таблицы XOR разностей), а о случайном. Таблица XOR разностей случайной подстановки определяется тем, что для нее является фиксированными числа ячеек каждого типа, определяемые с помощью закона распределения $\Pr\{\Lambda_f(\Delta x,\Delta y)=2k\}$ в виде [43, 173]:

$$\Lambda_{n,2k}=(2^n-1)^2\cdot\Pr(\Lambda_f(\Delta x,\Delta y)=2k)= \\ =\frac{(2^n-1)^2}{2^n!}\cdot\binom{2^{n-1}}{k}^2\cdot k!\cdot 2^k\cdot\Phi(2^{n-1}-k) \qquad (4.23)$$

В соответствии с этим соотношением таблица XOR разностей случайной подстановки имеет λ_0 ячеек, имеющих значение $\Lambda_{n,0}$, λ_1 ячеек, имеющих значение $\Lambda_{n,2}$, λ_2 ячеек, имеющих значение $\Lambda_{n,4}$, и т.д., – λ_k^* ячеек, имеющих значение $\Lambda_{n,2k^*}$. Все эти значения вместе дают общее число входов (ячеек) в подматрицу таблицы XOR разностей равное $2^{n-1}\times 2^{n-1}$, причем сами числа λ_0, λ_1, λ_2, . . . , λ_k^* определяются однозначно соотношениями (3.2), (3.3) и (3.8).

Применительно к шифрующим многоцикловым преобразованиям – случайным подстановкам дифференциальные вероятности DP^f должны теперь интерпретироваться в обозначениях подстановочных преобразований для ключезависимой функции f как

$$DP^f(\Delta x,\Delta y)=DP^f(\Delta x\to\Delta y)= \\ =\Pr\{\Lambda_f(\Delta x,\Delta y)=2k\}, \qquad (4.24)$$

причем эти вероятности следует считать одинаковыми для всех ячеек таблицы XOR разностей (для всех вариантов сочетаний входных и выходных разностей).

Возвратимся к нашей задаче. Итак, пусть r-цикловое шифрующее преобразование (последовательность r цикловых преобразований) f_r с n-битным размером входа (и выхода) обладает свойством 4, т.е. закон

распределения $DP^{f_r}(\Delta x, \Delta y)$ переходов входных разностей Δx в выходные разности Δy имеет вид (3.2) с нормировкой

$$\sum_{k=0}^{k^*} DP^{f_r}\ \Delta x, \Delta y\ = 1.$$

Тогда, если на входы очередного циклового преобразования (подстановки) поступают некоторые сочетания пар выходов предшествующего преобразования случайного типа (предшествующей случайной подстановки) подчиняющиеся закону распределения XOR разностей таблицы полных дифференциалов (3.2), то цикловое преобразование может осуществить лишь переименование выходов и соответствующих им разностей, оставляя результирующий закон распределения разностей неизменным (для операции XOR подстановка вместе с линейным цикловым преобразованием являются детерминированными преобразованиями и произведение случайной в оговоренном смысле подстановки на любую другую подстановку, является случайной подстановкой). Приведем математическое обоснование этого факта (который подтверждается многочисленными экспериментами с малыми шифрами).

Нас интересует закон распределения вероятностей $DP^{f_{r+1}}(\Delta x, \Delta z)$ для $r + 1$ цикла преобразований, где Δz является выходной разностью $r + 1$-го циклового преобразования. У нас имеется цепочка $\Delta x \to \Delta y \to \Delta z$ разностей, совместный закон распределения вероятностей для которой обозначим $DP^{f_{r+1}}(\Delta x, \Delta y, \Delta z) = DP^{f_{r+1}}(\Delta x \to \Delta y \to \Delta z)$. В соответствии с формулой умножения вероятностей можем записать представление для этой вероятности в виде

$$DP^{f_{r+1}}(\Delta x, \Delta y, \Delta z) =$$
$$= DP^{f_r}(\Delta x, \Delta y) DP^{f_1}(\Delta z \,/\, \Delta x, \Delta y).$$

И тогда дифференциальная вероятность $DP^{f_{r+1}}(\Delta x, \Delta z)$ для $r + 1$-го циклового преобразования может быть определена из совместной

вероятности $DP^{f_{r+1}}(\Delta x, \Delta y, \Delta z)$ путем ее усреднения по множеству промежуточных значений $\Delta y \in Z_2^n$, т.е.

$$DP^{f_{r+1}}(\Delta x, \Delta z) = \\ = \sum_{\Delta y \in Z_{2^n}} DP^{f_r}(\Delta x, \Delta y) DP^{f_1}(\Delta z / \Delta x, \Delta y).$$

Но в нашем случае закон распределения

$$DP^{f_r}(\Delta x, \Delta y) = Pr(\Lambda_f(\Delta x, \Delta y) = 2k)$$

является одним и тем же для каждой выходной разности r-циклового преобразования (для каждой ячейки таблицы XOR разностей случайной подстановки), а поэтому

$$DP^{f_{r+1}}(\Delta x, \Delta z) = \\ = DP^{f_r}(\Delta x, \Delta y) \sum_{\Delta y \in Z_{2^n}} DP^{f_1}(\Delta z / \Delta x, \Delta y).$$

Очевидно далее, что при фиксированных значениях Δy выходные разности Δz не зависят от того, какие значения принимают входные разности Δx и, следовательно,

$$\sum_{\Delta y \in Z_{2^n}} DP^{f_1}(\Delta z / \Delta x, \Delta y) = \\ = \sum_{\Delta y \in Z_{2^n}} DP^{f_1}(\Delta z / \Delta y) = \sum_{\Delta y \in Z_{2^n}} DP^{f_1}(\Delta y \to \Delta z).$$

Но в соответствии с (4.5) для подстановочного одноциклового преобразования f_1

$$\sum_{\Delta y \in Y} DP^{f_1}(\Delta x, \Delta y) = \\ = \sum_{\Delta y \in Y} DP^{f_r}(\Delta x \to \Delta y) = 1,$$

и, в итоге, приходим к результату

$$DP^{f_{r+1}}(\Delta x, \Delta z) = DP^{f_r}(\Delta x, \Delta y) \Rightarrow \\ \Rightarrow DP^{f_{r+1}}(\Delta x \to \Delta z) = DP^{f_r}(\Delta x \to \Delta y),$$

где

$$DP^{f_r}(\Delta x \to \Delta y) = Pr(\Lambda_f(\Delta x, \Delta y) = 2k).$$

Последнее и обозначает, что дополнительные цикловые преобразования уже не меняют закона распределения разностей на выходе шифра.

Остается теперь прокомментировать первую часть утверждения. Для этого заметим, что эффективность перемешивания входного текста при зашифровании в криптографии оценивается такими параметрами статистической безопасности, как лавинный эффект, коэффициент сжатия, ряд корреляционных показателей [201].

Если рассматривать тонкую структуру циклового преобразования, то в самом начале процедуры зашифрования (в первом цикле) при применяемых при построении большинства шифров решениях, как правило, не удается реализовать связь каждого выходного бита циклового преобразования с каждым входным битом. Например, биты входа влияют на выход только одного

S-блока многоблочного нелинейного преобразования, а используемое последующее линейное преобразование не обладает полнотой в том смысле, что оно передает воздействие входа не на все выходы текущего преобразования. Для характеристики этого свойства разработчики шифра Rijndael ввели специальную характеристику − коэффициент ветвления, а сам механизм распространения активных битов (S-блоков) в последовательных слоях (циклах) преобразований назвали стратегией широкого следа [202]. Но эту, же, стратегию пытались реализовать все разработчики известных шифров, хотя она была часто не такой эффективной, как, скажем, у Rijndael-я (умножение выходов нескольких S-блоков на матрицу МДР кода). Естественно, что если есть механизм расширения числа активных (задействованных) в ходе преобразования битов блока данных, то рано или поздно наступит момент, когда любой бит входа будет одинаково эффективно действовать на любой бит выхода. Этот момент как раз и будет обозначать, что шифрующее преобразование стало случайной подстановкой

(результирующий закон распределения переходов разностей пар входов в соответствующие им разности пар выходов принимает вид (3.2)).

4.9. Оценки максимальных значений полных дифференциалов и линейных корпусов Марковских шифров.

В этом разделе мы приведём свидетельства справедливости предлагаемой идеологии формирования показателей доказуемой стойкости блочных симметричных шифров. Будут приведены результаты экспериментов демонстрирующих приход шифров с ростом числа циклов к показателям случайных подстановок соответствующих степеней. Одновременно мы хотим в этом разделе привести дополнительные аргументы и уточнения ряда принципиальных моментов, связанных с понятиями и определениями теории Марковских шифров.

Выше было введено определение Марковского шифра *k*-того порядка и сделан общий вывод о том, что все известные итеративные блочные шифры являются Марковскими шифрами (первого или второго порядка). В этом разделе мы продолжаем обсуждение состояния ряда теоретических и практических вопросов в этом направлении, и здесь мы хотим привлечь внимание к подходам, имеющимся в публикациях, посвящённым оценкам максимальных значений дифференциалов и линейных корпусов Марковских шифров.

Достаточно детальный анализ широко эксплуатируемой сегодня концепции оценки показателей доказуемой стойкости блочных симметричных шифров представлен во втором разделе монографии. Там освещены подходы, строящиеся на концепции, что показатели стойкости блочных симметричных шифров определяются соответствующими дифференциальными и линейными показателями S-блоков, входящих в шифр.

Здесь, однако, мы отметим две работы, в которых оценки стойкости Марковских шифров строятся не на S-блоках, а на основе использования

матриц переходных вероятностей одноцикловых преобразований (практически опять идёт связь с S-блоками).

Например, в [74] авторы оперируют матрицами переходных вероятностей для цепей Маркова и рассматривают результат многоциклового преобразования с помощью возведения матрицы переходных вероятностей одноциклового преобразования в степень, равную числу циклов преобразования.

В другой работе [199] обосновывается сходимость последовательности матриц вероятностей дифференциальных аппроксимаций немарковского блочного шифра к равновероятной матрице при увеличении количества раундов. Здесь немарковским считается шифр ГОСТ, и опять итоговый результат строится на возведении матрицы переходных вероятностей одноциклового преобразования в степень равную числу циклов (раундов). Это, конечно, и в первом и во втором случаях не верно, как и не верно утверждение о сходимости последовательности матриц средних вероятностей немарковского шифра к равновероятной.

Возведение в степень справедливо лишь в том случае, когда с отсчётными значениями выборки, входящими в уравнение Марковского процесса, осуществляется линейное преобразование или когда рассматривается частная дифференциальная характеристика шифра, как, например, в атаке на шифр DES Э. Бихама и А. Шамира. Когда речь идёт о полных дифференциалах, то для практически всех известных итеративных шифров этого делать нельзя, так такие шифры цикловое преобразование строят с использованием нелинейных операций (S-блоков). Заметим, что шифры без S-блоков (например, [95]) тоже укладываются в эту общую схему (там тоже используются нелинейные операции).

Остановимся на этом принципиальном моменте более детально.

Начнём с Марковского процесс первого порядка, заданного уравнением для дифференциалов (см. подраздел 4.7):

$$\Delta \mathbf{y}_r = F_r\{\mathbf{x}\} \oplus F_r\{\mathbf{x} \oplus \Delta \mathbf{x}_r\} = F_r\{\mathbf{x}\} \oplus F_r\{\mathbf{x}^*\} = F_r^*\{\Delta \mathbf{x}_r\} = F_r^*\{\Delta \mathbf{y}_{r-1}\}, \qquad (4.25)$$

В (4.25) F_r^* – функция циклового преобразования разностей ($\Delta\mathbf{y}_{r-1} = \Delta x_r$).

Как следует из уравнения (4.25), при вычислении выходной разности $\Delta\mathbf{y}_r$ ключ последнего цикла $\mathbf{k}^{r+1}$, как и ключи предшествующих циклов вроде бы де уходят из уравнения (компенсируется). Однако это не так. Ключи текущего и предшествующих циклов шифрования (случайные компоненты) присутствуют в этом преобразовании через значения промежуточных разностей, участвующих в формировании разности шифртекстов на его выходе. И для выходной разности это, конечно, ключезависимый результат.

Тем не менее, решение уравнения (4.25) для фиксированного набора цикловых подключей можно представить в виде

$$\mathbf{y}_r \oplus \mathbf{y}_r^* = F_r\{\mathbf{y}_{r-1}\} \oplus \mathbf{k}^{r+1} \oplus F_r\{\mathbf{y}_{r-1}^*\} + \mathbf{k}^{r+1} = F_r\{\mathbf{y}_{r-1}\} \oplus F_r\{\mathbf{y}_{r-1}^*\} = F_r^*\{\Delta\mathbf{y}_{r-1}\}$$

и далее

$$\begin{aligned} F_r^*\{\Delta\mathbf{y}_{r-1}\} &= F_r\{\mathbf{y}_{r-1}\} \oplus F_r\{\mathbf{y}_{r-1}^*\} = F_r\{F_{r-1}\{\mathbf{y}_{r-2} \oplus \mathbf{k}^r\}\} \oplus F_r\{F_{r-2}\{\mathbf{y}_{r-1}^*\} \oplus \mathbf{k}^r\} = \\ &= F_r^*\{F_{r-1}^*\{\mathbf{y}_{r-2} \oplus \mathbf{k}^r \oplus F_{r-1}\{\mathbf{y}_{r-2}^*\} \oplus \mathbf{k}^r\} = F_r^*\{F_{r-1}^*\{\Delta\mathbf{y}_{r-2}\}\} = \\ &= \ldots = F_r^*\{F_{r-1}^*\{\cdots F_1^*\{\Delta\mathbf{y}_0\}\cdots\}, \end{aligned}$$

т.е.

$$\Delta\mathbf{y}_r = F_r^*\{F_{r-1}^*\{F_{r-2}^*\{\ldots\{F_1^*\{\Delta\mathbf{x}_1\}\}\ldots\}, \ \Delta\mathbf{x}_1 = \Delta\mathbf{y}_0. \qquad (4.26)$$

Можно далее ввести в рассмотрение полное множество значений входных и выходных разностей и тогда произведение преобразований $F_r^* \cdot F_{r-1}^* \cdot F_{r-2}^* \cdot \ldots \cdot F_1^*$ рассматривать как матрицу переходных вероятностей, например в виде:

$$F_r^* \cdot F_{r-1}^* \cdot F_{r-2}^* \cdot \ldots \cdot F_1^* = \begin{vmatrix} f_{0,0} & f_{0,1} & \cdots & f_{0,2^{n-1}} \\ f_{1,0} & f_{1,1} & \cdots & f_{1,2^{n-1}} \\ \vdots & \vdots & \vdots & \vdots \\ f_{2^{n-1},0} & f_{2^{n-1},1} & \cdots & f_{2^{n-1},2^{n-1}} \end{vmatrix}.$$

В этой матрице каждый элемент $f_{i,j}$ определяет вероятность перехода i-той разности $\Delta\mathbf{x}_i$ на входе шифра в j-тую выходную разность $\Delta\mathbf{y}_j$ (в данном случае для r циклов зашифрования). Для марковского шифра, достигшего стационарного состояния, значения элементов матрицы подчиняются закону распределения вероятностей переходов случайной подстановки соответствующей степени (матрица имеет вполне определённое число нулей, число двоек, четвёрок и т.д.). Наращивание числа циклов в этом случае приводит к матрице с другим размещением элементами, но для неё всё равно сохраняется закон распределения переходов (она снова имеет прежнее число нулей, двоек, четвёрок и т.д., вплоть до единственного, как правило, максимального значения для определённого выхода). Когда мы говорим для Марковского шифра о независимости результата от выбора открытых текстов, то имеется в виду, что речь идёт о переходе входной разности $\Delta\mathbf{x}_0$ в выходную разность $\Delta\mathbf{y}_r$, который не зависят от выбора открытых текстов.

Остаётся отметить, что для каждого числа циклов и каждого ключа зашифрования будет своя матрица переходных вероятностей (нормированная таблица дифференциальных разностей) и результирующая матрица переходных вероятностей для r циклового Марковского шифра будет определяться, как это следует из (4.25), на основе произведения подстановочных преобразований разностей для одноцикловых переходов.

Большинство известных итеративных шифров используют однотипные цикловые преобразования. Поэтому результирующее подстановочное преобразование для таких шифров будет действительно представлять собой соответствующую степень одноциклового преобразования. Здесь надо помнить, что произведение подстановочных преобразований есть последовательное их выполнение одного за другим. Таким образом, мы пришли к последовательному выполнению r однотипных преобразований:

$$F_r^* = (F_1^*)^r, \qquad (4.27)$$

но не к матричному возведению в степень.

Таким образом, попытки использования для вычисления оценок показателей стойкости Марковских шифров возведения в степень матриц переходных вероятностей для цепи Маркова представляются ошибочными. Мы здесь подошли к обоснованию ещё одного факта. В ряде работ (например, [208] и др.) произведение матриц переходных вероятностей используется для доказательства прихода шифров к асимптотически равномерным распределениям значений полных дифференциалов и линейных корпусов Марковских шифров. Из представленных в предыдущем разделе результатов следует, что все известные шифры асимптотически приходят к показателям случайных подстановок, и увеличение числа циклов зашифрования не приводит к изменению этого стационарного состояния. Очевидно, что соответствующие распределения вряд ли можно считать равномерными. Повторением шифром свойств случайной подстановки и является основным показателем эффективности шифрующего преобразования.

Что касается Марковских шифров второго порядка, то сходимость их к Марковским шифрам первого порядка предстоит ещё доказать. Это задача отдельного исследования.

4.10. Сходимость показателей доказуемой стойкости итеративных шифров к дифференциальным и линейным показателям случайных подстановок соответствующей степени.

Теперь можно перейти к демонстрациям правомерности предлагаемой новой методологии оценки показателей доказуемой стойкости шифров (Марковских шифров) к атакам дифференциального и линейного криптоанализа

Теперь можно перейти к иллюстрации справедливости приведенных в разделе положений с помощью выполненных нами вычислительных экспериментов. Заметим, что предлагаемая методика разрабатывалась на основе использования уменьшенных моделей шифров. Однако сегодня уже

стало ясно, что многие из вычислительных экспериментов можно выполнить с использованием больших прототипов. Вопросам обоснования правомерности перехода и больших шифров асимптотически к показателям случайных подстановок будет посвящён отдельный раздел монографии. Тем не менее, мы посчитали уместным здесь представить результаты экспериментов сразу с большими шифрами.

Мы приведём здесь некоторые примеры анализа дифференциальных и линейных свойств (оценки *AMDP* и *AMLHP*) двух известных шифров AES и ГОСТ 28147-89 и ещё двух шифров Калина и Мухомор, представленных на украинский конкурс (ещё раз подчеркнём, что рассматривались полные версии этих шифров). Длина ключа и блока для шифра AES взята равной 256 битам, а для шифров Калина и Мухомор длина ключа и блока взяты одинаковыми и равными 128 битам.

В таблице 4.5 представлены поцикловые средние значения максимумов полных дифференциалов этих шифров, полученные при использовании шифров в режиме шифрования 16-битных блоков данных (в качестве результатов шифрования из зашифрованного блока данных тоже вырезаются 16-битные сегменты) [203].

В таблице 4.6 представлены поцикловые средние значения максимумов смещений линейных корпусов этих же шифров.

Таблица 4.5

Поцикловые средние значения максимумов полных дифференциалов (AMDP) шифров

Кол-во циклов	AES	ГОСТ 28147-89	Калина	Мухомор
1	65536	65536	65536	**19,4**
2	3652,26	65536	1382	19,2
3	**19,0666**	61952	**18,8**	19,6
4	19,0666	56008.6	18,8	19,2
5	18,8666	31358	19,2	19
6	19,1332	2046,7	18,8	19,2
7	19,2666	973,4	18,6	19,8
8	19,1332	52,2	18,8	19

9	19,0666	**19,1**	19	19,2
10	19,3333	19,5	18,8	19,6
11	19.4	18,7	19,4	19
13	18,8666	19,1	18,6	18,6
14	18.9332	19,4	19	19,2

Таблица 4.6

Поцикловые средние значения максимумов смещений линейных корпусов (*AMLHP*) шифров

Кол-во циклов	AES	ГОСТ 28147-89	Калина	Мухомор
1	0	0	11008,392	**824,742**
2	9284.27	32768	**817,271**	818,621
3	**818.467**	17162	817,718	827,431
4	815	31181,7	814,19	824,193
5	818.5	16150,1	837,349	831,753
6	815.967	16669,5	810,733	814,155
7	832.1	2144,77	820,384	820,975
8	823.133	2380,93	837,917	823,024
9	829.9	**826,833**	809,273	810,196
10	827.4	828,1	821,755	821,316
11	815.6	823,767	827,462	822,385
12	819	821,433	820,291	816,753

Для всех трех шифров был выполнен подсчет максимумов дифференциалов и смещений линейных корпусов с использованием 10 различных случайно сгенерированных ключей. Из табл. 4.5 видно, что шифр AES повторяет по динамике перехода к стационарному состоянию поведение шифра Калина, но уступает шифру Мухомор.

Из табл. 4.6 следует, что по линейным показателям шифр AES уступает в динамике обоим шифрам украинского конкурса. В таблицах 4.7 и 4.8 представлены результаты экспериментов ещё с тремя шифрами: Rijndael, Serpent, Threefish. Опять взяты полные версии данных шифров. Длина ключа и блока для Rijndael и Serpent одинакова и равна 128 битам, а в реализации шифра Threefish использовалась длина для блока и ключа равная 512 битам. Заметим, что в шифре Threefish не применяются S-блоки (в явном виде).

Таблица 4.7

Поцикловые значения максимумов полных дифференциалов для 16-битных сегментов

Число циклов, *r*	MAX (Rijndael)	MAX (Serpent)	MAX (Threefish)
1	16384	18,93	65536
2	8904,25	19,24	65536
3	1911.47	18,64	65536
4	19,24	18,33	42440,04
5	20,31	18,75	30704,23
6	18,83	19,21	9534,57
7	19,21	18,98	37,75
8	19,4	18,37	19,27
9	18,33	19,24	18,78
10	19,17	19,63	18,44

Для всех трех шифров и в этом случае был выполнен подсчет максимумов полных дифференциалов и смещений линейных корпусов с использованием 10 различных случайно сгенерированных ключей.

Представленные результаты хорошо иллюстрируют переход шифров к

Таблица 4.8

Математические ожидания максимальных смещений линейных корпусов полных моделей шифров

Число циклов, *r*	MAX (Rijndael)	MAX (Serpent)	MAX (Threefish)
1	0	810,4	32768
2	16313,36	825,0667	32680,93
3	7728,66	828,2667	31306,13
4	817,43	825,9333	23730,93
5	821,98	828,4667	19722,67
6	825,716	824,8667	19722,67
7	817,367	820,3333	7899,8
8	820,167	817,5333	844,0667

9	821,767	820,4	822,1333
10	820,167	816,6	815,8

стационарному состоянию, повторяющему характеристики случайных подстановок соответствующей степени [43,44] (наиболее ″медленным″ оказался шифр Threefish), а вот шифр Мухомор вышел на асимптотические показатели сразу после первого цикла.

Остановимся теперь дополнительно на одном из принципиальных моментов рассматриваемого подхода – приходу шифров к стационарному состоянию свойственному случайной подстановке, ставшим для многих неожиданным.

Выше получен результат (4.26), в соответствии с которым для Марковского шифра первого порядка формирование матрицы переходных вероятностей всего шифра сводится к последовательному выполнению r однотипных (одноцикловых) преобразований (см. соотношение (4.27)).

По результатам экспериментов можно сделать вывод о том, что *произведение одноцикловых преобразований после небольшого начального числа их повторений приобретает свойства случайной подстановки соответствующей степени независимо от показателей случайности исходного одноциклового преобразования*.

Мы не смогли найти теоретического обоснования этому свойству. Удалось лишь показать [42], что после достижения стационарного распределения при дальнейшем наращивании числа циклов это стационарное распределение сохраняется. Поэтому мы здесь представляем дополнительные исследования по обоснованию правомерности приведенного утверждения. Мы заинтересовались процессами, происходящими при последовательном выполнении подстановочных преобразований.

Для подтверждения наших слов мы хотим далее привести результаты вычислительного эксперимента не с шифрами, а с подстановками 256-й степени (байтовыми подстановками). В таблице 4.9 представлены результаты

вычислительного эксперимента по определению максимумов XOR таблиц последовательности подстановочных преобразований для двух байтовых подстановок. Одна подстановка взята с показателем δ-равномерности равным

4-ём, а вторая с показателем δ-равномерности равным 8-и. Видно, что обе подстановки уже на втором цикле приходят к максимуму дифференциала равному 10-12, характерному для случайной подстановки степени 2^8 [43].

Интересно отметить, что результат не зависит от ключевых значений, если их ввести после каждого подстановочного преобразования.

Конечно, по законам комбинаторики этот процесс должен быть периодическим, но для интересующих нас значений мы, как правило, оказываемся очень далеко от циклового периода подстановки.

Таблица 4.9

Распределение максимумов XOR таблиц последовательности подстановочных преобразований байтовой подстановки

Число циклов (повторов)	1	2	3	4	5	6	7	8	9	10	11
Значение максимума XOR таблицы для AES S-блока	4	12	12	10	12	12	10	12	12	12	12
Значение максимума XOR таблицы для S-блока Мухомор	8	10	10	12	10	14	12	12	10	12	10

Далее мы представляем результат возведения подстановочного преобразования (10 2 0 6 15 1 12 4 14 11 7 13 9 5 3 8) в квадрат (произведение одинаковых полубайтовых подстановочных преобразований:

(10 2 0 6 15 1 12 4 14 11 7 13 9 5 3 8)×(10 2 0 6 15 1 12 4 14 11 7 13 9 5 3 8) =

= (7 0 10 12 8 2 9 15 3 13 4 5 11 1 6 14).

Ниже представляется результат произведения матриц переходов этого подстановочного преобразования (возведение матрицы переходных вероятностей в квадрат), правда в нашем случае элементы матрицы не

вероятности, а значения вероятностей умноженные на 2^{16} (значения переходов дифференциальной таблицы).

$$\begin{vmatrix}
16&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&2&2&0&4&0&2&2&2&0&2&0\\
0&0&0&2&2&2&2&0&0&2&4&0&0&2&0&0\\
0&2&2&2&0&0&2&0&0&0&0&2&4&2&0&0\\
0&0&2&2&2&4&0&2&0&0&0&0&2&0&2&0\\
0&0&2&0&2&0&0&0&0&0&2&4&0&2&2&2\\
0&0&0&2&0&0&4&2&2&0&0&0&0&2&2&2\\
0&2&2&0&2&0&2&0&2&2&0&0&0&0&4&0\\
0&0&0&0&4&0&2&2&0&2&0&2&2&0&0&2\\
0&4&0&0&2&0&0&2&2&0&2&0&2&2&0&0\\
0&2&0&0&0&2&0&0&0&2&0&0&2&4&2&2\\
0&0&2&0&0&2&0&4&2&2&0&2&0&2&0&0\\
0&2&0&4&0&0&0&2&0&2&2&2&0&0&2&0\\
0&2&0&2&2&2&0&0&2&0&0&2&0&0&0&4\\
0&0&2&2&0&0&0&0&2&4&2&0&2&0&0&2\\
0&2&4&0&0&2&2&2&0&0&2&0&0&0&0&2
\end{vmatrix}
\times
\begin{vmatrix}
16&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&2&2&0&4&0&2&2&2&0&2&0\\
0&0&0&2&2&2&2&0&0&2&4&0&0&2&0&0\\
0&2&2&2&0&0&2&0&0&0&0&2&4&2&0&0\\
0&0&2&2&2&4&0&2&0&0&0&0&2&0&2&0\\
0&0&2&0&2&0&0&0&0&0&2&4&0&2&2&2\\
0&0&0&2&0&0&4&2&2&0&0&0&0&2&2&2\\
0&2&2&0&2&0&2&0&2&2&0&0&0&0&4&0\\
0&0&0&0&4&0&2&2&0&2&0&2&2&0&0&2\\
0&4&0&0&2&0&0&2&2&0&2&0&2&2&0&0\\
0&2&0&0&0&2&0&0&0&2&0&0&2&4&2&2\\
0&0&2&0&0&2&0&4&2&2&0&2&0&2&0&0\\
0&2&0&4&0&0&0&2&0&2&2&2&0&0&2&0\\
0&2&0&2&2&2&0&0&2&0&0&2&0&0&0&4\\
0&0&2&2&0&0&0&0&2&4&2&0&2&0&0&2\\
0&2&4&0&0&2&2&2&0&0&2&0&0&0&0&2
\end{vmatrix}
\neq$$

$$\neq
\begin{vmatrix}
16&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&2&0&0&0&0&4&2&2&0&4&0&0&0&2&0\\
0&2&0&0&0&0&0&2&2&0&0&0&2&6&0&2\\
0&0&0&0&0&2&2&4&0&2&2&4&0&0&0&0\\
0&0&4&4&0&0&0&0&2&0&0&2&2&0&0&2\\
0&0&2&2&0&6&2&0&2&0&2&0&0&0&0&0\\
0&0&2&2&2&2&0&0&0&0&0&0&0&0&4&4\\
0&0&0&0&2&2&0&0&4&2&0&2&0&2&2&0\\
0&2&0&4&2&0&0&0&0&2&0&0&0&2&2&2\\
0&0&0&2&0&0&0&2&2&6&2&0&0&0&0&2\\
0&2&2&2&0&2&0&0&0&2&0&0&2&0&4&0\\
0&0&2&0&6&0&2&2&2&0&0&0&0&0&0&2\\
0&2&2&0&0&0&0&0&0&0&2&2&4&2&0&2\\
0&2&0&0&2&2&2&0&0&0&2&4&2&0&0&0\\
0&4&2&0&0&0&0&2&0&0&2&0&2&2&2&0\\
0&0&0&0&2&0&4&2&0&2&0&2&2&2&0&0
\end{vmatrix}$$

Непосредственное выполнение расчётов показывает, что результат матричного произведения не совпадает с дифференциальной таблицей произведения подстановок (она представлена второй частью неравенства).

Таким образом, *произведение* (*последовательность*) *подстановочных преобразований нетривиального типа* (*а не только шифров*) *является с большой вероятностью случайной подстановкой, независимо от свойств подстановки, участвующей в формировании этого степенного преобразования*.

Мы посчитали, что это и приведенное выше утверждение является неким ″законом природы″, который выполняется независимо от нашего

желания (может здесь надо более строго оговорить, какие подстановки удовлетворяют этому правилу, но это предмет отдельного исследования).

Аналогично к стационарному распределению свойственному случайной подстановке приходит и любой шифр. Стационарное распределение как раз соответствует тому, что шифр начинает повторять свойства случайной подстановки.

А вот тот факт, что последовательность шифрующих преобразований с нулевыми цикловыми подключами, асимптотически становится случайной подстановкой, оказался всё же неожиданным. Объяснением этому факту может быть лишь то, что сами по себе подстановки (исключая тривиальные их конструкции), как правило, представляют собой набор случайных переходов (уже в самой подстановке заложен механизм случайного перемешивания).

Теперь можно привести дополнительные результаты, уточняющие ряд поднятых ранее вопросов.

О гипотезе статистической эквивалентности.

Как следует из приведенных выше и многочисленных других имеющихся результатов, стационарное состояние, к которому приходит шифр (Марковский шифр) практически не зависит от ключа зашифрования, что хорошо видно из таблиц 4.5-4.8. Это говорит о том, что гипотеза статистической эквивалентности, о которой говорилось в предыдущем подразделе, оказывается справедливой для всех Марковских шифров как первого, так и второго порядков.

Гипотеза статистической эквивалентности в нашей интерпретации выглядит следующим образом:

Гипотеза статистической эквивалентности. *Для Марковских шифров значение максимума полных дифференциалов* (*также как и значение максимума смещений линейных корпусов*) *почти для всех ключей является одним и тем же, практически совпадающим с максимумом таблицы XOR*

переходов (*максимумом смещения таблицы линейных аппроксимаций*) *случайной подстановки соответствующей степени.*

Следующее наше уточнение связано с оценкой показателей стационарности для Марковских шифров.

Возвращаясь к теореме 3 работы [74] мы хотим обратить внимание на то, что по имеющимся многочисленным результатам экспериментов утверждение о том, что с увеличением числа циклов шифр приходит к равномерному стационарному распределению, встречающееся и в ряде других работ, является не верным. На самом деле каждый из известных итеративных шифров после нескольких начальных циклов приходит к стационарному распределению, повторяющему законы распределения (дифференциалов и смещений таблиц линейных аппроксимаций) случайной подстановки соответствующей степени и дальнейшее наращивание числа циклов этого распределения (далеко не равномерного) не изменяет. В нашей работе [204] приводится теоретическое доказательство этого факта. Поэтому утверждение теоремы 3 работы [74] во-первых, противоречит вроде бы де признанному факту о приходе Марковского шифра к стационарному состоянию, а во вторых, − в соответствии с приведенными выше и другими имеющимися результатами стационарное состояние шифра соответствует характеристикам случайной подстановки, а для случайной подстановки законы распределения переходов XOR таблиц и смещений таблиц линейных аппроксимаций являются явно не равномерными.

Таким образом, заключая этот подраздел, можно отметить главный его результат, состоящий в том, что как установлено все итеративные шифры (а они все Марковские) асимптотически имеют значения максимальных значений дифференциалов и линейных корпусов совпадающие со значениями максимумов XOR таблиц и смещений таблиц линейных аппроксимаций случайных подстановок соответствующей степени.

Стационарность распределения переходов таблицы полных дифференциалов также как стационарность смещений линейных корпусов

обозначает, что с увеличением числа циклов шифрования эти распределения сохраняются (не меняются).

Таким образом, к выводам по этой части можно отнести такие:

1. Имеющиеся в литературе подходы к определению Марковских шифров представляются не совсем аккуратными.

2. Предложен уточнённый подход к определению Марковских шифров, который строится на основе строгого математического определения Марковского случайного процесса *k*-того порядка.

3. Показано, что в соответствии с введенным определением практически любой итеративный шифр является Марковским, в частности SPN шифры формируют в результате зашифрования Марковские процессы первого порядка, в то время как шифры, построенные с использованием Фестель подобных схем формирования цикловых функций, создают в результате зашифрования Марковские процессы второго порядка.

4. Для каждого итеративного (Марковского) шифра существует определённое (небольшое) число начальных циклов шифрования, после которого законы распределения переходов XOR таблиц (дифференциалов) и смещений таблиц линейных аппроксимаций (линейных корпусов) шифра приходят к установившемуся (стационарному) значению. На первых шагах шифрования Марковскому шифру присущ переходный период. Шифр приходит к стационарному процессу (состоянию) асимптотически.

4.11. Расчетные соотношения для определения максимальных значений полных дифференциалов и линейных корпусов.

Здесь мы хотим подтвердить возможность получения показателей доказуемой стойкости итеративных блочных симметричных шифров расчётным путём.

Расчетные соотношения для определения максимальных значений полных дифференциалов и максимальных значений линейных корпусов могут быть получены применением законов распределения (3.2) и (3.8), полученных для случайных подстановок, непосредственно к шифрам,

рассматриваемым как случайные подстановки, что и сделано в работах [43] и [44].

Как показано выше, среднее значение максимума таблицы XOR разностей случайной подстановки степени 2^n находится путем определения максимального значения $k = k_D^*$, при котором выполняется соотношение

$$\frac{(2^n - 1)}{2^n!} \cdot \begin{pmatrix} 2^{n-1} \\ k_D^* \end{pmatrix}^2 \cdot k_D^*! 2^{k_D^*} \cdot \Phi(2^{n-1} - k_D^*) \approx 1. \qquad (4.28)$$

Уместно здесь будет обратиться к расчетам, представленным в табл. 3.7, выполненным в соответствии с соотношением (4.28) для случайных подстановок степени 2^n. Отмечено, что, к сожалению, для больших значений степени подстановки ($n \geq 32$) выполнить расчёты по формуле (4.28) не удаётся ввиду получения чисел, не укладывающихся в сетку компьютера. Поэтому актуальной становится задача поиска подходящих приближений. Одно из них предлагается в четвёртой колонке приведенной табл. 3.7.

Здесь представляются результаты расчетов по упрощенной формуле, полученной на основе простого подбора.

Очевидно, что для интересующего нас значения максимальной дифференциальной вероятности (максимальной вероятности полного дифференциала) $DP_{\max}^f$ можем записать выражение ($k_{\max} = 2k_D^*$):

$$DP_{\max}^f = \frac{2k_D^*}{2^n} \qquad (4.29)$$

В соответствии с предлагаемым упрощенным расчетным соотношением, приходим к результату:

$$DP_{\max}^f = \frac{n+4}{2^n}. \qquad (4.30)$$

Следует отметить, что к настоящему моменту удалось найти более поздние работы, приближающиеся по результатам, представленным в табл.

4.1. Имеются в виду две работы [205], [206]. В этих работах, однако, уточняется верхняя граница для среднего значения максимума дифференциальной таблицы, полученная в уж упомянутой работе 207], а также в работе [173], в которых показано, что при случайном равновероятном выборе подстановки $s \in S_n$, $n = 2^m$ среднее значение матрицы (дифференциальной таблицы) при достаточно больших m удовлетворяет неравенству:

$$EP^*(m) \le m/2^{m-1}.$$

Конечно, наш результат удовлетворяет этому ограничению, однако, оно для нас слишком грубое.

Другим удобным приближением может стать доказанное в подразделе выражение для закона вероятностей переходов XOR таблицы случайной подстановки в виде пуассоновского закона распределения вероятностей:

$$\Lambda_{n,2k} = (2^n - 1)^2 \cdot \frac{e^{-1/2}}{2^k k!}.$$

Для n = 32 из этой формулы следует при k = 17 $\Lambda_{n,2k}$ = 0,23; а для n = 32 и k = 16 получается $\Lambda_{n,2k}$ = 8,47, и, следовательно, $2k$ = 34 можно считать хорошим приближением.

Соответственно для n = 128 упрощенная формула даёт результат $DP^f_{\max} = \frac{n+4}{2^n} = 2^{-121}$, в то время как использование пуассоновского закона даёт такое практически же значение: $DP^f_{\max} = \frac{98}{2^{128}} = 2^{-121}$.

Можно сделать вывод, что упрощенная формула повторяет результат, следующий из пуассоновского закона, т.е. ею вполне можно пользоваться при выполнении практических расчётов.

Рассмотрим, как обстоят дела с расчётами линейных вероятностей?

В разделе 3. показано, что среднее значение максимума таблицы линейных аппроксимаций для случайной подстановки определяется

аналогично предыдущему случаю путем нахождения значения k_L^* являющегося целым решением (округлением в сторону ближайшего целого) уравнения

$$\frac{(2^n-1)^2 \cdot (2^{n-1}!)^2}{2^n!} \cdot \left(\frac{2^{n-1}}{2^{n-2} + \left| k_L^* \right|} \right)^2 \approx 1. \qquad (4.31)$$

Результаты расчетов по формуле (4.31) представлены в табл. 3.7).

Напомним, что для шифра с *n*-битовым размером входа максимальное значение линейной вероятности (максимальной вероятности линейного корпуса) $DL_{\max}^f$ представляется в виде ($k_{\max} = 2k_L^*$):

$$LP_{\max}^f = \left(\frac{2k_L^*}{2^{n-1}} \right)^2. \qquad (4.32)$$

Для линейной аппроксимационной таблицы соотношение (4.31) для расчёта $k_{\max} = 2k_L^*$ при больших значениях $n > 20$ оказывается трудным для вычислений. Приведем здесь оценочное соотношение, предложенное на основе обработки результатов вычислительных экспериментов в [44], являющееся удобной заменой выполнению расчетов по соотношению (4.31)

$$LP_{\max}^f \le \left(\frac{\left(\frac{3}{2} \right)^n}{2^{n-1}} \right)^2. \qquad (4.33)$$

Как показывает, однако, анализ это расчётное соотношение хорошо работает для значений $n \le 24$. Для получения практических результатов при больших значениях *n* воспользуемся теоремой 9 из работы [178], в которой предлагается аппроксимация соотношения (3.8) нормальным законом распределения вероятностей.

Теорема. *Для случайной n-битовой подстановки, с* $n \ge 5$ *дисбаланс* Imb(*v*,*u*) *аппроксимации является случайным значением с распределением, которое может быть аппроксимировано в виде*

$$\Pr(\mathrm{Imb}(v, u) = z) \approx 2Z\left(\frac{z}{2^{(n-2)/2}}\right) \qquad (4.34)$$

для z четного и ноль для z нечетного.

При $z = 2x$ (4.34) представляется в виде $\Pr(\mathrm{Imb}(v, u) = 2x) \approx Z\left(\frac{x}{2^{(n-4)/2}}\right)$.

При выводе этого соотношения авторы пользуются леммой, повторяющей наш результат (3.8):

$$\Pr(\mathrm{Imb}(v, u) = 2x) = \frac{\binom{2^{n-1}}{2^{n-2}+x}^2}{\binom{2^n}{2^{n-1}}}, \qquad (4.35)$$

т.е. в наших обозначениях дисбаланс $\mathrm{Imb}(v, u) = z$ при $z = 2k$ как раз соответствует $\lambda(\pi) = 2k$.

В табл. 4.10 приводятся для сравнения результаты оценки максимального значения дифференциальной таблицы случайной подстановки (половинного значения) полученные при использовании аппроксимирующих выражений (4.33), аппроксимации, использованной в выражении (4.34) и точного расчёта по формуле (4.31) с учётом (3.8).

Как следует из представленных результатов счёта, итоговая аппроксимация в виде нормального закона (4.34), оказывается не такой уж и плохой. К сожалению, выполнить расчёты по точной формуле удаётся только для значений n не превышающих 32, причём если аппроксимация нормальным законом даёт оценки заниженные, то наша аппроксимация приводит к завышенным оценкам.

Для значений $n > 32$ остаётся ориентироваться на аппроксимирующие соотношения. Если полагать, что соотношение граничных (оценочных) значений, следующих из соответствующих аппроксимирующих выражений, с ис-

Таблица 4.10

Сравнение результатов, полученных различными путями

n	Значение $x = k_L^*$ для нормального закона	Значение x для нашей аппроксимации	Расчётное значение k_L^*
8	16	12,81	16-17
16	334	328,42	374
20	1466	1662,62	1670
24	6342	8417	7302
28	27142	42611,346	31504
32	115080	215719	135649
128	$\frac{62316975567822669939}{\approx 2^{67} \to \approx 2^{-120}}$	$\frac{1732411920785470223756 0}{\approx 2^{75} \to \approx 2^{-104}}$	-

тинными значениями сохраняется и для значений $n > 32$, то можно перейти к расчётам ожидаемого значения максимума линейной вероятности для 128-битного шифра Rijndael.

Использование представленных аппроксимирующих соотношений для 128-ми битного шифра позволяет получить граничные значения (слева и справа) такого вида

$$2^{-120} \le LD_{\max}^{f} \le 2^{-104}.$$

Заметим, однако, что аппроксимация нормальным законом для значений $n \ge 24$ получается существенно точнее нашей аппроксимации (судя по приведенным данным, отношение расчётного и аппроксимирующего выражений для аппроксимации в виде нормального закона принимает значения: при $n = 20 \to \frac{1670}{1466} = 1{,}139$ при $n = 24 \to \frac{7302}{6384} = 1{,}144$, при $n = 32 \to \frac{135649}{115080} = 1{,}178$). Если считать, что близкое к этому соотношение (отношение меньше двойки) сохранится и для больших значений n, то можно прийти к ожидаемому значению вероятности более близкому к левой из двух приведенных границ, т.е. в качестве достаточно точной оценки линейной

вероятности для интересующего нас битового размера входа в шифр $n = 128$ следует рассматривать значение $LD^{f}_{\max} \approx 2^{-119}$. Предложенная нами аппроксимация в этом случае даёт ошибку в 2^{15} раза. Она оказывается хорошей для малых версий шифров. Работа по уточнению результатов продолжается.

В итоге получены значения вероятностей максимумов дифференциалов и смещений линейных корпусов для шифра Rijndael (и других 128-битных шифров, в частности шифров, представленных на Украинский конкурс), которые можно рассматривать как показатели доказуемой безопасности этих шифров.

В заключение можно отметить, что если применить методику определения MADP MALHP, предлагаемую в публикациях, к уменьшенным моделям, то мы получим для полубайтового S-блока с $p = 2^{-2}$ (в этой модели коэффициент ветвления равен 4 и имеет один уровень):

$$MADP^{\mathrm{R}_{16}} \le 1^{-2} \overline{4\times1} = 2^{-8},$$

в то время как на самом деле (реальный результат) есть

$$. \; AMDP^{\mathrm{R}_{16}} = \frac{20}{2^{16}} = 2^{-12}$$

Конечно же $2^{-12} < 2^{-8}$, но ошибка здесь для малой версии шифра уже превышает порядок.

Реальный результат для линейных показателей

$$AMLHP^{\mathrm{R}_{16}} = \left(\frac{820}{2^{15}}\right)^2 = 2^{-10,64}.$$

Подводя итоги материалам раздела по приведенным результатам и обоснованиям можно утверждать, что:

1. Современные блочные симметричные шифры (при полном наборе шифрующих многоцикловых преобразований) обладают свойствами случайных подстановок и для них справедливы законы распределения вероятностей для полных дифференциалов и линейных корпусов

свойственные таблицам XOR разностей и линейных аппроксимаций подстановок соответствующей степени (степени) (3.2) и (3.8). Объективные данные, подтверждающие это утверждение, будут представлены в следующем разделе.

2. Максимальные значения полных дифференциалов и линейных корпусов для современных БСШ, определяющие по современным меркам показатели стойкости шифров к атака дифференциального и линейного криптоанализа, могут быть получены расчетным путем. Они не зависят (при достаточном числе цикловых преобразований ни от свойств используемых в шифрах подстановочных конструкций, ни от методов введения в цикловые функции цикловых подключей, ни от способа построения расширяющего линейного преобразования цикловой функции, а являются функцией только размера битового входа в шифр (степени подстановки).

3. Для оценки максимальных значений полных дифференциалов блочных симметричных шифров с битовым размером входа равным n к атакам дифференциального криптоанализа можно пользоваться соотношением (4.30), являющимся хорошей аппроксимацией расчётов по точному выражению (4.28). К сожалению, при оценке линейных свойств подстановок сложность вычислений по полученному расчётному соотношению (3.8) с ростом степени подстановки ($n > 24$) становится непреодолимой. Здесь остаётся довольствоваться представленными в работе аппроксимирующими соотношениями (4.33), (4.34).

4. Общим результатом раздела является обоснование новой системы взглядов (новой идеологии) оценки стойкости блочных симметричных шифров к атакам дифференциального и линейного криптоанализа. Сущность предлагаемой новой методологии состоит в признании того факта, что современные блочные симметричные шифры (большинство известных шифров) асимптотически (при полном наборе цикловых преобразований) приобретают свойства случайных подстановок соответствующей степени. Поэтому максимумы полных дифференциалов и смещений таблиц линейных

аппроксимаций (линейных корпусов) современных блочных симметричных шифров, определяющие по современным меркам их показатели стойкости, для многих из них однозначно определяются значениями максимумов переходов таблиц XOR разностей и смещений таблиц линейных аппроксимаций случайных подстановок соответствующей степени, и, следовательно, эти показатели могут быть получены расчетным путём.

Одним из важных компонентов развиваемой методики является признание также факта, что большинство современных блочных симметричных шифров допускают построение уменьшенных версий (моделей), в которых удаётся сохранить или отразить все основные криптографические преобразования прототипа. Применение развиваемого подхода позволяет выполнить проверку соответствия показателей стойкости рассматриваемых шифров к атакам линейного и дифференциального криптоанализа на основе анализа показателей случайности уменьшенных версий этих преобразований, что позволяет существенно сократить временные и интеллектуальные затраты на проведение экспертизы и принятие решений о соответствии шифров заданным требованиям, – решений, обладающих по сравнению с известными подходами, существенно более высоким уровнем объективности и соответственно доверия.

Здесь мы констатируем конечные результаты, привязываясь к полному набору цикловых преобразований шифра (речь идёт о максимально достижимой линейной или дифференциальной вероятности, определяющей цену атаки на весь шифр), которая фиксируется при полным наборе цикловых преобразований. Связь максимальных значений дифференциальных вероятностей с числом циклов просматривается в представленных результатах, однако она не является здесь предметом изучения.

В рамках этой работы для оценки показателей доказуемой стойкости шифров к атакам линейного и дифференциального криптоанализа использовались малые модели. Представляется, что развиваемый подход

может быть с успехом применён и для оценки показателей стойкости шифров и ко многим другим атакам.

6 МЕТОДЫ И МОДЕЛИ ОЦЕНКИ РИСКОВ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ИНФОРМАЦИОННО-ТЕЛЕКОММУНИКАЦИОННЫХ СИСТЕМ

Управление различными технологическими процессами базируется на использовании информационно-телекоммуникационных систем (ИТС), к которым относятся источники информации, средства ее передачи, обработки, отображения, хранения, общее и специальное программное обеспечение. Во всех информационных технологических процессах, а также процессах управления, важную роль играют люди, а именно человеческий фактор.

Человек принимает непосредственное участие в разработке, производстве и эксплуатации и ремонте ИТС. Технологический процесс современных организаций в данный момент времени невозможен без участия человека, за которым остается наиболее ответственный процесс принятия решений.

Поскольку человек участвует на всех стадиях существования и использования ИТС, то возникает проблема причастности непосредственно как к обеспечению защиты информации, так и с другой стороны к возможным угрозам функционированию таких систем. Информация, используемая в современных системах, нуждается в защите в связи с влиянием значительного количества существующих дестабилизирующих факторов (ДФ). Эти факторы влияют на качество ИТС на этапах их разработки, производства, испытаний, использования по назначению и т.п.

В работе *разработан и предложен* унифицированный метод оценки рисков информационной безопасности, который удовлетворяет международным стандартам в области управления информационной безопасностью. Этапы предложенного метода оценки рисков информационной безопасности хорошо структурированы и, в совокупности,

представляют систематизированный метод оценки рисков, расширяющий известные на данный момент методы и методики оценки.

Методы исследования опираются на положениях теории принятия решений, нечетких множеств, графов и метода анализа иерархий.

Теория принятия решений — область исследования, вовлекающая понятия и методы математики, статистики, экономики, менеджмента и психологии с целью изучения закономерностей выбора людьми путей решения разного рода задач, а также способов поиска наиболее выгодных из возможных решений.

Вопрос об оценке рисков информационной безопасности (ИБ) становится все более актуальным. Сегодня построение и эффективная эксплуатация систем ИБ невозможны без анализа и управления рисками ИБ такой системы. Основной задачей данного направления является объективная идентификация угроз и оценка наиболее значимых информационных рисков для организации, а также адекватность используемых средств контроля рисков в целях повышения эффективности и рентабельности предприятия.

Анализ рисков ИБ – систематическое использование информации для идентификации источников и оценки величины риска. Анализ рисков служит основой для оценивания рисков, обработки и принятия рисков.

Оценивание риска ИБ – процесс сравнения оценочной величины риска с установленными критериями риска с целью определения уровня значимости риска

Основными задачами оценки рисков информационной безопасности являются:

1) выбор необходимых требований и средств защиты от угроз ИБ;

2) принятие своевременных решений относительно применения средств защиты;

3) правильная расстановка приоритетов в выборе механизмов обеспечения ИБ;

4) оценка экономической целесообразности и эффективности мер, обеспечивающих ИБ;

5) избежание кризисных ситуаций, которые могут иметь негативное влияние на функционирование информационно-телекоммуникационной системы.

6.1 Анализ проблемы управления рисками информационной безопасности

Информационные активы являются объектом для многих видов угроз. Угроза может стать причиной нежелательного инцидента, в результате которого предприятию или организации будет причинен ущерб. Этот ущерб может возникнуть в результате атаки на программные и аппаратные ресурсы ИТС, что приведет к несанкционированному раскрытию, модификации, повреждению, уничтожению информации в ИТС. Угрозы ИБ могут быть осуществлены посредством использования уязвимостей системы.

Уязвимости представляют собой слабости защиты, ассоциированные с информационными активами предприятия или организации. Эти слабости могут использоваться одной или несколькими угрозами, являющимися причиной нежелательных инцидентов, которые могут стать причиной нестабильного функционирования компонентов ИТС. Уязвимости – это любые факторы, делающие возможной успешную реализацию угроз. Можно с уверенностью констатировать, что уязвимости являются основной причиной возникновения атак. Наличие же слабых мест в ИТС может быть обусловлено самыми различными факторами, начиная с простой халатности сотрудников и заканчивая преднамеренными действиями злоумышленников.

Управление рисками ИБ как научная и управленческая деятельность представляет собой совокупность последовательных этапов научно-практических исследований, направленных на определение достоверных и

обоснованных характеристик риска, а также на выявление эффективных мер по его сокращению.

Анализ существующих подходов к проблеме управления рисками сложных систем показывает, что этот вопрос в большей степени является открытым, так как сама эта проблемная область еще плохо формализована и изучена. Модели этой проблемной области очень не точны, дают, как правило, качественные оценки, достоверность которых не всегда очевидна. Это связано со сложностью самой проблемы и с зависимостью ее от чисто субъективных факторов. Для описания таких моделей используется различный математический аппарат: методы субъективной вероятности, нечеткие множества, нейронные сети и т.д. Подобные модели являются средством уменьшения степени неопределенности при выборе возможных вариантов решений задач управления рисками.

При управлении рисками в сложных системах у экспертов всегда возникают трудности при анализе рисков, а если их число велико и они плохо структурированы, то линейное ранжирование рисков и выбор среди них наиболее значимого превращается в сложную задачу. Лимит времени, отводимого на аудит информационной безопасности корпоративных систем, необходимость учета нечисловых характеристик системы защиты информации требуют разработки новых подходов к ее решению. Возможный выход из создавшегося положения заключается в использовании в качестве основной процедуры сравнения и оценки сценариев метода парных сравнений.

В работе предлагается модель управления рисками ИБ (рис. 1), которая удовлетворяет международным стандартам в области управления информационной безопасности, а также учитывает особенности построения и функционирования ИТС.

В соответствии с международными стандартами управление рисками ИБ предполагает следующее:

1) определение основных целей и задач защиты информационных активов;

2) создание эффективной системы оценки и управления рисками ИБ.

Предложенная модель управления рисками ИБ включает в себя следующие этапы и модели:

1) оценка рисков информационной безопасности;

2) модель атак на ИТС;

3) анализ угроз информационной безопасности в ИТС;

4) модель оптимального выбора комплекса средств защиты;

5) модель выбора оптимальной ИТС.

Рис. 6.1 – Унифицированная модель управления рисками ИБ

6.2 Математическая модель оценки рисков информационной безопасности

Атаки на ИТС могут быть подразделены на внешние и внутренние. Внешние сетевые атаки проводятся извне с узлов, которые не входят в состав ИТС. Внутренние атаки проводятся с одного из компонентов ИТС, например.

Потенциальными целями злоумышленников могут выступать рабочие станции, серверы, коммуникационное оборудование, а также каналы связи ИТС. При этом если атака проводится по сети, то она может быть соотнесена с одним из пяти уровней модели ВОС.

Атаки могут носить однонаправленный или распределенный характер. Распределенные атаки в отличие от однонаправленных проводятся одновременно из нескольких источников. Примером таких атак служат распределенные атаки типа «отказ в обслуживании», которые реализуются путем формирования и одновременной посылки из нескольких источников большого числа паке заданных узлам, являющимся объектами атаки.

Для наглядности модели управления рисками информационной безопасности (рис. 1) необходимо построить математическую модель атак на ИТС. Построение такой модели является одним из этапов управления рисками ИБ.

Проведенные исследования существующих типов моделей атак на ИТС позволили констатировать, что созданные в настоящее время модели могут быть классифицированы по следующим базовым критериям:

1) степень формализуемости модели;

2) тип представления модели;

3) возможность расширения модели;

4) возможность учета в модели последовательности действий, совершаемых нарушителем в процессе проведения информационной атаки;

5) уровень детализации модели.

Необходимо отметить, что для эффективного использования моделей атак для исследования возможных действий нарушителя по отношению к ИТС они должны обладать следующими основными свойствами:

1) универсальность: модель может быть использована для представления различных типов атак вне зависимости от источника, объекта и средства реализации атаки;

2) гибкость: обеспечение возможности добавления в модель новых характеристик атаки. Это свойство позволяет пользователю изменять состав характеристик моделируемых атак в зависимости от среды ИТС, в которой они рассматриваются;

3) формализуемость: свойство, которое указывает на возможность использования математического аппарата для описания параметров модели;

4) простота: дает возможность пользователю легко воспринимать структуру и способы реализации моделируемой атаки;

5) многофакторность: позволяет учитывать три основных параметра моделируемой информационной атаки: уязвимость, активизируемая атакой, способ реализации атаки и ее возможные последствия.

Рассмотрим обобщенную математическую модель атаки на ИТС, которая базируется на следующих трех основных множествах: V – множество уязвимостей ИТС, A – множество методов реализации атак и C – множество последствий атак. Для описания взаимосвязи между элементами множества *A, V и C,* определим отношение W на множестве $U=A\times V\times C$. Принадлежность элемента (a, v, c) отношению W, где $a \in A$, $v \in V$, $c \in C$, интерпретируется следующим образом: «Атака на ИТС, реализуемая нарушителем методом a путем активизации уязвимости v и приводящая к последствию c ».

С каждой уязвимостью $v_i \in V$ связано множество A_i , являющееся подмножеством множества А и включающее атаки на ИТС, направленные на активизацию уязвимости v_i . При этом $0<|A_i|<|A|$, т.е. одна уязвимость v_i не может быть активизирована для реализации всех атак из множества A.

Вместе с тем не существует такой уязвимости, на основе которой не могло быть реализовано ни одной атаки на ИТС.

С каждой атакой на ИТС $a_j \in A$ связано множество V_j , являющееся подмножеством V и включающее уязвимости, активизируемый атакой a_j.. При этом $0<|V_i|<|V|$, т.е. одна атака a_j не может активизировать одновременно все уязвимости ИТС, и, наоборот, не существует такой атаки, которая бы не активизировала ни одной уязвимости ИТС.

С каждой атакой $a_j \in A$ связано множество C_j, являющееся подмножеством множества C и включающее последствия атаки a_j. При этом $0<|C_i|<|C|$, т.е. атака a_j не может привести одновременно ко всем последствиям, входящим во множество *C, и,* вместе с тем, атака a_j не может не иметь ни одного последствия.

С каждым последствием реализации атаки на ИТС $c_k \in C$ связано множество A_k, являющееся подмножеством множества A и включающее атаки, приводящие к последствию c_k. При этом $0<|A_k|<|A|$, т.е. не существует такого последствия, к которому бы не привело ни одной атаки, и в то же самое время, последствие не может быть следствием реализации всех атак, входящих в множество *A.*

6.3 Анализ защищаемых информационных ресурсов

Информационные ресурсы можно определить как весь имеющийся объем информации в рассматриваемой информационно-телекоммуникационной системе.

В настоящее время ИТ достигли такого уровня развития, когда объемы информации и уровень ее сложности потребовали создания информационной индустрии. Наличие информации определяет развитие организации. Информация стала стратегическим ресурсом, а информационные ресурсы являются одними из важнейших.

На этапе анализа рисков ИБ (рис. 1) – анализ защищаемых информационных ресурсов определяют множество информационных ресурсов ИТС– *D,* которые должны быть защищены от возможных атак нарушителей. Элементами этого множества могут являться: файловые ресурсы; служебные данные, хранящиеся в СУБД; пользовательские документы и др. Выбор защищаемых ресурсов должен осуществляться исходя из состава той информации, которая необходима для выполнения функциональных задач ИТС. В множество *D* также должны входить информационные ресурсы, защита которых должна быть обеспечена в соответствии в отечественной правовой базой. Одноэтапные информационные ресурсы могут объединяться в рамках одного элемента множества *D.*

Анализ защищаемых информационных ресурсов (рис.2) должен соответствовать определенным условиям, которые предполагают современные стандарты качества управления. Суть данных требований заключается в следующем:

1) адаптивность – использование анализа информационных ресурсов для любых компонентов ИТС;

2) завершенность – анализируемые показатели не нуждаются в измерении дополнительных параметров;

3) уникальность – измеряемые показатели должны быть единственными и отвечать определенным требованиям;

4) точность – оцениваемые показатели должны точно соответствовать фактическому состоянию анализируемого объекта, погрешности оценки должны быть минимальными;

5) комплексность – рассчитываемые показатели должны отражать все стороны и специфические особенности объекта исследования;

6) гибкость – возможность использования анализа информационных ресурсов независимо от стадии жизненного цикла, состояния и развития анализируемого объекта;

7) материальность – получение количественных результатов исследования, которые могут быть использованы для улучшения функционирования анализируемого объекта;

10) однозначность – применение для различных объектов исследования, а также разными экспертами одинаковых процедур анализа, однозначная трактовка получаемых результатов оценки информационных ресурсов всеми пользователями.

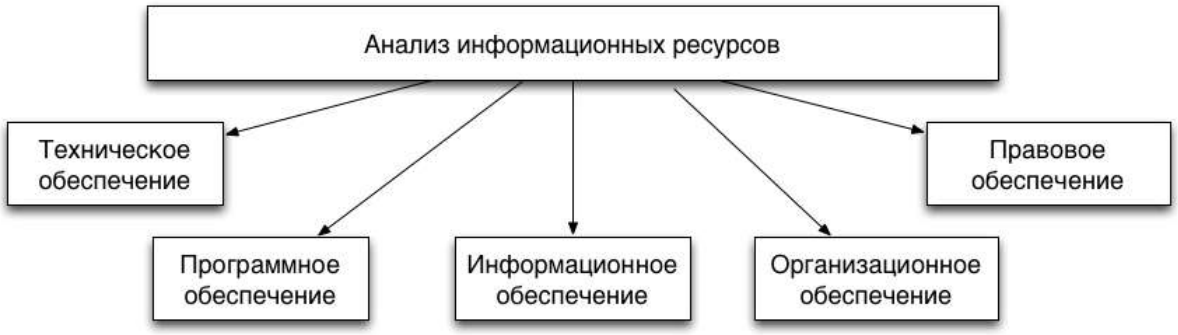


Рис. 6.2– Структура информационных ресурсов

Техническое (аппаратное) обеспечение ИТС– комплекс технических средств, устройств и процессов, позволяющих использовать и осуществлять управление информационными ресурсами.

Программное обеспечение ИТС – использование математических средств для работы с информационными ресурсами, программные продукты и техническая документация по их использованию.

Информационное обеспечение ИТС – комплекс классификации и кодирования информации, унификации систем документооборота, схемы движения информации и методология построения базы данных.

Организационное обеспечение ИТС – приемы и методы взаимодействия между сотрудниками и техническими средствами по вопросу управления информационными ресурсами.

Правовое обеспечение ИТС – совокупность нормативно- правовых актов, регулирующих вопросы возникновения, перемещения и эксплуатации информационных ресурсов предприятия.

Для проведения эффективной оценки рисков ИБ необходимо детально проанализировать аппаратное и программное обеспечение.

6.4 Анализ программного и аппаратного обеспечения

На этапе анализа программного и аппаратного обеспечения метода оценки рисков ИБ (рис. 1) формируются два множества: *S* – множество программного обеспечения и *H* – множество аппаратного обеспечения, которое используется в ИТС для хранения и обработки информационных ресурсов, входящих в множество *D* (рис. 3).

Рис. 6.3 – Анализ множеств и их компонентов в методе оценки рисков ИБ

Для того чтобы задать взаимосвязь между аппаратным обеспечением ИТС и установленным на нем программным обеспечением, вводится бинарное отношение B_1, определенное на множестве $U_1 = S \times H$. При этом принадлежность элемента (s,h) отношению B_1, где $s \in S$, $h \in H$, интерпретируется следующим образом: «программное обеспечение s, установленное на аппаратном обеспечении h». Далее определяется взаимосвязь между элементами множеств D, S и H при помощи тернарного отношения B_2, заданного на множестве $U_2 = D \times S \times H$. Принадлежность элемента (d, s, h) отношению B_2, где $d \in D$, $s \in S$, $h \in H$, интерпретируется следующим образом: «Информационный ресурс d, обрабатываемый

средствами программного обеспечения s, которое установлено на аппаратном обеспечении *h*». В случае, если информационный ресурс обрабатывается только аппаратным обеспечением, например, коммутатором, взаимосвязь между элементами множеств *D* и *H* можно определить через бинарное отношение B_3, заданное на множестве $U_3 = D \times H$. На основе отношений B_1 , B_2 , B_3 формируется таблица, состоящая из следующих полей: порядковый номер элемента таблицы, наименование аппаратного обеспечения, перечень установленного на нем программного обеспечения и список информационных ресурсов, которые хранятся или обрабатываются программно-аппаратным обеспечением ИТС. Необходимо отметить, что в некоторых случаях на аппаратном обеспечении может отсутствовать программное обеспечение, если оно интегрировано в аппаратные компоненты устройства. Это характерно, например, для некоторых коммутаторов и маршрутизаторов, используемых в ИТС, логика работы которых реализована на аппаратном уровне в виде специализированных микросхем.

6.5 Анализ информационных потоков

На этапе анализа информационных потоков определяется множество категорий пользователей ИТС— *U*, а также права доступа этих пользователей к информационным ресурсам множества *D* (рис. 4).

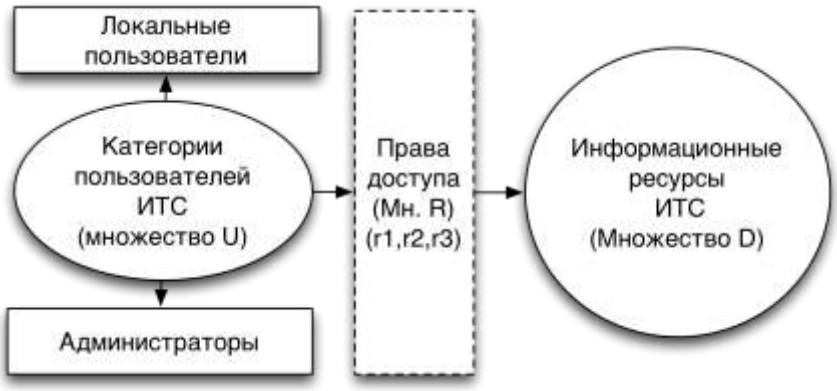


Рис. 6.4 – Связь между пользователями и информационными ресурсами ИТС

Категории пользователей формируются на основе организационной структуры предприятия, а также функциональных обязанностей его сотрудников. Возможные права пользователей по доступу к информации определяются в множестве R, которое, как минимум, включает в себя три следующих элемента: r_1 — доступ к информационному ресурсу только на чтение данных, r_2 — доступ к информационному ресурсу только на запись данных, r_3 — доступ к информационному ресурсу на чтение и на запись данных. Для того чтобы указать, какие права доступа к информации имеют категории пользователей, определяется тернарное отношение B_4, заданное на множестве $U_4 = U \times D \times R$. При этом принадлежность элемента (u, d, r) отношению B_4, где $u \in U, d \in D, r \in R$, интерпретируется следующим образом: «Категория пользователей u имеет доступ к информационному ресурсу d с правами r ». На основе отношения B_4 формируется таблица, содержащая следующие поля: порядковый номер элемента таблицы, категория пользователей ИТС, список информационных ресурсов и прав доступа к ним.

6.6 Идентификация существующих комплексов средств защиты

На данном этапе определяется множество комплексов средств защиты *Z*, которые используются в ИТС. В множество *Z* включаются как организационные меры, так и технические средства защиты. После этого формируется бинарное отношение B_5, определенное на множестве $U_5 = Z \times D$. Принадлежность элемента (*z*, *d*) отношению B_5, где $z \in S$, $d \in D$, интерпретируется следующим образом: «Средство *z* предназначено для защиты информационного ресурса *d*». Одно средство защиты $z \in Z$ может использоваться для зашиты одновременно нескольких информационных ресурсов, и, наоборот, один информационный ресурс может защищаться одновременно несколькими средствами защиты.

На основе отношения B_5 формируется таблица, содержащая следующие поля: порядковый номер элемента таблицы, информационные ресурсы ИТС, а также перечень средств их защиты. Взаимоотношение между элементами множеств *D, S, H* и *Z* схематично показано на рис. 5.

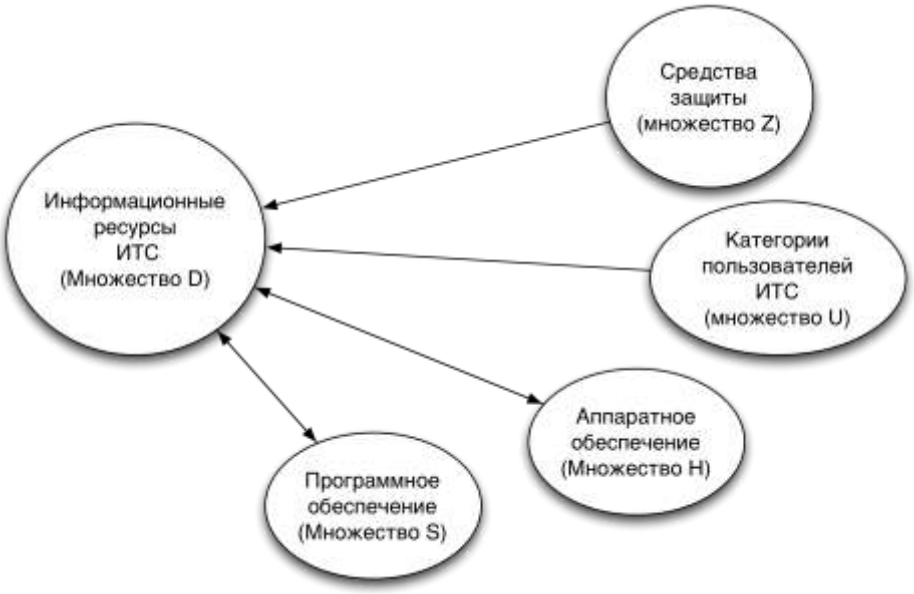


Рис. 6.5 – Схема взаимосвязи между элементами множеств *D, S, H* и *Z*

6.7 Оценка уязвимостей на основе собранных данных

Уязвимости могут присутствовать как в программно-аппаратном, так и в организационно-правовом обеспечении ИТС. Характерно, что основная часть уязвимостей организационно-правового обеспечения обусловлена отсутствием нормативных документов, качающихся вопросов информационной безопасности. В качестве примера уязвимости данного типа может служить отсутствие в организации утвержденной концепции или политики информационной безопасности, которые бы определяли требования к защите ИТС и ее компонентов, а также конкретные пути их реализации. Организационно-правовые уязвимости имеют место и в тех случаях, когда существующие нормативно-правовые документы не полностью охватывают все необходимые аспекты защиты от информационных атак или же когда такие документы существуют лишь на бумаге и не внедряются в практику.

Уязвимость программно-аппаратного обеспечения могут присутствовать в программных или аппаратных компонентах рабочих станций пользователей ИТС, серверов, а также коммуникационного оборудования и каналов связи.

Уязвимости могут существовать как на технологическом, так и на эксплуатационном этапах жизненного цикла ИТС и ее компонентов. На технологическом этапе нарушителями могут стать инженерно-технические работники, участвующие в процессе проектирования, разработки, установки и настройки программно-аппаратного обеспечении ИТС.

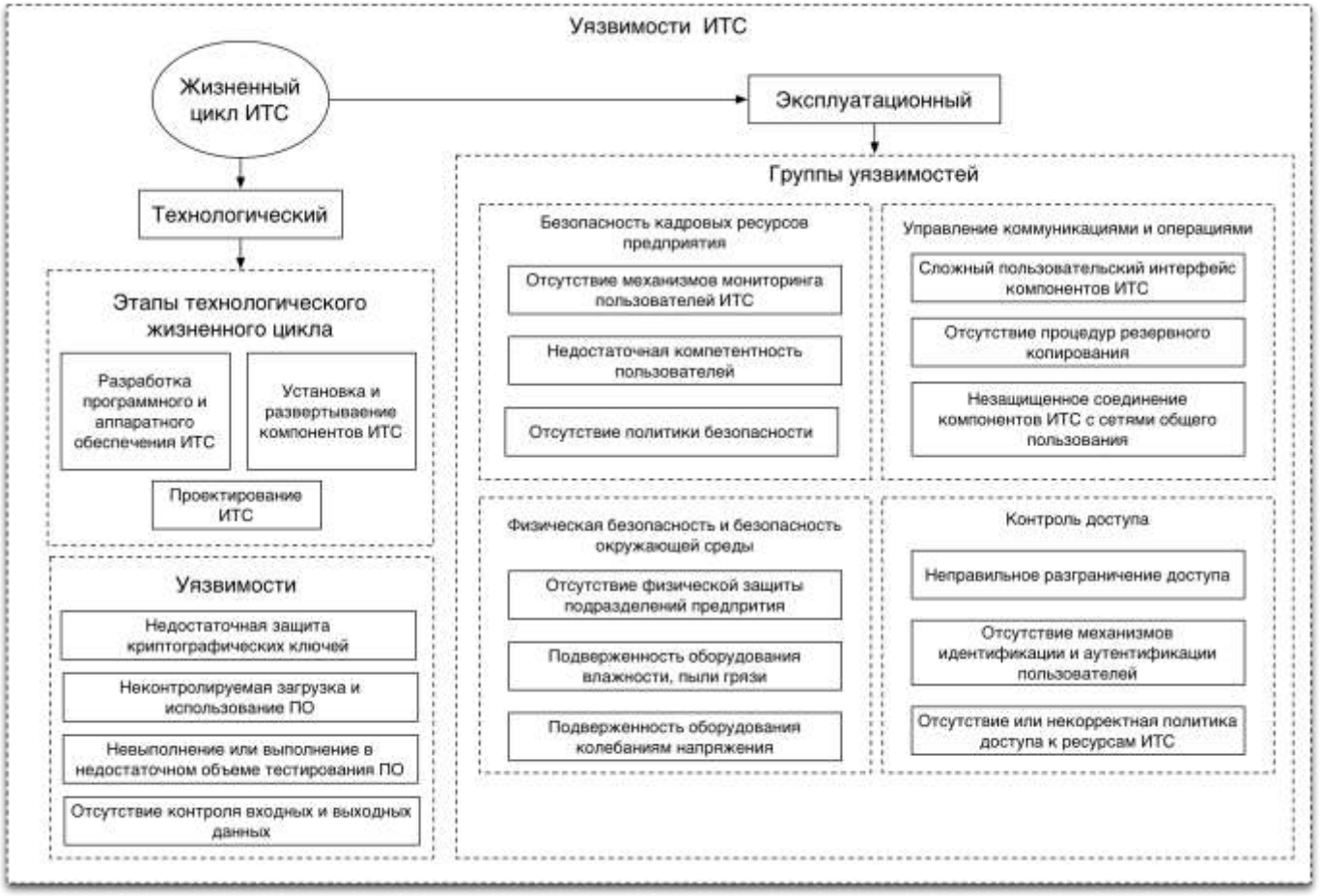


Рис. 6.6 – Анализ уязвимостей ИТС

Пусть $G = (S, \in E, R \in A \in F, H)$ — состояние ИТС. Определим элементы состояния G: E — множество компонентов; EC — множество узлов; V — множество уязвимостей; CC — множество коммуникационных каналов; IC — множество интерфейсов взаимодействия; S — множество субъектов; R — множество ребер графа-состояния G, соответствующих правам доступа пользователей к компонентам; A – множество ребер графа-состояния G, соответствующих доступам пользователей к компонентам; F — множество ребер графа-состояния G, соответствующих информационным потокам между компонентами; $H : E \to 2^E$ – функция иерархии сущностей, ее значения на множестве узлов EC соответствуют заданной в ИТС иерархии подчиненности компьютеров. При этом выполняются следующие условия:

1) $V \in E, CC \in E, IC \in E, EC \in E$;

2) множества V,CC,EC,IC попарно не пересекаются;

3) каждый компонент $e \in E$:

- либо является узлом ИС ($e \in EC$), либо размещена на некотором единственном для каждого компонента узле (для сущности $e \in E$ существует единственный узел $c \in EC$ такой, что $e < c$);

- либо является коммуникационным каналом ($e \in EC$);

- либо является интерфейсом взаимодействия с ИС($e \in IC$);

- либо является уязвимостью ($e \in V$);

- либо является пользователем или процессом пользователя ($e \in S$).

Для каждого узла (компьютера) $c \in EC$ определены доверенные пользователи, обладающие правом доступа владения к каждому компоненту, размещенным на данном узле, и коммуникационные каналы (компоненты-объекты), через которые осуществляется передача данных между субъектами узлов ИС. Если пользователь $os_c \in S$ является доверенным пользователем компьютера с или пользователь $s \in S$ обладает правом доступа владения own_r к компьютеру c в состоянии G, то он обладает правом доступа владения к каждому компоненту, размещенным на данном узле. Для каждого узла определены коммуникационные каналы и интерфейсы взаимодействия (компоненты-объекты), через которые осуществляется передача данных между субъектами узлов ИТС.

В зависимости от архитектуры системы безопасности современных ИТС возможно несколько способов передачи прав доступа при активизации и функционировании пользовательского процесса. Для реализации нарушений безопасности информации субъектами-нарушителями используются следующие уязвимости ИТС, приводящие к получению субъектом-нарушителем прав пользователей рассматриваемой ИТС:

1) ошибки в ПО, реализующем прикладные и системные процессы операционных систем;

2) ошибки в реализации, конфигурировании и использовании ИТС, приводящие к реализации информационных потоков по памяти и по времени между нарушителем и субъектами ИТС;

3) возможность получения или изменения некоторых параметров ИТС.

В ИТС выполняются условия:

1) при активации субъектом-пользователем $u \in S$ некоторого процесса $p \in S$ последниий наследует все права пользователя u;

2) уязвимости процесса функционально ассоциированы с субъектом-пользователем, от имени которого данный процесс запущен;

3) уязвимости, связанные с раскрытием параметров функционирования ИТС, параметрически ассоциированы с субъектом-пользователем, преобразования данных которого определяются этими параметрами.

6.8 Оценка вероятности проведения атак

На этапе оценки вероятности проведения атак анализ рисков определяется вероятность того, что в случае проведения атак на защищаемые ресурсы могут быть успешно преодолены все средства защиты, используемые в ИТС. Для этого для каждого информационного ресурса $d \in D$ разрабатываются графовые модели возможных атак, направленных на нарушение конфиденциальности, целостности или доступности ресурса. Графовые модели создаются рабочей группой, приводящей анализ рисков, на основе ранее составленных множеств D, S, H и Z , а также отношений B_1, B_2, B_3, B_4, B_5.

Каждая модель атаки представляет собой граф $G=<L, E>$, где L – множество вершин, а E – множество дуг графа G. Для графа G определено отношение $T \in \{E \times W\}$, которое каждой дуге из множества E ставит в

соответствие один или несколько элементов отношения *W*, что позволяет интерпретировать каждую дугу $e \in E$ как один из этапов развития атаки.

Для каждого графа составляется множество путей G_p, в котором каждый путь $g_p=(e_{p1}, e_{p2}, e_{p3}, \ldots, e_{pn})$, $e_{pn} \in E$, $g_p \in G_p$ описывает один из возможных сценариев развития атаки. Так, например, для графа *G*, изображенного на рис. 7, множество G_p включает два элемента, и, таким образом, описывает два возможных сценария атаки $G_p=\{(e_1), (e_3, e_2)\}$.

T={ $(e_1, (a_2, v_1, c_3))$,

$(e_2, (a_1, v_3, c_3))$,

$(e_3, (a_2, v_3, c_1))$,

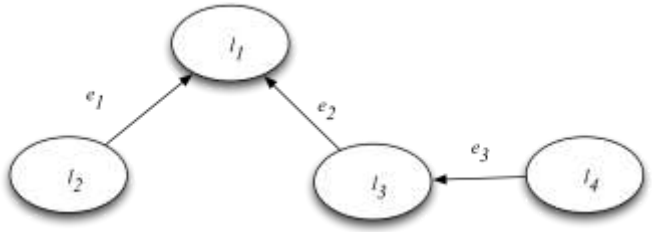


Рис. 6.7 – Пример графа атаки на ИТС

6.9 Оценка ущерба в случае нарушения услуг информационной безопасности (идентификация последствий)

Этап анализа рисков ИБ предполагает оценку ущерба *L*, который может быть нанесен в случае успешной атаки на информационные ресурсы из множества *D*. Оценка ущерба проводится по отношению к возможным последствиям атаки – нарушением конфиденциальности, целостности или доступности информационного ресурса. Модель предусматривает использование как количественных, так и качественных шкал для определения уровня ущерба. Количественные шкалы предполагают оценку математического ущерба по десятибалльной системе. При этом каждый балл шкалы должен соответствовать определенному уровню материальных

потерь, которые может понести организация в случае проведения успешной атаки на ресурс.

При использовании качественных шкал ущерб оценивается по пяти понятийным уровням — «малый ущерб», «умеренный ущерб», «ущерб средней тяжести», «большой ущерб», «критический ущерб». При оценке ущерба необходимо учитывать возможную потерю репутации компании, финансовые потери, правовые и судебные штрафы и др. Уровень ущерба определяется не членами рабочей группы, проводящей анализ рисков, а представителями организации, в ведении которой находится обследуемая ИТС.

В результате проведения пятого этапа оценки рисков формируется таблица, состоящая из следующих полей: порядковый номер элемента таблицы, информационный ресурс из множества D, тип последствия атаки на этот ресурс и уровень ущерба L.

6.10 Расчет значения рисков информационной безопасности

На данном этапе вычисляется значение риска R на основе ранее определенного уровня ущерба и вероятности атак. Для каждого защищаемого информационного ресурса $d \in D$ могут вычисляться три значения риска: риск нарушения конфиденциальности, целостности и доступности информационных ресурсов ИТС ОрВД. Значение риска может определяться по количественной или качественной шкале оценки. При использовании качественной шкалы оценка риска вычисляется на основе матрицы, которая приведена в табл. 2.6. В случае применения количественной шкалы риск вычисляется по стобалльной шкале как произведение значений ущерба и вероятности атаки, которые были определены на пятом и шестом этапах оценки, соответственно — $R_{ij} = P(G_j) \cdot L_j$, $i = \overline{1,...,|D|}, j = \overline{1,...,3}$. На этом завершается процесс оценки рисков, после которого следует процедура

анализа полученных значений и разработка рекомендаций по минимизации значений рисков информационной безопасности.

6.11 Разработки рекомендаций по управлению выявленными рисками информационной безопасности в ИТС

На данном этапе разрабатываются рекомендации по управлению рисками ИБ. В процессе управления рисками могут предприниматься следующие типы действий:

1) уменьшение риска за счет использования организационных и технических средств защиты, позволяющих снизить вероятность проведения атаки или уменьшить возможный ущерб от атаки;

2) уклонение от риска путем изменения архитектуры или схемы информационных потоков ИТС, что позволяет исключить возможность проведения той или иной атаки;

3) изменение характера риска в результате принятия мер по страхованию. В качестве примеров такого изменения характера риска можно привести страхования оборудования ИТС от пожара или страхование ресурсов ИТС от возможного нарушения их конфиденциальности, целостности и доступности;

4) принятие риска в том случае, если он уменьшен до того уровня, на котором он не представляет опасности для ИТС.

Процесс анализа рисков не является однократной процедурой, а должен повторяться при возникновении событий, требующих повторного проведения такого анализа, например, в случае установки новых средств защиты, выявления новых потенциальных источников атак и др.

6.12 Выводы

Риск информационной безопасности ИБ информационно-телекоммуникационной системы представляет собой интегральную оценку того, насколько эффективно существующие средства защиты способны противостоять информационным атакам. Процесс анализа рисков включает три основных этапа: этап сбора исходной информации об ИТС, этап оценки рисков на основе собранных данных и этап разработки рекомендаций по управлению выявленными рисками информационной безопасности ИТС.

Предложенная в НИР модель оценки и управления рисками информационной безопасности ИТС может быть внедрена в концепцию информационной безопасности организаций и предприятий как унифицированная.

7 МЕТОДЫ ГЕНЕРАЦИИ ПСЕВДОСЛУЧАЙНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ И ВОЗМОЖНОСТЬ ИХ РЕАЛИЗАЦИИ С ИСПОЛЬЗОВАНИЕМ ГРАФИЧЕСКИХ ПРОЦЕССОРОВ

Любая информационная система, используемая в сфере защиты информации, должна предоставлять всем пользователям основные услуги, среди которых: целостность, наблюдаемость, доступность, конфиденциальность и неопровержимость. Создание комплексной системы защиты информации является одним из наиболее эффективных решений при проектировании и использовании системы в сфере защиты информации, что позволяет противодействовать возможным угрозам и минимизировать убытки. При этом качество предоставления описанных выше услуг, обеспечивается только при использовании криптографических преобразований, таких как электронная цифровая подпись, симметричное шифрование и д.р. Для реализации криптографических преобразований информации используются аппаратные, программные, и программно-аппаратные средства. Стойкость криптографических преобразований обеспечивается при условии, что ключевые параметры генерируются равновероятно и случайно с высокой производительностью. Для генерации ключевых параметров используются генераторы псевдослучайных последовательностей (ПСП).

На сегодняшнее время среди всех известных генераторов псевдослучайных последовательностей, большинство имеют ряд серьёзных недостатков, а последовательности, полученные при помощи таких генераторов, не отвечают основным требованиям, которые выдвигаются к генераторам ПСП для использования в криптографических целях. Основными недостатками генераторов ПСП являются: недопустимо

короткий или недоказанный период повторения последовательности Y_i; недостаточный уровень необратимости относительно нахождения ключа K_i (УП), что позволяет решать задачу нахождения Y_i с полиномиальной или субэкспоненциальной сложностями; недостаточная неразличимость Y_i, что также делает ее определенным образом предсказуемой «вперед и назад»; свойства случайности, равновероятности, независимости и однородности не соответствуют требованиям и другие [1].

Одним из наиболее важных и необходимых направлений исследований при создании эффективных генераторов псевдослучайных последовательностей является разработка методов та средств оценки статистических свойств псевдослучайных последовательностей. Статистические показатели и построенные на основе них критерии оценки – являются инструментом проверки правильности технических решений относительно их построения.

Функциональные классы и оценочные методологии построения генераторов ПСП представлены в стандартах AIS-20 [3], AIS-31 [4], ISO/IEC-18031 [5], которые позволяют систематизировать и сделать выводы о проведенных исследованиях.

Одним из ключевых аспектов в принятии решений об эффективности процесса формирования псевдослучайных последовательностей является тестирования ПСП. Используются различные категории тестов, в частности, графические тесты и статистические тесты. К графическим тестам относят следующие: гистограмма распределения элементов последовательности, распределение на плоскости, графический спектральный тест и др. К наиболее широко используемым статистическим тестам относят: Diehard, NIST, тесты Д. Кнута.

В работе приведены оценки, полученные в результате проведения тестирования генераторов ПСП в рамках анализа объекта исследования и усовершенствованных методов.

7.1 Критерии и показатели оценки свойств генераторов псевдослучайных последовательностей.

Под критерием будем понимать признак, на основе которого осуществляется оценка, определение или классификация чего-либо, то есть, по сути, будем понимать правило предпочтения. Рассматривают две совокупностей критериев: безусловные и условные. Оценку свойств генератора ПСП рекомендуется выполнять в два этапа. На первом этапе проверяется их соответствие безусловным критериям, а на втором получают соответствующие интегральные оценки с использованием условных критериев.

К безусловным критериям относят те критерии, выполнение которых для генератора ПСП является обязательным, то есть безусловным. Анализ применения, опыт разработки и оценки свойств генератора ПСП, достигнутые результаты при практическом решении задач криптоанализа и реализации различных атак позволяют в качестве основных выбрать, как минимум, такие безусловные критерии оценки [1]:

- надежность математической базы, применяемой в генераторе ПСП;

- практическая защищенность генератора ПСП от известных атак;

- реальная защищенность генератора ПСП от всех известных и возможных крипто аналитических атак;

- статистическая безопасность генератора ПСП;

- непредсказуемость псевдослучайных последовательностей (вперед и назад), генерируемых генератором ПСП;

- отсутствие слабых тайных (личных) ключей;

- сложность генерирования составляет не выше полиномиального характера.

Рассмотрим более подробно эти критерии, как в части понятий и определений, так и в части особенностей применения.

1. Надежность математической базы рассматривается, как отсутствие у нарушителя возможностей совершать атаки типа «универсальное раскрытие» за счет несовершенства математической базы генератора ПСП или слабостей, которые могут быть заложены за счет специфических свойств общих параметров и ключей. При этом критерием оценки надежности математической базы является тот факт, что сложность атаки «универсальное раскрытия» $W_{УР}$ имеет экспоненциальное характер, а критерием ненадежности - субэкспоненциальный или полиномиальный характер сложности. Будем обозначать этот критерий $W_{б1}$.

2. Практическая защищенность генератора ПСП от известных силовых и аналитических атак, которая достигается за счет выбора размеров общих параметров и ключей. Критерием практической защищенности генератора ПСП является выбор таких размеров общих параметров и ключей, при которых сложность атаки $W_{са}$ существенно (на необходимое число порядков) превышает существующую мощность крипто аналитических систем на уровне технологически развитых государств (нарушителя третьего уровня), в том числе с учетом прогноза увеличения мощности крипто аналитических систем за счет развития математического, программного обеспечения и т.д. Обозначим этот безусловный критерий $W_{б2}$.

3. Реальная защищенность генератора ПСП от всех известных и возможных крипто аналитических атак, где под защищенностью понимают тот факт, что все известные крипто аналитические атаки типа «полное раскрытие» имеют экспоненциальную сложность $W_{эс}$, а под критерием незащищенности - субэкспоненциальный $W_{сэ}$ и ниже характер сложности

атаки «полное раскрытие». Будем обозначать этот безусловный критерий $W_{б3}$ и понимать под ним оценку необратимости.

4. Статистическая безопасность генератора ПСП, под которой понимают статистическую независимость исходных значений (выхода) генератора, например, от входных. Будем обозначать этот безусловный критерий $W_{б4}$.

5. Непредсказуемость «вперед и назад», под которой понимается отсутствие атак, связанных с определением как будущих и предыдущих ПСП, сложность которых меньше, чем сложность атаки типа «полное раскрытие». Этот безусловный критерий будем обозначать $W_{б5}$.

6. Отсутствие слабых личных (тайных) ключей генератора ПСП, при которых сложность крипто аналитических атак типа «полное раскрытие» и «универсальное раскрытия» меньше, чем сложность атаки «полное раскрытие» для других («не слабых») личных ключей. Обозначим этот безусловный критерий как $W_{б6}$.

7. Сложность генерирования $I_{ген}$ носит полиномиальный характер и не превышает допустимой величины $I_д$. Обозначим этот безусловный критерий также $W_{б7}$.

Так как приведенные частичные критерии являются безусловными, то интегральным критерием отбора является логическое изменение да / нет (1/0), и безусловный критерий можно записать в виде:

$$W_{\delta 1}, W_{\delta 2}, W_{\delta 3}, W_{\delta 4}, W_{\delta 5}, W_{\delta 6}, W_{\delta 7} \in \{0;1\}. \qquad (7.1)$$

Таким образом, качество генератора ПСП может быть оценено с использованием безусловного интегрального критерия - функции соответствия генератора ПСП требованиям: $f_{фв} \in [0;1]$.

Количественная оценка генераторов может быть сделана с использованием таких показателей, как [2]:

- сложность обращения генератора ПСП, т.е. нахождение используемого ключа (УП);

- энтропия источника ключей $H_к$, в том числе для случая, когда генератора ПСП используется как источник ключей;

- период $l_п$ (длина) повторения ПСП и безопасное время t_b;

- основа алфавита;

- вероятность перекрытия в пространстве или во времени двух сегментов битов Y_r и Y_μ, то есть в разных абонентах или у одного абонента в течение времени, так, что $Y_r = Y_\mu$;

- расстояние равнозначности конкретной последовательности битов Y_υ;

- пространственная $I_п$ и временная сложности $I_в$ формирования последовательности битов Y т.д.

Требованием минимальной защиты для генератора ПСП является требование достаточно большой длины k случайного начального числа, такой, чтобы поиск по 2^k - элементов был бы невыполнимым для нарушителя.

Основными общими требованиями к генератора ПСП являются:

- требование неразличимости исходных последовательностей генератора ПСП от истинно случайных последовательностей;

- требование непредсказуемости выходных битов для нарушителя с ограниченными вычислительными ресурсами:

- требование необратимости генератора в смысле предварительно заданной малой вероятности компрометации ключа самого генератора ПСП.

Таким образом, ПСП должна иметь некоторые статистические свойства, которые присущи для псевдослучайных последовательностей.

Принципиальным отличием генератора ПСП для стохастических моделирований и криптографических приложений является то, что в криптографических системах, в системах связи, использующих динамическую смену соответствия «m бит - 2^m сложных сигналов», генератора ПСП дополнительно должны обеспечивать устойчивость против осуществления различного рода атак. Как показал анализ, атаки в системах, использующих генератора ПСП, могут осуществляться в целях:

- нахождения неизвестных исходных данных (ключей, ключевой информации и параметров, генерируемых для других криптографических приложений);

- нахождения информации о внутреннем состоянии, в первую очередь значение ключа непосредственно самого генератора;

- манипулирования исходными данными генератора.

Потенциально возможные атаки можно разделить на три таких класса:

- крипто аналитические атаки, которые сводятся к решению математических задач обращения генератора, сложность которых меньше или существенно меньше атаки «грубая сила»;

- атаки, основанные на использования входных данных генератора;

атаки, основанные на недостаточных показателях непредсказуемости.

По особенностям воздействия нарушителя потенциально возможные атаки можно разделить на две большие группы:

- пассивные атаки, при осуществлении которых нарушитель отслеживает источник шума и части выходных данных генератора. Если он сможет обнаружить любые корреляции между этими двумя потоками, то он сможет использовать эти знания для прогнозирования предстоящих выходных данных, то есть ключей и / или параметров;

- активные атаки, при осуществлении которых нарушитель может повлиять на источник шума и, таким образом, иметь возможность влиять на последовательность битов, создаваемых генератором. Данный вид атаки особенно эффективен, так как злоумышленник может попытаться адаптировать входные шумы с характерными свойствами генератора ПСП для получения предполагаемых исходных данных.

Генераторы ПСП должны быть защищены от таких основных атак:

- крипто аналитические атаки, сводятся к решению математических задач, сложность которых меньше или существенно меньше атаки грубая сила;

- атаки, основанные на входных данных генератора, которые осуществляются посредством определения или манипулирования входными данными генератора ПСП с целью влияния на выходные данные (ключи, параметры и т.д. для других криптографических приложений);

- атаки, которые сводятся к распространению компрометируемого состояния генератора на другие состояния - предварительные или последующие.

Безусловным требованием к генераторам ПСП является необходимость предотвращения любых атак, которые позволяют злоумышленнику получить информацию о его внутреннем состоянии.

В связи с выявленными противоречиями и проблемными вопросами, на наш взгляд, в дальнейшем должны быть решены такие проблемные вопросы.

1. Обоснование и построение ДГСП на основе бихэвиористического подхода, при котором последовательность, генерируемая ДГСП, считается псевдослучайной, если не существует эффективного алгоритма, который сможет ее отличить от действительно случайной последовательности той же длины, т.е. обладает хорошими свойствами неразличимости.

2. Совершенствование методов и средств оценки как самого генератора, так и статистических свойств псевдослучайных последовательностей, а также сравнительного анализа генератора ПСП как по отдельным, так и интегральным критериям и показателям.

3. В связи с тем, что на сегодняшний день разработаны и широко используются ряд методик тестирования статистических свойств псевдослучайных последовательностей, таких как FIPS PUB 140-1 [6], FIPS PUB 140-2 [7], AIS 20 [8] и AIS 31 [9], необходимо разработать методику исследования свойств генератора ПСП, основанную на комплексном контроле эффективности функционирования генератора ПСП.

4. Заслуживает внимания и дальнейшего развития ряд новых и усовершенствованных методов построения генератора ПСП, в том числе на основе спаривания точек эллиптических кривых, результат которого определяет подгруппы расширения поля с заранее известным гарантированным периодом, каждый элемент которого хешируется.

5. Теоретические исследования методов и генерации ПСП, основанные на рекуррентном формировании элементов полей и подполей Галуа и их преобразовании методом хеширования в ПСП с необходимыми свойствами необратимости, неразличимости, непредсказуемости и быстродействия.

7.2 Методы генерации псевдослучайных последовательностей.

7.2.1 Конгруэнтный метод генерации ПСП

Наиболее распространенным методом генерации ПСП является конгруэнтный метод, который для генерации ПСП использует правило: $x_n = (ax_{n-1} + k) \bmod m$, где m - модуль, a - множитель, k - аддитивная константа и x_0 - инициализирующий случайный сид.

Если a является примитивным корнем простого числа p, а x_0 - случайным сидом из множества $Z = \{1,2,3,..., p-1\}$, то последовательность, порожденная $x_n = (ax_{n-1} + k) \bmod m$, будет строго периодическая, с периодом $p-1$, и каждый элемент этой последовательности будет равномерной случайной величиной на множестве Z, однако при этом элементы такой последовательности не будут независимыми между собой.

Если взять четыре последовательных числа x, y, z и w, сгенерированных линейным конгруэнтным генератором ПСП с множителем a, то [10-11]:

- точка (x, y, z) попадает на решетку точек, сгенерированных линейных комбинаций точек $(1, a, a^2)$, $(0, m, 0)$, $(0, 0, m)$ с целочисленными коэффициентами;

- аналогично, любая точка (x, y) будет попадать на решетку, сгенерированную всеми линейными комбинациями точек $(1, a)$ и $(0, m)$ с целочисленными коэффициентами;

- точка (x, y, z, w) в четырехмерном пространстве будет попадать на решетку целочисленных комбинаций четырех точек $(1, a, a^2, a^3)$, $(0, m, 0, 0)$, $(0, 0, m, 0)$, $(0, 0, 0m)$.

Предположим, что α, β, τ - какие-либо три точки на плоскости с координатами, которые являются последовательными результатами генератора ПСП. Тогда определитель матрицы 2×2 с рядами $(\beta - \alpha)$ и $(\tau - a)$ будет давать объем параллелепипеда, определенного этими тремя точками и объем параллелепипеда должен быть кратен m. Таким образом, наибольший общий делитель пяти или шести таких определителей будет равен m и, в этом случае, можно определить аддитивную константу k и множитель a.

7.2.2 XORSHIFT метод генерации ПСП

Рассматривая 32-х (или 64-х) битное целое число как элемент векторного пространства с компонентами в поле $\mathrm{mod}\,2$, сложение двух векторов может быть реализовано как исключающее или (XOR) операция, что в сочетании с операцией сдвига, может быть использовано для создания некоторых линейных преобразований над этим векторным пространством. Множество сидов Z является множеством всех ненулевых двоичных векторов 1×32, а f является линейной трансформацией на множестве Z, представленной двоичной матрицей T 32×32, где T - невырожденная.

Для случайного сида $y\in Z$ последовательность $yT, yT^2, yT^3\ldots$ будет иметь период $2^{32}-1$ тогда и только тогда, когда порядок матрицы T равен $2^{32}-1$ в группе 32×32 невырожденных двоичных матриц. Если $T=(I+L^a)\cdot(I+R^b)\cdot(I+L^c)$, где L-матрица, которая дает сдвиг влево на 1 (в C, $y\wedge=(y<<1)$, таким образом yL^a в C - есть $y\wedge=(y<<a)$, то можно получить простой и быстрый способ для формирования матричного произведения [11]. Матрица R, являющаяся транспонированной L, дает сдвиг вправо на единицу. Таким образом, для $T=(I+L^a)\cdot(I+R^b)\cdot(I+L^c)$, для случайного 32-битного сида y из Z, каждый новый y в последовательности $yT, yT^2, yT^3\ldots$ может быть получен посредством последовательного применения следующих трех инструкций: $y\wedge=y<<13$, $y\wedge=y>>17$, $y\wedge=y<<5$.

Для 32-х (или 64-х) битных двоичных векторов отсутствуют двухсдвиговые матрицы $T=(I+L^a)\cdot(I+R^b)$, которые имеют полный период и нет одноздвиговых, поэтому необходима 3-х сдвиговая матрица T. Существует точно 81 тройка [11] $[a,b,c]$, $a<c$, для которых 32×32 двоичная матрица $T=(I+L^a)\cdot(I+R^b)\cdot(I+L^c)$ имеет порядок: $2^{32}-1$.

Если матрица $T=(I+L^a)\cdot(I+R^b)\cdot(I+L^c)$ имеет полный период, то и $(I+L^c)\cdot(I+R^b)\cdot(I+L^a)$ и $(I+L^a)\cdot(I+L^c)\cdot(I+R^b)$ также имеют полный период, что дает возможность получить 4×81 матриц T с порядком $2^{32}-1$. Но тогда и транспонированная матрица каждой из них имеет полный период. Таким образом, могут быть получены 648 матриц.

7.2.3 Метод генерации ПСП Фибоначчи с запаздываниями

Базовое рекуррентное соотношение для генераторов ПСП Фибоначчи с запаздываниями представляется в виде $x_n = x_{n-r} \bullet x_{n-s}$ для r и s при $r > s$ [12]. Операция $\bullet$ определяется как бинарные отношения для пар элементов в некотором множестве χ, и множество сидов Z представляет собой множество r-кортежей $(x_1, x_2,..., x_r)$, где $x \in \chi$. Обычно χ является множеством 32-х битных целых чисел, а операция $\bullet$ является сложением или вычитанием по $\bmod 2^{32}$. Правило для $\bullet$ может быть основано на выражении элементов x, y из χ в форме $x=\pm3^a, y=\pm3^b \mod 2^{32}$ так, что $x\bullet y=\pm3^{(a+b)\bmod 2^{30}}$ и верно рекуррентное правило для сложения $\bmod 2^{30}$. Для генераторов ПСП Фибоначчи с запаздываниями используется обозначение $F(r,s,\bullet)$. Для правильного выбора запаздываний r, s период $F(r,s,\pm \bmod 2^{32})$ должен быть 2^{32+r}, в то время как $F(r,s,\oplus)$, будет 2^r не зависимо от размера слова. Для правильного выбора $r > s$ период генератора $F(r,s,*$нечетных по модулю $2^{32})$ будет 2^{30}.

В отличие от конгруэнтного и Xorshift методов генерации ПСП, метод генерации ПСП Фибоначчи с запаздываниями требует множество сидов Z и функцию f, определенную над Z. Под Z в данном случае подразумевается множество кортежей $\{[x_1, x_2,...x_r]\}$, где x из множества χ, на котором определено бинарное отношение и функция f: $f([x_1, x_2,..., x_r]) = [x_2,..., x_r, x_1 \bullet x_{r-s+1}]$.

Реализация последовательностей Фибоначчи, с запаздываниями $r > s$ требует хранения таблицы последних r значений. Одно из самых полезных применений генератора ПСП Фибоначчи с запаздываниями является непосредственная генерация равномерных случайных чисел с плавающей запятой на интервале $[0,1)$.

Предположим, необходимо сгенерировать 64-х битные равномерные $[0,1)$ случайные переменные воспользовавшись стандартом IEEE 754 (стандарт формата представления чисел с плавающей точкой, используемый как в программных реализациях арифметических действий, так и аппаратных реализациях): 1 знаковый бит, 11 битов экспоненты, 52 бита мантиссы с подразумеваемой ведущей единицей. Для указанного бинарного отношения $x \bullet y$ используют правило: если $x \geq y$, тогда $x - y$, в противном случае - $x - y + 1$. Если x и y - представление рациональных чисел $a / 2^{53}$ и $b / 2^{53}$ с плавающей запятой, то $x \bullet y$ будет давать в результате точное значение $c / 2^{53}$ с плавающей запятой, где $c = (x - y) \bmod 2^{53}$.

7.3 Методы генерации ПСП с использованием графических процессоров.

7.3.1 Метод генерации ПСП Блюма-Блюма-Шуба (BBS) и возможность его реализации на GPU

Генератор псевдослучайных чисел BBS [13] представляет собой метод, который можно описать уравнением:

$$x_{n-1} = x_n^2 (\bmod M) \tag{7.2}$$

где M - является произведением двух простых чисел p и q.

Простые числа p и q имеют особые свойства и описываются соотношениями: $p \equiv 3 \bmod 4$, $p = 2p_1 + 1$, $p_1 = 2p_2 + 1$, где p_1 и p_2 простые числа.

Исходное значение генератора x_0 является квадратичным вычетом от M. Указанное означает, что существует целое число y, такое, что $y^2 = x_0 \bmod M$. Значение x_0 может быть получено путем выбора случайного сида (seed) s меньшего, чем $\frac{M}{2}$, но большего чем 0 и удовлетворяющего выражению: $НОД(s, M) = 1$, тогда $x_0 = s^2$.

Генератор BBS имеет периодичность и это означает, что после некоторого числа итераций, он будет генерировать x_0 и сгенерированная последовательность будет повторяться. Периодичность может быть увеличена за счет выбора простых чисел p и q, для которых $НОД(p-1, q-1)$ будет как можно меньше.

Функция от натурального числа n - $\lambda(n)$, называемая Carmichael функция, определяется как наименьшее натуральное число m такое, что: $a^m \equiv 1 (\bmod n)$. Возможно рассчитать периодичность, используя функцию $\lambda(\lambda(M))$, однако для этого необходимо, чтобы выражения $p = 2p_1 + 1$ и $p_1 = 2p_2 + 1$ выполнялись для чисел p и q. Генератор BBS дает возможность перехода на любой шаг i в последовательности, если использовать следующее выражение:

$$x_n = x_0^{2^i \bmod \lambda(M)} (\bmod M). \qquad (7.3)$$

Количественная оценка стойкости генератора может быть выполнена с использованием таких показателей как: энтропия источника ключей H_x, период l_n (длина) повторения ПСП, основа алфавита m последовательности,

вероятность перекрытия в пространстве или во времени битов Y_r и Y_μ и другие.

Количественная оценка стойкости, проведенная Блюмом, позволила установить число бит, которые могут быть получены и использоваться в криптографических целях для каждой итерации генератора. Полученное значение равное $\log_2(\log_2 M)$.

Поскольку безопасность генератора основывается на большом числе M не меньшем чем 1000 бит, что превышает длину регистров любого современного процессора, поэтому требуется применение методов вычислений с многократной точностью.

Такие методы могут основываться на использовании позиционной системы счисления, где машинное слово некоторого процессора является цифрой в представлении числа, состоящего из некоторого количества машинных слов, а максимальное значение, которое можно хранить в машинном слове, является базой этой системы счисления.

Использование алгоритма Монтгомери позволит ускорить выполнение операций умножения и возведения в квадрат, что необходимо при возведении числа в степени по модулю, когда модуль составляет порядка 1000 бит. Также Монтгомери определил способы выполнения модульных операций на базе операций сложения и умножения.

Операцию умножения, определенную в алгоритме Монтгомери, можно использовать для реализации BBS (Blum Blum Shub) метода на GPU (Graphics processing unit).

7.3.2 Метод ПСП Вихрь Мерсена и его реализация на GPU

Для синтеза генератора ПСП, который бы обладал высокими скоростными показателями необходимо использовать алгоритмы,

основанные на элементарных операциях с двоичными (битовыми) строками. Наряду с генераторами, основанными на регистрах сдвига с линейными обратными связями (LFSR), применяются генераторы на регистрах сдвига с обобщенными обратными связями (GFSR), к числу которых принадлежит и рассматриваемый ниже генератор Вихрь Мерсена [14].

Выходные значения генератора Вихря Мерсена есть битовые вектора фиксированной длины посредством следующего рекуррентного соотношения:

$$x_{k+n} = x_{k+m} + (x_k^{upper} \mid x_{k+1}^{lower}) \bullet A, \tag{7.4}$$

где: $n > m$ - фиксированные положительные целые числа;

$x_k, k = 0,1,...$ - последовательность битовых векторов фиксированной длины w, равная 32;

$x_k^{upper} \mid x_{k+1}^{lower}$ - конкатенация r старших битов слова X_k и $w-r$ младших битов слова X_{k-1};

$(x_k^{upper} \mid x_{k+1}^{lower}) \bullet A$ - произведение конкатенированного битового вектора на битовую матрицу A.

Матрица A выбирается из следующего соотношения:

$$x \bullet A = \{x_{w-1}, x_{w-2},..., x_0\} \bullet \begin{Vmatrix} 0 & 1 & 0 & 0 & & \\ 0 & 0 & 1 & 0 & \cdots & \\ 0 & 0 & 0 & 1 & & \\ & & & & 1 & \\ & & & \cdots & & \ddots \\ & & & & & 1 \\ a_{w-1} & a_{w-2} & & & a_1 & a_0 \end{Vmatrix}, \tag{7.5}$$

где $x = \{x_{w-1}, x_{w-2},..., x_0\}$, и $a = \{a_{w-1}, a_{w-2},..., a_0\}$.

Битовые значения складываются по модулю 2 (то есть с использованием XOR операции).

Вихрь Мерсена имеет следующий набор параметров:

w - длина слова;

n, m - степень рекурсии и средний терм;

r - точка разделения между x_k^{upper} и x_k^{lower};

a - битовый вектор, нижняя строка матрицы A;

l, u, s, t - параметры сдвига;

b, c - маски, представленные в виде битовых векторов.

Несмотря на большой период генератора, равный $2^{19937}-1$, и на высокие показатели прохождения тестов DIEHARD, что свидетельствует о хороших статистических свойствах, последовательность, сгенерированная таким генератором может быть использована в криптографических целях только совместно с алгоритмами хеширования.

Метод, на основе которого реализован генератор Вихрь Мерсена, требует большого объема памяти для хранения своих состояний на разных стадиях алгоритма. В случаях, когда точность вычислений преобладает над скоростью, имеет смысл использование данного метода генерации ПСП. Готовая реализация данного метода содержится в NVIDIA SDK (комплект средств разработки от NVIDIA).

7.3.3 Методы параллельных вычислений Монте-Карло

Реализация генераторов псевдослучайных чисел на графических процессорах наиболее применима в контексте симуляции методом Монте-Карло, так как решение задач с применением метода Монте-Карло достаточно хорошо распараллеливается и, в тоже время, такие задачи требуют значительных вычислительных мощностей для проведения большого количества итераций [15-17].

Важнейшим принципом построения методов Монте-Карло есть сведение поставленной задачи к расчету математических ожиданий. Для того, чтобы приближенно вычислить некоторую скалярную величину α, надо определить такую случайную величину ξ, что $M[\xi]=\alpha$, тогда вычислив N независимых значений $\xi_1,...,\xi_N$ величины ξ, можно считать, что $\alpha \approx \frac{\xi_1+\xi_2+...+\xi_N}{N}$.

Численные методы Монте-Карло, используемые для решения математических задач при помощи моделирования случайных величин (для моделирования случайных величин используются генераторы ПСП), требуют больших вычислительных мощностей.

Определим несколько подходов, позволяющих распараллелить методы Монте-Карло.

Первый подход заключается в разделении последовательности на потоки. Пусть p - период используемого генератора псевдо случайных чисел, а n- длина последовательности, такая что $n \le p$. Тогда последовательность длиной n разделяется на последовательности равной длины $n_1 = n_2 = ... = n_i$ по одной из каждого P потока. Сложность данного подхода заключается в вычислении начального элемента в i-том потоке, что является трудоемкой операцией.

Второй подход заключается в параметризации (использовании различных входных параметров для генератора ПСП), что позволяет использовать в разных потоках разные независимые случайные последовательности. При таком подходе необходимо убедиться в том, что сгенерированные последовательности независимы между собой.

С использованием параллельных вычислений проблема использования больших вычислительных мощностей легко решаема и делает доступной использование методов Монте-Карло. При этом методы Монте-Карло

являются достаточно легко программируемыми и позволяют рассчитывать задачи, не доступные для классических численных методов, чем и обосновывается выбор данных методов для дальнейших исследований.

7.4 Выводы

В рамках проведенного моделирования методов генерации псевдослучайных последовательностей были получены следующие результаты [18]:

- конгруэнтный генератор ПСП не отвечает требованиям, выдвигаемым тестом DIEHARD, в частности, тестам: BINARY RANK TEST для 6x8 матриц, BITSTREAM TEST (OVERLAPPING 20-кортежей), OPSO, OQSO и DNA, COUNT-THE-1's TEST;

- Xorshift генератор ПСП, не отвечает требованиям определенным тестом DIEHARD, в частности таким как: BINARY RANK TEST для 6x8 матриц, OPSO, OQSO, COUNT-THE-1's TEST.

- генератор ПСП Фибоначчи с запаздываниями, не отвечает требованиям определенным тестом DIEHARD, в частности требованиям: OPSO, OQSO, DNA.

В контексте базовых арифметических алгоритмов операция умножения может быть легко распараллелена в отличие от операции деления. Распараллеливание операции сложения, можно выполнить аналогично с реализацией сумматоров с переключением переноса.

Выигрыш в производительности при переносе реализации метода BBS (Blum Blum Shub) на GPU (Graphics processing unit) будет только в том случае, если GPU параллельно будет выполнять ряд сопутствующих задач, например, реализация методов Монте-Карло. Если ставить целью непосредственную реализацию рассмотренных алгоритмов генерации, то затраты на построение связи между CPU и GPU будут значительно большие, чем выигрыш от переноса вычислений на GPU [19].

ВЫВОДЫ

1. Универсальное хеширование определяет требование - размер ключевого пространства должен быть не меньше пространства сообщения. Почти универсальное хеширования ослабляет это условие, размер ключа может быть больше или равен размеру хеш кода.

2. Строго универсальный хеш класс характеризуется предельным наименьшим значением вероятности имитации и подмены, но размер ключевого пространства не меньше чем в квадрат раз превышает размер хеш кода.

3. Коллизионные оценки почти строго универсального хеширования связываются с распределениями хешей для пар сообщений по ключевому пространству, что определяет безусловную аутентификацию на массивах аутентификаторов связности два.

4. Результаты международных проектов NESSIE и NIST SHA-3 Competition подтверждают актуальность решения задачи разработки хеш функций гарантированной стойкости для построения национального стандарта хеширования.

5. Практические алгоритмы вычислений должны поддерживать эффективные вычисления на различных типах платформ процессоров 8, 32 и 64 разрядности, а перспективе и 128 разрядов, возможность реализации алгоритмов с параллельными вычислениями и наборами инструкций с высоким быстродействием, структурно простыми и прозрачными для анализа криптографической стойкости.

6. Гарантированная стойкость к атакам реализуется в теории доказуемо стойкой аутентификации. Доказуемо стойкая аутентификация строится на основе универсального хеширования. Практическим алгоритмом является UMAC алгоритм. Алгоритмы универсального хеширования определяются строгими теоретическими оценками безопасности и имеют самую высокую скорость вычислений.

7. Для точного вычисления имитационной и коллизионной стойкости MAC кодов необходимо использовать статистику совместных распределений MAC кодов по ключам для исходных и навязываемых сообщений. Для практических MAC кодов знание такой статистики выглядит проблематичным из-за чрезвычайно больших размеров массива аутентификаторов.

8. Нижние границы для вероятностей имитации и подмены определяются мощностью пространства ключей и MAC кодов, не учитывают статистические свойства массивов аутентификаторов. Требования к вероятности подмены и имитации определяют минимальные требования к размеру ключевого пространства и пространства MAC значений.

9. Верхние границы для вероятностей имитации и подмены связаны с комбинаторными свойствами MAC массивов и определяют значения вероятности коллизий на пространстве $A \times B$ для наихудшего случая выбора ключей и сообщений.

В рамках проведенного моделирования методов генерации псевдослучайных последовательностей были получены следующие результаты:

- конгруэнтный генератор ПСП не отвечает требованиям, выдвигаемым тестом DIEHARD, в частности, тестам: BINARY RANK TEST для 6x8 матриц, BITSTREAM TEST (OVERLAPPING 20-кортежей), OPSO, OQSO и DNA, COUNT-THE-1's TEST;

- Xorshift генератор ПСП, не отвечает требованиям определенным тестом DIEHARD, в частности таким как: BINARY RANK TEST для 6x8 матриц, OPSO, OQSO, COUNT-THE-1's TEST.

- генератор ПСП Фибоначчи с запаздываниями, не отвечает требованиям определенным тестом DIEHARD, в частности требованиям: OPSO, OQSO, DNA.

В контексте базовых арифметических алгоритмов операция умножения может быть легко распараллелена в отличие от операции деления.

Распараллеливание операции сложения, можно выполнить аналогично с реализацией сумматоров с переключением переноса.

Выигрыш в производительности при переносе реализации метода BBS (Blum Blum Shub) на GPU (Graphics processing unit) будет только в том случае, если GPU параллельно будет выполнять ряд сопутствующих задач, например, реализация методов Монте-Карло. Если ставить целью непосредственную реализацию рассмотренных алгоритмов генерации, то затраты на построение связи между CPU и GPU будут значительно большие, чем выигрыш от переноса вычислений на GPU.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Столингс В. Криптография и защита сетей. Принципы и практика. 2-е изд-е / В.Столингс. – К.: Изд. дом “Вильямс”, 2001. - 669с.

2. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке СИ / Б.Шнайер. – М.: «Триумф», 2002. – 797с.

3. Задірака В. Комп’ютерна криптологія / В.Задірака, О.Олексик. – К., 2002. – 502с.

4. Бессалов А. Криптосистемы на эллиптических кривых / А.Бессалов, А.Телиженко. – К.: «Політехніка», 2004. – 224с.

5. Горбенко І.Д. Захист інформації в інформаційно-телекомунікаційних системах. Част. 1. Криптографічний захист інформації / І.Д.Горбенко, Т.О.Гриненко. – Харків: ХНУРЕ, 2004. – 367с.

6. Вембо Мао. Современная криптография. Теория и практика / Вембо Мао. – Компания Hewlett-Packard – Санкт-Петербург–Киев, 2005. – 763с.

7. Горбенко Ю.І. Інфраструктури відкритих ключів. Електронний цифровий підпис. Теорія та практика: монографія / Ю.І.Горбенко, І.Д.Горбенко. –Харків: Видавництво «Форт», 2010. - 608 с.

8. Корченко А.Г. Построение систем защиты информации на нечетких множествах / А.Г. Корченко. – К.: «МК-Пресс», 2006. – 320 с.

9. International Organization for Standardization, “ISO/IEC 9797-1: Information Technology - Security Techniques - Message Authentication Codes (MACs) – Part 1: Mechanisms using a block cipher", 1999. [Электронный ресурс].– Режим доступа: http://www.iso.org/iso/catalogue_detail.htm? csnumber

10. International Organization for Standardization, \ISO/IEC 9797-2: Information Technology - Security Techniques - Message Authentication Codes (MACs) – Part 2: “Mechanisms using a dedicated hash-function", 2002. [Электронный ресурс].– Режим доступа: http://www.iso.org/iso/catalogue.

11. ДСТУ ISO/IEC 9798-1 Методи захисту. Автентифікація об’єктів. Частина 1: загальні положення (ISO/IEC 9798-1) [Электронный ресурс].– Режим доступа: http://lindex.net.ua/shop/bibl/500/doc/10491.

12. ДСТУ ISO/IEC 9798-2 Методи захисту. Автентифікація об’єктів. Частина 2: механізми, що ґрунтуються на використанні алгоритмів симетричного шифрування. (ISO/IEC 9798-2) [Электронный ресурс].– Режим доступа: http://lindex.net.ua/shop/bibl/500/doc/10491.

13. ДСТУ ISO/IEC 9798-3 «Інформаційні технології. Методи захисту. Автентифікація об’єктів. Частина 3: Механізми, що ґрунтуються на використанні алгоритмів цифрового підпису». [Электронный ресурс].– Режим доступа: http://lindex.net.ua/shop/bibl/500/doc/10491.

14. ДСТУ ISO/IEC 9798-4 Інформаційні технології. Методи захисту. Автентифікація об’єктів. Частина 4: протоколи, що ґрунтуються на використанні функцій обчислення криптографічного контрольного значення. (ISO/IEC 9798-4) [Электронный ресурс].– Режим доступа: http://lindex.net.ua/shop/bibl/500/doc/10491.

15. ДСТУ ISO/IEC 9798-5 Інформаційні технології. Методи захисту. Автентифікація об’єктів. Частина 5: протоколи, що використовують методи які грунтуються на нульових знаннях. (ISO/IEC 9798-5) [Электронный ресурс].– Режим доступа: http://lindex.net.ua/shop/bibl/500/doc/10491.

16. ISO/IEC 10118-1. Information technology – Security techniques – Hash-functions – Part 1: General. [Электронный ресурс].– Режим доступа: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber.

17. ISO/IEC 10118-2. Information technology – Security techniques – Hash-functions – Part 2: Hash-functions using an n-bit block cipher. [Электронный ресурс].– Режим доступа: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm.

18. ISO/IEC 10118-3 Information technology – Security techniques – Hash-functions – Part 3: Dedicated hash-functions. [Электронный ресурс] .– Режим доступа: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm.

19. ISO/IEC 10118-4 Information technology – Security techniques – Hash-functions – Part 4: Hash-functions using modular arithmetic. [Электронный ресурс].– Режим доступа: http://www.iso.org/iso/iso_catalogue/catalogue_tc/ catalogue_detail.htm.

20. Performance of Optimized Implementations of the NESSIE Primitives. NESSIE, Performance Evaluation. V 2.0, Feb 20, 2003. [Электронный ресурс].– Режим доступа: https://www.cosic.esat.kuleuven.be/nessie/deliverables/D21-v2.pdf

21. Federal Register Notice published on November 2, 2007.– [Электронный ресурс].– Режим доступа: http://csrc.nist.gov/groups/ST/hash.

22. Горбенко Ю.І. Мета, стан та попередні підсумки проекту SHA-3 / Ю.І.Горбенко, А.В.Бойко, А.М. Герцог // Прикладная радиоэлектроника. – 2009. –Т. 8, № 3. – С. 315–321.

23. Hattersley B. NIST SHA-3 Competition Waterfall Hash Algorithm Specification and Analysis [Электронный ресурс] / B.Hattersley //.–2010. – Режим доступа: http://ehash.iaik.tugraz.at/uploads/1/19/Waterfall_Specification.

24. Bellare M. New Paradigm for Collision-free Hashing: Incrementality at Reduced Cost [Электронный ресурс] / M.Bellare, D.Micciancio //. –2009. – Режим доступа: http://groups.csail.mit.edu/cis/incremental/inchash.pdf.

25. Sarkar P. A Parallelizable Design Principle for Cryptographic Hash Functions [Электронный ресурс] / P.Sarkar, P.J.Shellenberg //. –2002. – Режим доступа: http://eprint.iacr.org/2002/031.ps.

26. Fleischmann E. Classification of the SHA-3 Candidates [Электронный ресурс] / E.Fleischmann, C.Forler, M.Gorski //. –2008. – Режим доступа: http://eprint.iacr.org/2008/511.pdf.

27. NIST Computer security resource center. Announcing request for Candidate algorithm nominations for a new cryptographic hash algorithm nominations for a new cryptographic hash algorithm (SHA-3) family. [Электронный ресурс]. –2007. – Режим доступа :http://csrc.nist.gov/groups/ST/.

28. Симмонс Г.Д. Обзор методов аутентификации информации / Г.Д.Симмонс // ТИИЭР. -1988. - т.76, № 5. -С. 105-125.

29. Carter J. L. Universal classes of hash functions / J. L.Carter, M.N.Wegman // Journal of Computer and Systems Science. -1979. - V.18. -P.143-154

30. Wegman. M. N. New hash functions and their use in authentication and set equality / M.N.Wegman, J.L.Carter // Journal of Computer and Systems Science.– 1981. – V. 22. – P. 265–279.

31. Kabatianskii G. On the Cardinality of Systematic Authentication Codes Via Error-Correcting Codes / G.Kabatianskii, B.Smeets, T.Johansson //IEEE Transactions on Information Theory. – 1991. - Vol. IT-42. –P.583-602.

32. Bierbrauer J. On families of hash functions via geometric codes and concatenation. / J.Bierbrauer, T.Johansson, G.Kabatianskii, B.Smeets //Advances in Cryptology-CRYPTO ’93 Proceedings, Springer-Verlag.-1994.- P. 331-342.

33. Bierbrauer J. Universal hashing and geometric codes, to appear in Designs/ J.Bierbrauer //Designs, Codes and Cryptography. -1997. -N.11.-P.207-221.

34. Bierbrauer J. Authentication via algebraic-geometric codes / J.Bierbrauer // Recent Progressin Geometry, Supplemento ai Rendiconti del Circolo Matematico di Palermo. -1998. N.51. –P.139- 152.

35. Milne J.S. Algebraic Geometry / J.S.Milne. –Springer, 2003. – P.206.

36. Simmons G.J. Authentication theory/coding theory, in Advances in Cryptology / G.J.Simmons // Proceedings of Crypto 84, Lecture Notes in Computer Science. - 1985.- V.196. -P.411-431.

37. Preneel B. «Cryptographic primitives for information authentication state of the art» in State of the Art in Applied Cryptography / B.Preneel, V.Rijmen // in Lecture Notes in Computer Science, Springer-Verlag. -1998. -No.1528. -P.50-105.

38. Petrank E. «CBC MAC for real-time data sources» /E. Petrank, C.Rackoff // Journal of Cryptology. - 2000. - Vol. 13, no. 3, - P. 315–338.

39. Bellare M. Keying hash functions for message authentication / M.Bellare, R.Canetti, H.Krawczyk // Advances in Cryptology, Proceedings Crypto'96, LNCS 1109, N. Koblitz, Ed., Springer-Verlag. -1996. - P.1-15.

40. Van Rompay B. TTMAC / B.Van Rompay, B.Den Boer // Primitive submitted to NESSIE. - 2000. -Sept. -P. 153-157.

41. Горбенко И.Д. TWO-TRACK-MAC алгоритм высокой защиты / И.Д.Горбенко, О.Г.Халимов //Радиотехника. Тематический выпуск «Информационная безопасность». Харьков. – 2005. – Вып. 141. – С. 161-167.

42. Black, J. UMAC: Fast and secure message authentication. / J.Black, S.Halevi, H.Krawczyk, T.Krovetz and P.Rogaway //In Advances in Cryptology ‘CRYPTO '99, of Lecture Notes in Computer Science, Springer-Verlag. -1999. - vol. 1666. - P. 216-233.

43. Krovetz T. UMAC / T.Krovetz, J.Black, S.Halevi, A.Hevia, H.Krawczyk and P.Rogaway // Primitive submitted to NESSIE. - 2000. -Sept. -P. 157-160.

44. Халимов Г.З. Высокоскоростной UMAC алгоритм / Г.З. Халимов // Правове, нормативне та метрологічне забезпечення систем захисту інформації в Україні.– НТУУ “КПІ” МОНУ, ДСТСЗІ СБУ. -2005. – Вип.11.-С.167-173.

45. Krovetz T. Fast universal hashing with small keys and no preprocessing: The PolyR construction. / T.Krovetz, P.Rogaway // In Information Security and Cryptology – ICICS 2000. Springer-Verlag.– 2000. –P.73–89.

46. Krovetz T. Message Authentication on 64-bit Architectures [Электронный ресурс] / T.Krovetz //Department of Computer Science, California State University, Sacramento. Report Series: CA 95819 USA http://tdk@acm.org. -2005. -P. 11.

47 Халимов Г.З. Асимптотические границы вероятности коллизии для МАС с алгебраическим кодированием / Г.З.Халимов, А.В.Дунь // Восточно-европейский журнал передовых технологий. –Х.,-2007. -вып. 5/2(29).–С.23-26

48. Stinson D.R. Combinatorial techniques for universal hashing / D.R.Stinson //Journal of Computer and Systems Science.- 1994.- V. 48.- P. 337-346.

49. Stinson D.R. Universal hashing and authentication codes. / D.R.Stinson // Designs, Codes and Cryptography. – 1994. – N.4. - P.369–380.

50. Халимов Г.З. Аутентификация и универсальное хеширование/ Г.З.Халимов, А.А.Кузнецов // Радиотехника. Всеукр. межвед. науч.-техн. сб. – 2001. -Вып. 119. –С. 88-94.

51. Халимов Г.З. Двухкаскадное универсальное хеширование с использованием АГ кодов / Г.З.Халимов, А.Ю.Иохов // Восточно-европейский журнал передовых технологий. –Х., -2005. – Вып. 2/2 (14). –С. 115-119.

52. Mukhopadhyay A.L. Construction of some series of ortogonal array / A.L. Mukhopadhyay //Sankya B43.- 1981. -P. 81-92.

53 Bierbrauer J. Bounds on orthogonal arrays and resilient functions / J.Bierbrauer //Journal of Combinatorial Designs. -1995. –N.3.- P.- 179-183.

54. Bierbrauer J. Weakly biased arrays, almost independent arrays and error-correcting codes / J.Bierbrauer, H.Schellwat //Publication in Proceedings of AMS-DIMACS, 2000. - P.33.

1. Лисицкая Ирина Викторовна. Методология оценки стойкости блочных симметричных криптопреобразований на основе уменьшенных

моделей: дисс. … докт. тех. Наук : 05.13.05 / Лисицкая Ирина Викторовна. – 2012. – 293 с. – Библиогр. : 294–293.

2. И. Моисеенков. Безопасность компьютерных систем. Компьютер пресс. Совместное советско-американское предприятие «СОВАМИНКО». – 1991. – №11 – С. 7-21.

3. Архитектура, протоколы и тестирование открытых информационных сетей. Под ред. Э.Я. Якубайтиса. М.: «Финансы и статистика». - 1990. - С. 70-78.

4. Закон України ″Про електронний цифровий підпис″ [електронний ресурс]: (Відомості Верховної Ради України (ВВР), 2003, № 36, стр. 276) (Із ВВР, 2009, № 24, с. 296), - Режим доступу: http://zakon.rada.gov.ua/cgi-bin/laws/main.cgi?nreg=852-15.

5. Шнаер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. – М:. Триумф. – 2002. – 727 с.

6. National Bureau of Standards. Announcing the data encryption standard. Tech. Report FIPS Publication 46. - 1977.

7. ГОСТ 28147-89. Системы обработки информации. Защита криптографическая. Алгоритмы криптографического преобразования. Введен 01.01.89. - М.: Изд-во стандартов. – 1989. – 78 с.

8. Lai, J. Massey, and S. Murphy, ″Markov Ciphers and Differential Cryptanalysis," Advances in Cryptology - CRYPTO '91, Springer-Verlag. – 1991. – pp. 17-38

9. Diffi W., Hellmann M.E. New Directions in Cryptography // IEEE Transactions on Information Theory. - 1976. - vol. IT-22. - N6. - P. 644-654.

10. NIST FIPS PUB XX. Digital signature standard. Technical report, National Institutes of Standards and Technology, U.S. Department of Commerce, DRAFT, 1 Feb 1993.

11. NIST FIPS PUB 180. Secure hash standard. Technical report, National

Institutes of Standards and Technology, U.S. Department of Commerce, DRAFT, Apr 1993.

12. ГОСТ Р 34.10-94. Информационная технология. Криптографическая защита информации. Процедура выработки и проверки электронной цифровой подписи на базе асимметричного алгоритма; Введ. 01.01.95. - М.: Гос. стандарт РФ, 1994. - 14 с.
13. ГОСТ Р 34.11-94. Информационная технология. Криптографическая защита информации. Функция Хеширования; Введ. 01.01.95. - М.: Гос. стандарт РФ. – 1994. – 14 с.
14. R.L. Rivest, "The RC5 Encryption Algorithm" Fast Software Encryption, Second International Workshop Proceedings, Springer-Verlag, 1995, pp. 86-96.
15. К. Шеннон. Теория связи в секретных системах. Работы по теории информации и кибернетики. Изд. иностр. лит., – 1963. – С. 333-402.
16. Рябко С. Технология SKIP / С. Рябко, Н. Царёв // Открытые системы. – 1996. – №3. – С. 38-41.
17. ISO/IEC 18033-2:2006(E) Інформаційна технологія – Методи захисту – Алгоритми шифрування.
18. Горбенко И.Д. Система защиты информации в электронной почте национального банка Украины: Программное обеспечение системы защиты в электронной почте Национального Банка Украины. Шифр ″Щит-ЭП″. Отчет о НИР (итоговый отчет.) / И.Д. Горбенко, Е.Г. Качко, В.И. Долгов и др.// АО ИИТ - ХТУРЭ, - Т.1-4. Инв. № 2 / нио – Харьков, 1996.
19. Горбенко И.Д. Система защиты информации в ССОИ НАКУ КА, шифр ″Центр-3″: Отчет о НИР / И.Д. Горбенко, Е.Г. Качко, Ю.В. Стасев и др. // АО ИИТ - ХТУРЕ, - Т.1-5. Инв. № 3 / нио - Харьков, 1995.
20. X. Lai. On the design and security of block ciphers, volume 1 of ETH Series in Information Processing. Hartung-Gorre Verlag, 1992.

21. X. Lai. A proposal for a new block encryption standard. / X. Lai and J. Massey. // In I. Damg˚ard, editor, Advances in Cryptology - EUROCRYPT'90, volume 473 of Lecture Notes in Computer Science, pages 389–404. Springer-Verlag, 1991.
22. Susan Landau. Polynomials in the Nation's Service: Using Algebra to Design the Advanced Encryption Standard, February, 2004.
23. W. Meier and O. Staffelbach, Nonlinearity criteria for cryptographic functions, in Advances in Cryptology: Eurocrypt '89, J.-J Quisquater and J. Vandewalle, eds., Springer-Verlag, Berlin, 1989.
24. J. Pieprzyk, Nonlinearity of exponent permutations, in Advances in Cryptology: Eurocrypt '89, J.-J Quisquater and J. Vandewalle, eds., Springer-Verlag, Berlin, 1990, pp. 89-92.
25. J. Pieprzyk, On Bent Permutations, Technical Report CS91/11, Department of Computer Science, University of New South Wales; presented at International Conference on Finite Fields, Coding Theory, and Advances in Communications and Computing, Las Vegas, 1991.
26. K. Nyberg, Differentially uniform mappings for cryptography, in Advances in Cryptology: Eurocrypt '93, T. Helleseth, ed., Springer-Verlag, Berlin, 1994, pp. 53-64.
27. J. Daemen, Cipher and Hash Function Design Strategies Based on Linear and Differential Cryptanalysis, Ph.D. thesis, Katholieke Universiteit, Leuven, Belgium, 1995.
28. V. Ridjmen, J. Daemen, B. Preneel, A. Bosselaers, and E. De Win, The cipher SHARK, in Fast Software Encryption: Third International Workshop, D. Gollman, ed., Springer-Verlag, Berlin, 1996, pp. 99-112.
29. T. Jakobsen and L. Knudsen, Attacks on block ciphers of low algebraic degree, J. Cryptology 14 (2001), pp. 197-210.
30. J. Daemen, L. Knudsen, and V. Ridjmen, The Block Cipher Square, in Fast Software Encryption, E. Biham ed., LNCS 1267, Springer-Verlag, Berlin, 1997.

31. J. Daemen and V. Ridjmen. The Design of Rijndael: AES — the Advanced Encryption Standard, Springer-Verlag, Berlin, 2002.
32. R. Lidl and H. Niederreiter, Introduction to Finite Fields and Their Applications, Cambridge University Press, Cambridge, 1986.
33. B. Weeks, M. Bean, T. Rozylowicz, and C. Ficke, Hardware Performance Simulation of Round 2 Advanced Encryption Standard Algorithms (May 15, 2000); available at http://csrc.nist.gov/encryption/aes/ round2/r2anlsys.htm.
34. P. Junod and S. Vaudenay, FOX: a new family of block ciphers. In H. Handschuh and A. Hasan, editors, Selected Areas in Cryptography: 11th International Workshop, SAC 2004, Waterloo, Canada, August 9-10, 2004. Revised Selected Papers, volume 3357 of Lecture Notes in Computer Science, pages 114-129. Springer-Verlag, 2004.
35. Горбенко І.Д. Принципи побудування та властивості блокових симетричних IDEA подібних шифрів / І.Д. Горбенко, В.І. Долгов, Р.В. Олійников, В.І. Руженцев, М.С. Михайленко, Ю.І. Горбенко, П.О. Колесніков // Прикладная радиоэлектроника. – Харьков: ХНУРЭ. – 2007. Том. 6, № 2, С. 158-173.
36. P. Junod, FOX specifications version 1.1. / P. Junod and S. Vaudenay. Technical Report EPFL/IC/2004/75, Ecole Polytechnique F´ed´erale, Lausanne, Switzerland, 2004.
37. Daemen, J. ‘Understanding two-round differentials in AES’./ Daemen, J., and Rijmen, V.: / Proc. Security and Cryptography for Networks (SCN 2006), LNCS, 4116, 2006, edited by De Prisco, R., and Yung, M., (Springer), pp. 78–94.
38. Долгов В.И. Дифференциальные свойства блочных симметричных шифров, представленных на украинский конкурс. / В.И. Долгов, А.А. Кузнецов, С.А. Исаев. // Электронное моделирование. – 2011.– Т 33, № 6. – С. 81-99.

39. Кузнецов А.А. Линейные свойства блочных симметричных шифров, представленных на украинский конкурс. / А.А. Кузнецов, И.В. Лисицкая, С.А. Исаев // Прикладная радиоэлектроника. − 2011. − Т.10, №2 − С. 135-140.

40. Лисицкая И.В. 32-х битная мини-версия блочного симметричного алгоритма криптографического преобразования информации ″Мухомор″. Оценка максимального значения полного дифференциала шифра. / И.В. Лисицкая, И.Ф. Ставицкий // Научные ведомости Белгородского государственного университета – 2011. – № 7 (102). – Вып. 18/1 – С. 177-186.

41. Лисицкая И.В. О новой методике оценки стойкости блочных симметричных шифров к атакам дифференциального и линейного криптоанализа. Системы обработки информации. - Харьковский университет Противовоздушных Сил имени Ивана Кожедуба, – 2011. – Вып. 4(94). – С. 167-173.

42. Горбенко І.Д. Новая идеология оценки стойкости блочных симметричных шифров к атакам дифференциального и линейного криптоанализа. / І.Д. Горбенко, В.И. Долгов, И.В. Лисицкая, Р.В. Олейников // Прикладная радиоэлектроника. – 2010. – Т. 9, № 3. – С. 212-320.

43. Олейников Р.В. Дифференциальные свойства подстановок / Р.В. Олейников, О.И. Олешко, К.Е. Лисицкий, А.Д. Тевяшев // Прикладная радиоэлектроника. − 2010. − Т.9. − № 3. − С. 326-333.

44. Долгов В.И. Свойства таблиц линейных аппроксимаций случайных подстановок. / В.И. Долгов, И.В. Лисицкая, О.И. Олешко // Прикладная радиоэлектроника. – Харьков: ХНУРЭ. – 2010. – Т. 9, № 3. – С. 334-340.

45. NESSIE security report // http:// www. Cryptonessie. Org / NES/DOC/ENS/WPS/D20/2, 2003.

46. Молдавян А.А. и др. Криптография: скоростные шифры. – СПб.: БХВ-Петербург – 2002. – 496 с.

47. Положення про проведення відкритого конкурсу криптографічних алгоритмів. http://dstszi.gov.ua/dstszi/control/uk/publish/, 2006.

48. Горбенко І.Д. Перспективний блоковий симетричний шифр "Калина" – основні положення та специфікації. / І.Д. Горбенко, В.І. Долгов, Р.В. Олейніков, В.І. Руженцев, М.С. Михайленко, Ю.І. Горбенко, О.С. Тоцкій, С.В. Казьміна // Прикладна радіоелектроніка. – 2007. – Т.6. – № 2. – С. 195-208.

49. Горбенко І.Д. Перспективний блоковий симетричний шифр «Мухомор» – основні положення та специфікація / І.Д. Горбенко, М.Ф. Бондаренко, В.І. Долгов, Р.В. Олійников, В.І. Руженцев, М.С. Михайленко , Ю.І. Горбенко, О.І. Олешко, С.В. Казьміна // Прикладная радиоэлектроника. – Харьков: ХТУРЭ. – 2007. – Том. 6, №2. – С. 147-157.

50. Головашич С.А. Спецификация алгоритма блочного симметричного шифрования «Лабиринт» // Прикладная радиоэлектроника. – Харьков: ХТУРЭ. – 2007. – Том. 6, №2. – С. 230-240.

51. Кузнецов А.А. Симметричный криптографический алгоритм ADE (Algorithm of Dynamic Encryption). / А.А. Кузнецов, Р.В. Сергиенко, А.А. Наумко // Прикладная радиоэлектроника. – Харьков: ХТУРЭ. – 2007. – Том 6, №2 – С. 241-249.

52. Nicolas Courtois. General Principles of Algebraic Attacks and New Design Criteria for Components of Symmetric Ciphers. In H. Dobbertin, V. Rijmen, and A. Sowa, editors, Fourth Conference on the Advanced Encryption Standard - AES4, volume 3373 of Lecture Notes in Computer Science, pages 67-83. Springer-Verlag, 2004.

53. Горбенко І.Д. Принципи побудування та властивості блокових симетричних IDEA подібних шифрів. / І.Д. Горбенко, В.І. Долгов, Р.В.

Олійников, В.І. Руженцев, М.С. Михайленко, Ю.І. Горбенко, П.О. Колесников // Прикладная радиоэлектроника. − Харьков: ХНУРЭ. − 2007. – Том 6, № 2. – С. 158-173.

54. Горбенко И.Д. Сравнительный анализ блочных симметричных шифров, представленных в проекте NESSIE. / І.Д. Горбенко, С.А. Головашич, А.И. Лепёха // Радиотехника: Всеукр. межвед. научн.-техн. сб. − 2003. – Вип.134. – С. 26-40.
55. Горбенко И.Д. Стандартизация алгоритмов шифрования. Требования к проекту национального стандарта блочного симметричного шифрования на современном этапе развития криптографии. / И.В. Горбенко, И.В. Лисицкая // Межведомственный научн. технический сборник ″Радиотехника″. – 2011. – вып. 166. – С. 5-10.
56. Долгов В.И. Подход к криптоанализу современных шифров. / В.И. Долгов, И.В. Лисицкая, Р.В. Олейников // Материалы второй международной конференции ″Современные информационные системы. Проблемы и тенденции развития″, Харьков-Туапсе, Украина, 2–5 октября. – 2007. – С. 435-436.
57. Лисицкая И.В. О новой идеологии оценки стойкости блочных симметричных шифров к атакам дифференциального и линейного криптоанализа, Красноярск, 2011. Сборник материалов I-ой Всероссийской электронной научно-практической конференции-форума молодых ученых и специалистов ″Современная российская наука глазами молодых исследователей, Красноярск - 2011″, февраль. – 2011. – С. 118-120.
58. L. Keliher, Linear Cryptanalysis of Substitution-Permutation Networks. A thesis submitted to the School of Computing in conformity with the requirements for the degree of Doctor of Philosophy, 2003, 160 p.

59. H. Feistel, Cryptography and computer privacy, Scientific American, Vol. 228, No. 5, pp. 15–23, May 1973.

60. H. Feistel, W.A. Notz, and J.L. Smith, Some cryptographic techniques for machine to machine data communications, Proceedings of the IEEE, Vol. 63, No. 11, pp. 1545-1554, November 1975.

61. A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997.

62. E. Biham, New types of cryptanalytic attacks using related keys, Advances in Cryptology—EUROCRYPT'93, LNCS 765, pp. 398–409, Springer-Verlag, 1994.

63. J. Daemen and V. Rijmen, the Design of Rijndael: AES − The Advanced Encryption Standard, Springer-Verlag, 2002.

64. K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita, Camellia: A 128-bit block cipher suitable for multiple platforms—design and analysis, Seventh Annual International Workshop on Selected Areas in Cryptography (SAC 2000), LNCS 2012, pp. 39–56, Springer-Verlag, 2001.

65. B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, and N. Ferguson, The Twofish Encryption Algorithm: A 128-Bit Block Cipher, John Wiley and Sons, 1999.

66. R. Anderson, E. Biham, and L. Knudsen, Serpent: A flexible block cipher with maximum assurance, The First Advanced Encryption Standard Candidate Conference, Proceedings, Ventura, California, August 1998.

67. E. Biham, On Matsui's linear cryptanalysis, Advances in Cryptology − EUROCRYPT'94, LNCS 950, pp. 341-355, Springer-Verlag, 1995.

68. K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita, Camellia: A 128-bit block cipher suitable for multiple platforms—

design and analysis, Seventh Annual International Workshop on Selected Areas in Cryptography (SAC 2000), LNCS 2012, pp. 39–56, Springer-Verlag, 2001.

69. M. Kanda, Practical security evaluation against differential and linear cryptanalysis for Feistel ciphers with SPN round function, Seventh Annual International Workshop on Selected Areas in Cryptography (SAC 2000), LNCS 2012, pp. 324-338, Springer-Verlag, 2001.

70. M. Kanda, S. Moriai, K. Aoki, H. Ueda, M. Ohkubo, Y. Takashima, K. Ohta, and T. Matsumoto, E2 − A candidate cipher for AES, The First Advanced Encryption Standard Candidate Conference, Proceedings, Ventura, California, August 1998.

71. B. Schneier and J. Kelsey, Unbalanced Feistel networks and block-cipher design, Fast Software Encryption: Third International Workshop, LNCS 1039, pp. 121-144, Springer-Verlag, 1996.

72. C. Adams, The CAST-256 encryption algorithm, The First Advanced Encryption Standard Candidate Conference, Proceedings, Ventura, California, August 1998.

73. C. Burwick, D. Coppersmith, E. D'Avignon, R. Gennaro, S. Halevi, C. Jutla, S. Matyas, Jr., L. O'Connor, M. Peyravian, D. Safford, N. Zunic, MARS − a candidate cipher for AES, The First Advanced Encryption Standard Candidate Conference, Proceedings, Ventura, California, August 1998.

74. X. Lai, J. Massey, and S. Murphy, Markov ciphers and differential cryptanalysis, Advances in Cryptology—EUROCRYPT'91, LNCS 547, pp. 17-38, Springer-Verlag, 1991.

75. R. Rivest, M. Robshaw, R. Sidney, and Y. Lin, The RC6 block cipher, The First Advanced Encryption Standard Candidate Conference, Proceedings, Ventura, California, August 1998.

76. N. Ferguson, R. Schroeppel, and D. Whiting, A simple algebraic representation of Rijndael, Eighth Annual International Workshop on Selected Areas in Cryptography (SAC 2001), LNCS 2259, pp. 103-111, Springer-Verlag, 2001.

77. L. Keliher, H. Meijer, and S. Tavares, Toward the true random cipher: On expected linear probability values for SPNs with randomly selected s-boxes, chapter in Communications, Information and Network Security, V. Bhargava, H. Poor, V. Tarokh, and S. Yoon (Eds.), pp. 123-146, Kluwer Academic Publishers, 2003.

78. Вербицький О.В. Вступ до криптології. Видавн. Науково-технічної літератури. Львів, 1998, 247 с.

79. Лисицкая И.В. О результатах применения новой методики ускоренного криптоанализа блочных симметричных шифров. Спеціальні телекомунікаційні системи та захист інформації, Київ. – 2011. – Вип. 1 (19). – С. 23-30.

80. Jorge Nakahara, Cryptanalysis and Desisgn of Block Siphers. Preisschrift vorgetragen tot hat behalen van hat Doktorats in de toegepaste wetenschappen Door Jorge NAKAHARA Junior, 2003, p. 235.

81. AES. The Advanced Encryption Standard Development Process. http://csrc.nist.gov/encryption/aes/, 1997.

82. C.M. Adams. The CAST-256 Encryption Algorithm. 1st AES Conference, California, USA, Aug 1998. http://csrc.nist.gov/encryption/aes/.

83. H. Lipmaa, P. Rogaway, and D. Wagner. CTR-Mode Encryption. 2nd NIST Modes of Operation Workshop, Aug 2001.

http://csrc.nist.gov/CryptoToolkit/modes/.

84. L.R. Knudsen. DEAL – a 128-bit Block Cipher. Tech Report #151, University of Bergen, Dept. of Informatics, Norway, Feb 1998.

85. H. Gilbert, M. Girault, P. Hoogvorst, F. Noilhan, T. Pornin, G. Poupard, J. Stern, and S. Vaudenay. Decorrelated Fast Cipher: an AES candidate. 1st AES Conference, California, USA, Aug 1998. http://csrc.nist.gov/encryption/aes/.

86. Corp NTT. Specification of E2 – a 128-bit Block Cipher. 1st AES Conference, California, USA, Jun 1998. http://csrc.nist.gov/encryption/aes/.

87. D.G. Georgoudis, D. Leroux, and B.S. Chaves. FROG. 1st AES Conference, California, USA, Aug 1998. http://csrc.nist.gov/encryption/aes/.

88. R. Schroeppel. Hasty Pudding Cipher Specification. 1st AES Conference, California, USA, Aug 1998. http://csrc.nist.gov/encryption/aes/.

89. L. Brown and J. Pieprzyk. Introducing the New LOKI97 Block Cipher. 1st AES Conference, California, USA, Aug 1998.

http://csrc.nist.gov/encryption/aes/.

90. M.J. Jacobson and K. Huber. The MAGENTA Block Cipher Algorithm. 1st AES Conference, California, USA, Aug 1998.

http://csrc.nist.gov/encryption/aes/.

91. R.L. Rivest, M.J.B. Robshaw, R. Sidney, and Y.L. Yin. The RC6 Block Cipher. 1st AES Conference, California, USA, Aug 1998.

http://csrc.nist.gov/encryption/aes/.

92. J. Daemen and V. Rijmen. AES Proposal: Rijndael. 1st AES Conference, California, USA, 1998. http://www.nist.gov/aes.

93. J.L. Massey, G.H. Khachatrian, and M.K. Kuregian. Nomination of SAFER+ as Candidate Algorithm for the Advanced Encryption Standard. 1st AES Conference, California, USA, Jun 1998.

http://csrc.nist.gov/encryption/aes/.

94. E. Biham, R. Anderson, and L.R. Knudsen. Serpent: A New Block Cipher Proposal. In S. Vaudenay, editor, 5th Fast Software Encryption Workshop, LNCS 1372, pages 222-238. Springer-Verlag, 1998.

95. B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, and N. Ferguson. Twofish: a 128-bit Block Cipher. 1st AES Conference, California, USA, Aug 1998. http://csrc.nist.gov/encryption/aes/.

96. J. Nechvatal, E. Barker, L. Bassham, W. Burr, M. Dworkin, J. Foti, and E. Roback. Report on the Development of the Advanced Encryption Standard. Computer Security Division, Information Technology Laboratory, NIST, Oct 2000.

97. NESSIE. New European Schemes for Signatures, Integrity and Encryption, Jan 2000. http://cryptonessie.org

98. O. Goldreich, Foundations of Cryptography | Basic Tools, vol. 1. Cambridge University Press, Aug. 2001. [p. 10, 314].

99. E. Biham and A. Shamir. Differential Cryptanalysis of DES-like Cryptosystems. Journal of Cryptology, 4(1): 3-72, 1991.

100. X. Lai. On the Design and Security of Block Ciphers, ETH Series in Information Processing // Lecture Notes in Computer Science. — Berlin / Heidelberg: Springer-Verlag, 10 April 2006 — T. 1179/1996. — P. 213-222. — ISBN 978-3-540-62031-0.

101. X. Lai. Higher Order Derivatives and Differential Cryptanalysis. In Proc. of Symp. on Communication, Coding and Cryptography, Monte Verita, Switzerland, pages 227-233, Feb 1994.

102. L.R. Knudsen. Truncated and Higher Order Differentials. In B. Preneel, editor, 2nd Fast Software Encryption Workshop, LNCS 1008, pages 196–211. Springer-Verlag, 1995.

103. J. Daemen, L.R. Knudsen, and V. Rijmen. The Block Cipher SQUARE. In E. Biham, editor, 4th Fast Software Encryption Workshop, LNCS 1267, pages 149-165. Springer-Verlag, 1997.

104. E. Biham, A. Biryukov, and A. Shamir. Cryptanalysis of Skipjack Reduced to 31 Rounds using Impossible Differentials. Tech Report CS0947 revised, Teknion, CS Dept., 1998.

105. D. Wagner. The Boomerang Attack. In L.R. Knudsen, editor, 6th Fast Software Encryption Workshop, LNCS 1636, pages 156-170. Springer-Verlag, 1999.

106. J. Kelsey, T. Kohno, and B. Schneier. Amplified Boomerang Attacks against Reduced-Round MARS and Serpent. In B. Schneier, editor, 7th Fast Software Encryption Workshop, LNCS 1978, pages 75-93. Springer-Verlag, 2000.

107. E. Biham, O. Dunkelman, and N. Keller. The Rectangle Attack – Rectangling the Serpent. In B. Pfitzmann, editor, Advances in Cryptology, Eurocrypt'01, LNCS 2045, pages 340-357. Springer-Verlag, 2001.

108. D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. Journal of Symbolic Computation, 9(3):251-280, 1990.

109. Mitsuru Matsui, Linear Cryptanalysis Method for DES Cipher. 386-397.

110. Ferguson, Niels, and Kelsey, John, and Lucks, Stefan, and Schneier, Bruce, and Stay, Mike, and Wagner, David, and Whiting, Doug. 2000. Improved Cryptanalysis of Rijndael.

111. K. Nyberg, Linear approximation of block ciphers, Advances in Cryptology − Eurocrypt'94, Lecture Notes in Computer Science, vol. 950, Springer-Verlag, 1994.

112. L. O'Connor, "Convergence in Differential Distributions", Advances in Cryptology, Proceedings of Eurocrypt '95, LNCS 921, L. Guillou and J. Quisquater, Eds., Springer-Verlag 1995, pp.13-23.

113. S. Vaudenay, “On the Security of CS-Cipher,” Fast Software Encryption ’99, LNCS 1636, L. Knudsen, Ed., Springer-Verlag, 1999, pp. 260-274.

114. K. Nyberg and L.R. Knudsen, “Provable security against a differential attack”, Journal of Cryptology, Volume 8, No 1, 1995, pp. 27-38.

115. K. Aoki and K. Ohta, “Strict evaluation of the maximum average of differential probability and the maximum average of linear probability”, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences (Japan), Vol. E80-A, No 1, 1997, pp. 2-8.

116. M. Matsui, “New block encryption algorithm Misty”, Fast Software Encryption ’97, LNCS 1267, E. Biham, Ed., Springer-Verlag, 1997, pp. 64-74.

117. L. Keliher, H. Meijer, and S. Tavares, “New method for upper bounding the maximum average linear hull probability for SPNs,” Advances in Cryptology, Proceedings of Eurocrypt ’01, LNCS 2045, B. Pfitzmann, Ed., Springer-Verlag, 2001, pp. 420-436.

118. L. Keliher, H. Meijer, and S. Tavares, “Improving the upper bound on the maximum average linear hull probability for Rijndael”, Advances in Cryptology, Selected Areas in Cryptography ’01, LNCS 2259, S. Vaudenay, A.M. Youssef, Eds., Springer-Verlag, 2001, pp. 112-128.

119. S. Park, S.H. Sung, S. Chee, E-J. Yoon and J. Lim, “On the security of Rijndael-like structures against differential and linear cryptanalysis,” Advances in Cryptology, Proceedings of Asiacrypt ’02, LNCS 2501, Y. Zheng, Ed., Springer-Verlag, 2002, pp. 176-191.

120. S. Park, S.H. Sung, S. Lee and J. Lim, “Improving the upper bound on the maximum differential and the maximum linear hull probability for SPN structures and AES,” Fast Software Encryption ’03, LNCS 2887, T. Johansson, Ed., Springer-Verlag, 2003, pp. 247-260.

121. S. Vaudenay, “Resistance Against General Iterated Attacks,” Advances in Cryptology, Proceedings of Eurocrypt ’99, LNCS 1592, J. Stern, Ed., Springer-Verlag, 1999, pp. 255-271.

122. S. Vaudenay, “Decorrelation: a theory for block cipher security,” Journal of Cryptology, Vol. 16, No. 4, 2003, pp. 249-286.

123. M.N. Wegman, J.L. Carter, “New hash functions and their use in authentication and setequality,” Journal of Computer and System Sciences, Vol. 22, 1981, pp. 265-279.

124. K. Nyberg. Differentially uniform mappings for cryptography. In Advances in cryptology – Proceedings of EUROCRYPT'93 (1994) vol. 765, Lecture Notes in Computer Science Springer-Verlag, Berlin, Heidelberg, New York, pp. 55-65.

125. K. Nyberg. Linear approximation of block ciphers, Advances in Cryptology – Eurocrypt’94, Lecture Notes in Computer Science, vol. 950, Springer-Verlag, 1994.

126. M. Matsui. On a Structure of Block Ciphers with Provable Security against Differential and Linear Cryptanalysis. IEICE Trans/ FundaMENTALS, vol. E82-A, NO. 1 January 1999, pp. 117-122.

127. S. Hong, S. Lee, J. Lim, J. Sung, D. Cheon and I. Cho. Provable Security against Differential and Linear cryptanalysis for SPN Structure. B. Schneier (Ed.): FSE 2000, LNCS 1978, pp. 273-283, 2001.

128. Thomas Baignoires and Serge Vaudenay. Proving the Security of AES Substitution-Permutation Network. http://lasecwww.epfl.ch. 2004. p. 16.

129. Алексийчук А.Н. Оценки практической стойкости блочного шифра ″Калина″ относительно методов разностного, линейного криптоанализа и относительно алгебраических атак, основанных на гомоморфизмах. / А.Н. Алексийчук, Л.В. Ковальчук, Е.В. Скрыпник, А.С. Шевцов // Прикладная радиоэлектроника. – 2008. – Т.7. – №3. – С. 203-209.

130. F. Sano, K. Ohkuma, H. Shimizu, S. Kawamura. On the Security of Nested SPN Cipher against the Differential and Linear Cryptanalysis/ IEICE Trans. Fundamentals, vol. E86-a, NO.1 January 2003, pp. 37-46.

131. E.F. Brickell, J.H. Moore, and M.R. Purtill. Structure in the S-boxes DES. Advances in cryptology, CRYPTOZb, Lecture Notes in Computer Science, vol. 263. A.M. Odlyzko ed., Springer-Verlag, pages 3-8, 1987.

132. C. M. Adams. A formal and practical design procedure for Substitution-Permutation network cryptosystem. PhD thesis, Department of Electrical Engineering, Queen's University at Kingston, 1990.

133. C. M. Adams. And S.E. Tavares. The Structured design of cryptographically good S-boxes. Journal of Cryptology, 3(1): 27-41, 1990.

134. R. Forré. Methods and instruments for designing S-boxes. Journal of Cryptology, 2(3): 115-130,1990.

135. K. Nyberg. Perfect nonlinear S-boxes**.** In Advances in cryptology - EUROCRYPT91, volume 547, Lecture Notes in Computer Science, pp. 378-386. Springer-Verlag, Berlin, Heidelberg, New York, 1991.

136. M. H. Dawson. A unified framework for substitution box design based on information theory. Vaster's thesis, Queen's University, Kingston, Ontario, Canada, 1991.

137. Matsui, M.: Linear cryptanalysis method for DES cipher. In Advances in Cryptology.- EUROCRYPT'93 (1994) vol. 765. Lecture Notes in Computer Science Springer-Verlag, Berlin, Heidelberg, New York pp. 386-397.

138. K. Nyberg and L.R. Knudsen. Provable security against differential cryptanalysis. In Advances in cryptology - EUROCRYPT'92, volume Lecture Notes in Computer Science, Springer-Verlag, Berlin, Heidelberg, New York, 1992, pp. 566-574.

139. T. Beth and C. Ding. On permutations against differential cryptanalysis. In Advances in cryptology - EUROCRYPT'93. Springer-Verlag, Berlin, Heidelberg, New York, 1993.

140. Seberry J., Zhang X.M., Zheng Y. "Pitfalls in Designing Boxes (Extended Abstract)"//, Copyright © Springer-Verlag, 1998, pp. 383-396.

141. Seberry J., Zhang X.M., Zheng Y.: Relationships among nonlinearity criteria. Presented at *EUROCRYPTV4,* 1994.

142. Zhang X.M, Zheng Y, Imai. H.: Non-existence of Certain Quadratic S-boxes and Two Bounds on Nonlinear Characteristics of General S-boxes // October 1997, pp. 1-18.

143. K. Nyberg. On the construction of highly nonlinear permutations. In Advances in cryptology - Proceedings of EUROCRYPT'92 (1993) vol. 740, Lecture Notes in Computer Science Springer-Verlag, Berlin, Heidelberg, New York pp. 92-98.

144. Seberry J., Zhang X.M., Zheng Y. Improving the strict avalanche characteristics of cryptographic functions. Information Processing Letttrs, 50:37-41, 1994.

145. Долгов В.И. Принципы защиты алгоритма DES от атак дифференциального криптоанализа. / В.И. Долгов, И.В. Лисицкая, С.А. Головашич, Р.В. Олейников // Радиотехника. Всеукр. Межвед. Науч.-техн. сб. – 2000. – Вып 113. – С. 148-157.

146. Лисицкая И.В. Уточненные критерии отбора таблиц подстановок с заданными характеристиками случайности. / И.В. Лисицкая, А.С. Коряк // Радиотехника. – 2000. – Вып. 114. – С. 47-56.

147. Долгов В.И. Дополнительные требования к отбору таблиц подстановок для ГОСТ 28147-89. / В.И. Долгов, И.В. Лисицкая, Р.В. Олейников, С.А. Головашич // Радиотехника. Всеукр. Межвед. Науч.-техн. сб. − 2001. − Вып. 119. – С. 153-159.

148 Лисицкая И.В. Исследование возможностей модернизации шифра ГОСТ 28147-89 с целью дальнейшего повышения его безопасности. / И.В. Лисицкая, Т.В. Цепурит, В.В. Лесняк, М.В. Пинчук, А.П. Мелецкий //

Радиотехника. Всеукр. Межвед. Науч.-техн. сб. – 2001. – Вып. 119. – С. 160-165.

149. Лисицкая И.В. Обеспечение стойкости шифра DES к атакам линейного криптоанализа. Требования к отбору S блоков, защищенных от атак на характеристики обнуляющего типа, четырехцикловые и шестицикловые итеративные аппроксимации. / И.В. Лисицкая, А.С. Бондаренко, А.И. Колыбельников // Радиотехника. Всеукр. Межвед. Науч.-техн. сб. – 2001. – Вып. 119. – С. 177-190.

150. Долгов В.И. Обеспечение стойкости шифра DES к атакам дифференциального криптоанализа, перекрытие итеративных характеристик обнуляющего типа и четырёхцикловых итеративных характеристик. / В.И. Долгов, И.В. Лисицкая, В.И. Руженцев // Радиотехника. Всеукр. Межвед. науч.-техн. сб. – 2001. – Вып. 120. – С. 192-197.

151. Олейников Р.В. Исследование подстановок ГОСТ 28147-89, построенных на основе анализа свойств координатных функций. / Р.В. Олейников, И.В. Лисицкая // Труды V-ої Міжнародної науково-практичної конференції Безпека інформації в інформаційно-телекомунікаційних системах, Київ, 20-24 травня. – 2002. – стор. 54.

152. Лисицкая И.В. Отбор таблиц подстановок для алгоритма DES, устойчивых к атакам дифференциального криптоанализа. / И.В. Лисицкая, Р.В. Олейников // Сб. научных трудов 5-ой международной конференция “Теория и техника передачи, приема и обработки информации”, Харьков, 27-30 сентября. – 1999. – С. 348-350.

153. Лисицкая И.В. Критерии отбора таблиц подстановок алгоритма шифрования DES, устойчивых к атакам линейного криптоанализа. / И.В. Лисицкая, С.А. Головашич // Сб. научных трудов. 5-й международной конференции “Теория и техника передачи, приема и обработки

информации”, Харьков, 27-30 сентября, Харьков: ХТУРЭ. – 1999. – С. 351-353.

154. Lisitskaya I.V. The selection criteria of random substitution tables for symmetric enciphering algorithms. / I.V. Lisitskaya, A.S. Koriak., S.A. Golovashich O.I. Oleshko, R.V. Oleinik // Abstracts of XXVIth General Assembly. Toronto, Ontario Canada, August 13-21. – 1999. – P. 204.

155. Лисицкая И.В. Обеспечение стойкости шифра DES к атакам дифференциального и линейного криптоанализа. / И.В. Лисицкая, А.С. Бондаренко, Т.В. Цепурит // Сб. научных трудов 6-ой Международной конференции ″Теория и техника передачи, приема и обработки информации″, Харьков, ХТУРЭ. – 2000. – С. 61-63.

156. Долгов В.И. Стойкость алгоритма шифрования ГОСТ 28147-89 при использовании стандартной таблицы подстановок. / В.И. Долгов, И.В. Лисицкая, Р.В. Олейников, А.И. Шумов // 6-ая Международная конференция ″Теория и техника передачи, приема и обработки информации″, Харьков: ХТУРЭ, – 2000. – С. 59-60.

157. Лисицкая И.В. К вопросу построения долговременных ключей для алгоритма ГОСТ-28147−89. Информационно-управляющие системы на железнодорожном транспорте. – 1997. – №3. – С. 54-57.

158. Бильчук В.М. К вопросу построения долговременных ключей для алгоритма ГОСТ 28147-89. Критерии отбора подстановок второго уровня. / В.М. Бильчук, И.В. Лисицкая //Информационно-управляющие системы на железнодорожном транспорте. – 1998. – №1. – С.10-17.

159. Широков Алексей Викторович. Методы формирования S-блоковых конструкций случайного типа с улучшенными криптографическими показателями (для блочных симметричных шифров с доказуемой безопасностью): дис. … канд. техн. наук : 05.13.21 / Широков Алексей Викторович. – Харьков, 2010. – 265 с. – Библиогр. : с. 215-232.

160. Подстановочные конструкции современных симметричных блочных шифров. / В.И. Долгов, Р.В. Олейников и др. // Радіоелектронні та комп'ютерні системи, Харьков, ХАИ. – 2009. – № 6 (40). – С. 89-93.

161. Лисицкая И.В. Оценка числа случайных подстановок с заданным распределением парных разностей XOR таблиц и смещений таблиц линейных аппроксимаций. / И.В. Лисицкая, А.В. Широков, Е.Д. Мельничук, К.Е. Лисицкий // Прикладная радиоэлектроника. – Харьков: ХНУРЭ. – 2010. – Т. 9, № 3. – С. 341-345.

162. Долгов В.И. Случайные подстановки в криптографии. / В.И. Долгов, И.В. Лисицкая, К.Е. Лисицкий // Радіоелектронні та комп'ютерні системи. – 2010. – № 5 (46). – С. 79-85.

163. Borst. Block Ciphers: Design, Analysis and Side-Channel Analysis. PhD thesis, Dept. Elektrotechniek, Katholieke Universiteit Leuven, Belgium, Sep 2001.

164. Завало С.Т., Костарчук В.Н., Хацет Б.И. Алгебра и теория чисел. / - ч.2. - К.: Вища школа -1980. - 407 с.

165. Сачков В.Н. Введение в комбинаторные методы дискретной математики. - М.: Наука – 1982 – 384с.

166. Сачков В.Н. Комбинаторные методы дискретной математики. - М.: Наука, - 1977. - 319с.

167. Математическая энциклопедия: В 5 т. / Гл. ред. Виноградов И.М. - М.: Советская энциклопедия, 1979. - Т.2: Д-КОО. - 278 с.

168. Сачков В.Н. Вероятностные методы в комбинаторном анализе. - М.: Наука, 1978. - 285 с.

169. Риордан Дж. Введение в комбинаторный анализ: Пер с англ. - М.: ИЛ, 1963. - 287 с.

170. Райзер Г. Дж. Комбинаторная математика: Пер с англ. - М.:ИЛ, 1966. - 97

171. Бронштейн И.Н. Справочник по математике для инженеров и учащихся вузов. / И.Н. Бронштейн, К.А. Семендяев // Изд-во М:. ″Наука″ 1980, 976 стр.

172. Олейников Р. В., Лисицкий К. Е. Исследование дифференциальных свойств подстановок различных цикловых классов. Двенадцатая Международная научно-практическая конференция ″Безопасность информации в информационно-телекоммуникационных системах ″, 19-22 МАЯ 2009 г., Тезисы докладов. – К.: ЧП “ЕКМО”, НИЦ “ТЕЗИС” НТУУ “КПИ”, 2009. – С. 24-25.

173. L.J. O’Connor. On the Distribution of Characteristics in Bijective Mappings. Advances in Cryptology. EUROCRIPT 93, Lecture Notes in Computer Science, vol. 795, T. Helleseth ed., Springer-Verlag, pages 360–370, 1994.

174. Luke O’Connor. Properties of Linear Approximation Tables. Email: oconnor@dsts. Edu. au, 1995.

175. Luke O’Connor. On Linear Approximation Tables and Ciphers secure against Linear Cryptanalysis. Email: oconnor@dsts. Edu. au, 1995. (семь страниц).

176. Долгов В.И. Анализ циклических свойств блочных шифров. / В.И. Долгов, И.В. Лисицкая, В.И. Руженцев // Прикладная радиоэлектроника. – 2007. – Т.6, №2 – С. 257-263.

177. Nicolas T. Courtois, Gregory V. Bard, and Shaun V. Ault. Statistics of Random Permutations and the Cryptanalysis Of Periodic Block Ciphers, J. Math. Crypt. 2 (2008), 1-20.

178. Joan Daemen, Vincent Rijmen Probability distributions of Correlation and Differentials in Block Ciphers. / Joan Daemen, Vincent Rijmen // April 13, 2006, pp. 1–38.

179. Долгов В.И. Rijndael – это новое или хорошо забытое старое. / В.И. Долгов, И.В. Лисицкая, Р.И. Киянчук // Сборник трудов Первой Международной научно-технической

конференции ″Компьютерные науки и технологии″, 8-10 октября 2009, Белгород, Ч. 2, С. 32-35.

180. Криптографические свойства уменьшенной версии шифра ″Мухомор″. / И.В. Лисицкая, О.И. Олешко, С.Н. Руденко и др. // Спеціальні телекомунікаційні системи та захист інформації. Збірник наукових праць, Київ. − 2010. – Вип. 2(18). − С. 33-42.

181. Долгов В.И. Исследование циклических и дифференциальных свойств уменьшенной модели шифра Лабиринт. / В.И. Долгов, И.В. Лисицкая, А.В. Григорьев, А.В. Широков // Прикладная радиоэлектроника. – 2009. – Т.8, №3 – С. 283-289.

182. Долгов В.И. Атака на полный дифференциал уменьшенной версии шифра Rijndael. / В.И. Долгов, И.В. Лисицкая, В.Э. Хряпин // Прикладная радиоэлектроника – 2010. – Т.9, №3 – С. 355-360.

183. Долгов В.И. Вариации на тему шифра Rijndael / В.И. Долгов, И.В. Лисицкая, А.В. Казимиров // Прикладная радиоэлектроника. – 2010. – Т.9, №3 – С. 321-325.

184. Долгов В.И. Мини-версия блочного симметричного алгоритма криптографического преобразования информации с динамически управляемыми криптопримитивами (Baby-ADE). / В.И. Долгов, А.А. Кузнецов, Р.В. Сергиенко, А.Л. Белоковаленко // Прикладная радиоэлектроника – 2008. – Т.7, №3 – С. 215-224.

185. Долгов В.И. Криптоанализ уменьшенной модели симметричного блочного алгоритма шифровании Nimbus / В.И. Долгов, Д.С. Ивонин // Збірник наукових робіт Харківського національного економічного університету №7, – 2008, – с. 32.

186. Долгов В.И. Исследование криптографических показателей уменьшенных моделей шифров ГОСТ и DES / В.И. Долгов, Я.А. Макарчук, А.В. Григорьев, Е.В. Дробатько // Прикладная радиоэлектроника, – 2011. − Т.10. − № 2. − С. 127−134.

187. Долгов В.И. Криптографические свойства уменьшенной версии шифра ″Калина″ / В.И. Долгов, Р.В. Олейников, А.Ю. Большаков, А.В. Григорьев, Е.В. Дробатько // Прикладная радиоэлектроника, 2010. – Т.9. – № 3. – С. 349-354.

188. Raphael Chung-Wei Phan. Mini Advanced Encryption Standard (Mini-AES): A testbed for Cryptanalysis Students / Raphael Chung-Wei Phan // Cryptologia. – October 2002. – XXVI(4). – P. 283-306.

189. Исследование криптографических свойств нелинейных узлов замены уменьшенных версий некоторых шифров. / В.И. Долгов, А.А. Кузнецов, И.В. Лисицкая, Р.В. и др. // Прикладная радиоэлектроника. − Харьков: ХТУРЭ. – 2009. – Т. 8, № 3. – С. 268-277.

190. Лисицкая И.В. Сравнение по эффективности суперблоков некоторых современных шифров. Радіоелектроніка. Інформатика. Управління. Запоріжжя 1(26)’ − 2012. – С. 37- 43.

191. A Description of Baby Rijndael, ISU CprE/Math 533; NTU ST765-U, February 19, 2003.

192. N. Ferguson and B. Schneier, Practical Cryptography, John Wiley and Sons, 2003.

193. Z.G. Chen and S.E. Tavares, Towards provable security of substitution-permutation encryption networks, Fifth Annual International Workshop on Selected Areas in Cryptography (SAC’98), LNCS 1556, pp. 43-56, Springer-Verlag, 1999.

194. H.M. Heys and S.E. Tavares, Avalanche characteristics of substitution-permutation encryption networks, IEEE Transactions on Computers, Vol. 44, No. 9, pp. 1131-1139, September 1995.

195. J.B. Kam and G.I. Davida, Structured design of substitution-permutation encryption networks, IEEE Transactions on Computers, Vol. C-28, No. 10, pp. 747-753, October 1979.

196. A.M. Youssef, Analysis and design of block ciphers, Ph.D. Thesis, Queen's University, Kingston, Canada, 1997.

197. Долгов В.И. Вопросы теории и цифрового моделирования нормальных марковских процессов. Издательсто М.О. 1978 г. 76 стр.

198. Иванов В.А. Математические основы теории автоматического регулирования / В.А. Иванов, Б.К.Чемоданов, В.С. Медведев // Изд-во ″Высшая школа″ М:. – 1971 – 755 с.

199. Ковальчук Л.В. Сходимость последовательности матриц вероятностей дифференциальных аппроксимаций немарковского блочного шифра к равновероятной матрице при увеличении количества циклов. // Прикладная радиоэлектроника – 2007. – Т.5, № 2 – С. 274-276.

200. ГОСТ. Государственный стандарт 28147-89. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. Государственный комитет СССР по стандартам, 1989.

201. Горбенко И.Д. Критерии и показатели защищённости информации / И.Д. Горбенко, А.В. Потий // Обработка информации: Сб. научн. тр.- Харьков: НАНУ, ПАНУ, ХВУ, 1995.- С. 11-15.

202. J. Daemen, V. Rijmen Security of a Wide Trail Design. INDOCRYPT 2002, LNCS 2551, pp. 1-11, 2002.

203. Лисицкая И.В. Большие шифры − случайные подстановки. / И.В. Лисицкая, А.А. Настенко // Межведомственный научн. технический сборник ″Радиотехника″. − 2011. − вып. 166. – С. 50-55.

204. Лисицкая И.В. Методология оценки стойкости блочных симметричных шифров. / И.В. Лисицкая // Автоматизированные системы управления и приборы автоматики. – 2011. – № 163. – С. 123-133.

205. Сачков В.Н. Цепи Маркова итерационных систем преобразований. Труды по дискретной математике. Том 6. с. 165-183. М.: ФИЗМАТЛИТ, 2002.

206. K. Ohkuma Security Assessment of Hierocrypt and Rijndael against the Differential and Linear Cryptanalysis/ K. Ohkuma, H. Shimizu, F. Sano, S. Kawamura// In Proceedings of the 2nd NESSIE workshop (2001).

207. L. Keliher Linear Cryptanalysis of Substitution-Permutation Networks, A thesis submitted to the School of Computing in conformity with the requirements for the degree of Doctor of Philosophy, 2003, P. 160.

208. P. Hawkes and L. O'Connor. XOR and Non-XOR Differential Probabilities/ EUROCRYPT'99, LNCS 1592, pp. 272-285, 1999.

209. Lisitskaya I.V. Importance of S-Blocks in Modern Block Ciphers. / I.V. Lisitskaya, Nastenko Nastenko A.A. and K.E. Lysytskiy. // I.J. Computer Network and Information Security, 2012, 10, 1-12. ISSN: 2074-9104.

210. Лисицкая Ирина Викторовна. Об участии S-блоков в формировании максимальных значений дифференциальных вероятностей блочных симметричных шифров. / И.В. Лисицкая, А.В. Казимиров // Proceedings International Conference SAIT 2011, Kyiv, Ukraine, May 23-28. – 2011. – с. 459.

211. Лисицкая Ирина Викторовна. Об участии S-блоков в формировании максимальных значений линейных вероятностей блочных симметричных шифров. / И.В. Лисицкая, В.В. Ковтун // Межведомственный научн. технический сборник ″Радиотехника″. – 2011. – вып. 166. – С. 17-25.

212. Markku-Juhani O. Saarinen Cryptographic Analysis of All 44-Bit S-Boxes. 2008.

213. Токарева Н. Н. Квадратичные аппроксимации специального вида для четырёхразрядных подстановок в S-блоках. Прикладная дискретная математика. 2008. Т. 1, N 1. С. 50-54.

214. Лисицкая И.В. Сравнительный анализ механизмов образования лавинного эффекта в алгоритмах DES и ГОСТ 28147-89. / И.В. Лисицкая, А.С. Бондаренко, Т.В. Цепурит // Інформаційно-керуючи системи на залізничному транспорті. – 1999. – №3. – С.24-30.

215. H. M. Heys. A Tutorial on Linear and Differential Cryptanalysis, CRYPTOLOGIA, v 26, N 3, 2002, p 189-221.

216. Ростовцев А.Г., Маховенко Е.Б. Введение в теорию итерированных шифров // СПб.: НПО «Мир и Семья», 2003.

217. Kim K., Park S., Lee S. Reconstruction of s2DES S-Boxes and their Immunity to Differential Cryptanalysis // Korea – Japan Workshop on Information Security and Cryptography. (Seoul, Korea. October 24–26, 1993) Proc. P. 282-291.

218. Долгов В.И. Исследование дифференциальных свойств мини-шифров Baby-ADE и Baby-AES. / В.И. Долгов, А.А. Кузнецов, Р.В. Сергиенко, О.И. Олешко // Прикладная радиоэлектроника. – 2009.– Т.8, №.3. – С. 252-257.

219. J. Seberry. Relationships among nonlinearity criteria / J. Seberry, X. S. Zhang and Y. Zheng. Presented at EUROCRYPT-94, 1994.

220. Krzysztof Chmiel. On Differential and Linear Approximation of S-box Functions / Biometrics, Computer Security Systems and Artificial Intelligence Applications. / Edited by Khalid Saeed, Jerzy Pejas and Romuald Mosdorf. // Poland, Springer – 2006. – P. 111-120.

221. Лисицкая И.В. Большие шифры − случайные подстановки. Сравнение дифференциальных и линейных свойств шифров, представленных на украинский конкурс и их уменьшенных моделей. / И.В. Лисицкая, А.А. Настенко, Лисицкий Е.К. // Прикладная радиоэлектроника. − 2012. − Т.00, № 0 – С. 000-000.

222. Лисицкая И.В. Дифференциальные свойства шифра FOX. / И.В. Лисицкая, Д. С. Кайдалов // Прикладная радиоэлектроника. – 2011. – Т. 10, № 2. – С. 122-126.

223. Лисицкая И.В. Большие шифры − случайные подстановки. Сравнение показателей статистической безопасности блочных симметричных шифров, представленных на украинский конкурс. / И.В. Лисицкая, А.А. Настенко, Лисицкий Е.К. // Прикладная радиоэлектроника. – 2012. – Т.00, № 0 – С. 000-000.

224. Жильников В. Криптография от компьютера до папируса / В. Жильников // М.: ABF, 1996.– 336 с.

225. Pascale S. The degrees of completeness, of avalanche effect, and of strict avalanche criterion for MARS, RC6, Rijndael, Serpent, and Twofish with reduced number of rounds. / S. Pascale // Siemens AG, ZT IK 3. April 3, 2000.

226. Результаты анализа алгоритма шифрования ADE / Р.В. Олейников, В.И. Руженцев, М.С. Михайленко, А.Б. Небывайлов / Прикладная радиоэлектроника. – 2008. – Т.7, №3 – С.210-214.

227. Rukhin A. et al. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. NIST Special Publication 800-22. Washington, 2000.

228. NIST IR 6390. A Statistical Test Suite for the Validation of Random Number Generators and Pseudo Random Number Generators for Cryptographic Applications, 2000.

229. Долгов В.И. О роли схем разворачивания ключей в атаках на итеративные шифры. / В.И. Долгов, А.А. Настенко // Прикладная радиоэлектроника. – 2012. – Т.0, №0 – С.000-000.

230. Lars R. Knudsen and John E. Mathiassen. On the Role of Key Schedules in Attacks on Iterated Ciphers. P. Samarati et al. (Eds.): ESORICS 2004, LNCS 3193, pp. 322–334, 2004. Springer-Verlag Berlin Heidelberg 2004.

231. Uri Blumenthal and Steven M. Bellovin, A better Key Schedule for DES-like ciphers. Proceedings of Pragocrypt’96 30 September - 3 October 1996.

232. Лисицкая И.В. Блочные симметричные шифры и марковские процессы. / И.В. Лисицкая // Прикладная радиоэлектроника. – 2012. – Т. 11, № 2 – С. 137-143.

233. Лисицкая И.В. Оценки максимальных значений дифференциалов и линейных корпусов Марковских шифров. / И.В. Лисицкая., В.И. Долгов, А.А. Настенко // Прикладная радиоэлектроника. – 2012. – Т. 11, № 2 – С. 144-151.

234. E. Biham. New types of cryptanalytic attacks using related keys. In T. Helleseth, editor, Advances in Cryptology: EUROCRYPT’93, Lecture Notes in Computer Science 765, pages 398–409. Springer Verlag, 1993.

235. L.R. Knudsen. Cryptanalysis of LOKI’91. In J. Seberry and Y. Zheng, editors, Advances in Cryptology, AusCrypt 92, Lecture Notes in Computer Science 718, pages 196–208. Springer Verlag, 1993.

236. A. Biryukov and D. Wagner. Slide attacks. In L. R. Knudsen, editor, Fast Software Encryption, Sixth International Workshop, Rome, Italy, March 1999, Lecture Notes in Computer Science 1636, pages 245–259. Springer Verlag, 1999.

237. L.R. Knudsen, Iterative characteristics of DES and s^2-DES, Advances in Cryptology, Proc. Crypto '92, LNCS 740, E. F. Brickell, Ed., Springer-Verlag, 1993, pp. 497-511.

238. Головашич *С.А.* Ключевые группы в атаках дифференциального криптоанализа DES-подобных шифров. /С.А. Головашич // Радиотехника.– 2000. – Вып. 114. – С. 57-62.

239. Eli Biham, Adi Shamir. Differential Cryptanalys is of DES-like Cryptosystems. Journal of Cryptology, Vol. 4 No. 1 1991.

240. Biham, Adi Shamir. Differential Cryptanalysis of the full 16- round DES. Technical Report # 708, Technion - Israel Institute of Technology.

241. Gilles Piret1, Fran¸cois-Xavier Standaert2 Provable Security of Block Ciphers Against Linear Cryptanalysis - a Mission Impossible. 2004.

242. http://en.wikipedia.org/wiki/Key_schedule.

243. Lars R. Knudsen. Practically secure Feistel ciphers. In Fast Software Encryption. Cambridge Security Workshop, Proceedings, pages 211–221. Springer-Verlag, 1994

244. Lars R. Knudsen and John Erik Mathiassen. On the Role of Key Schedules in Attacks on Iterated Ciphers. In Samarati et al. (Eds.): ESORICS 2004, LNCS 3193, pp. 322–334, 2004.

245. Biham E. New Types of Cryptanalytic Attack Using Related Keys. J. of Cryptology, vol. 7, 1994, pp.

246. Лепеха А.Н. Сравнительный анализ схем разворачивания ключей блочных симметричных шифров. / А.Н. Лепеха // Радиотехника. Всеукр. межвед. науч.-техн. сб. – 2005. – Вып. 141. – С. 64 -69.

247. Biryukov A., Wagner D. Slide Attack // FSE'99, LNCS 1636, Springer-Verlag, 1999, pp. 245-259.

248. Van Oorschor P.C. Wiener M.J. Improving Implementable Meet-In-The-Middle Attack by Order of Magnitude // Bell-Northern Research, P.O. Ontario, Canada, 1996.

249. Kelsey J., Schneier B., Wagner D. Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER and 3-DES // CRYPTO'96, Springer-Verlag, 1996, pp. 237-251.