ИНФОРМАТИКА ИНФОРМАЦИОННЫЕ СИСТЕМЫ

УДК 004.934

Модификация алгоритмов на основе сети Фейстеля посредством внесения избыточности с помощью кодов Хэмминга

Е. И. Александрова, А. П. Шкарапута
Пермский государственный национальный исследовательский университет
Россия, 614990, г. Пермь, ул. Букирева, 15
kaaate11@gmail.com, shkaraputa@psu.ru

Выявлены достоинства и недостатки классической сети Фейстеля, на основании которых были выдвинуты требования к алгоритмам на основе сети Фейстеля для повышения их криптостойкости. В соответствии с выдвинутыми требованиями предложен модифицированный алгоритм на основе сети Фейстеля с использованием кодов Хэмминга и элемента случайности; проведен анализ основных характеристик алгоритма: времени выполнения, объема зашифрованного текста, криптостойкости, – относительно классической сети Фейстеля. В результате анализа было выявлено, что модифицированный алгоритм более криптографичеки стойкий, чем классическая сеть Фейстеля, однако время выполнения модифицированного алгоритма в два раза больше, чем время выполнения классической сети Фейстеля.

Ключевые слова: *шифрование; сеть Фейстеля; коды Хэмминга.*

DOI: 10.17072/1993-0550-2018-3-95-103

Введение

С развитием информационного общества криптография стала иметь основополагающее значение в обеспечении безопасности информации. Одним из видов криптографического преобразования информации является шифрование – обратимое преобразование данных с целью их сокрытия от третьих лиц.
Начиная со времен развития Древнего Египта, и по сей день появляется и разрабатывается множество алгоритмов и методов шифрования данных. Одним из таких методов является метод блочного симметричного шифрования "Сеть Фейстеля", который получил широкое распространение благодаря тому, что, с одной стороны, он обеспечивает выполнение требования о многократном использовании ключа и материала исходного блока информации, а с другой стороны, достаточно прост и компактен. Созданный в 1970-х гг. прошлого столетия, данный алгоритм лежит в основе многих современных алгоритмов шифрования таких, как блочный шифр "Магма", RTEA, CLEFIA, Camellia, Sinople и др.

За время своего существования и в силу своего широкого распространения сеть Фейстеля и основанные на ней алгоритмы шифрования являются предметом исследования многих криптоаналитиков, которые выявили как сильные, так и слабые их стороны. Для сокращения числа уязвимостей и иных недостатков разрабатываются модификации и разновидности классической сети Фейстеля, в которых используются различные вспомогательные функции, например применение S-блоков, циклический сдвиг и пр.

© Александрова Е. И., Шкарапута А. П., 2018

В данной статье рассматривается модификация сети Фейстеля с использованием кодов Хэмминга и элемента случайности. Данная модификация может быть использована при разработке алгоритмов шифрования для повышения их криптостойкости.

1. Сеть Фейстеля

Среди симметричных блочных шифров широкое распространение получила сеть Фейстеля.

Сеть Фейстеля (Feistel network, Feistel cipher) – алгоритм симметричного блочного шифрования, разработанный Хорстом Фейстелем в 1973 г. [1]. Представляет собой последовательность обратимых преобразований текста, при котором значение, вычисленное от одной из частей текста, накладывается на другие части.

Структура сети выполняется таким образом, что для шифрования и дешифрования используется один и тот же алгоритм – различие состоит только в порядке использования материала ключа.

Структура алгоритмов шифрования и дешифрования сети Фейстеля представлена на рис. 1 и 2 соответственно.

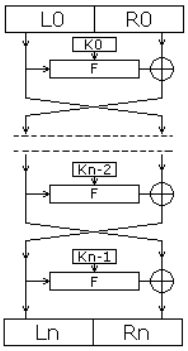

Рис. 1. *Структура алгоритма шифрования сети Фейстеля*

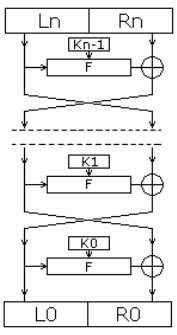

Рис. 2. *Структура алгоритма дешифрования сети Фейстеля*

1.1. Достоинства сети Фейстеля

Среди главных достоинств классической сети Фейстеля можно выделить следующие [2]:

1. Простота аппаратной реализации, позволяющая применять ее на платформах с ограниченными ресурсами, поскольку требует небольших объемов памяти.
2. Простота программной реализации, благодаря чему значительная часть функций поддерживается на аппаратном уровне в современных компьютерах (например, сложение по модулю 2 ("xor"), сложение и умножение по модулю 2^N и т.д.).
3. Возможность распараллеливания вычислений, поскольку блоки открытого текста не связаны между собой.
4. Хорошая изученность (исследования М. Люби и Ч. Ракоффа, 1988 г. [3], М. Наора и О. Рейнголда, 1997 г. [4]).

1.2. Недостатки сети Фейстеля

Классическая сеть Фейстеля имеет ряд существенных недостатков:

1. Одинаковые блоки открытого текста соответствуют одинаковым блокам закрытого текста, что делает данный алгоритм уязвимым для атаки на основе открытых текстов. В частности, Дж. Трегер и Дж. Патарин в своей работе [5] продемонстрировали, что для четырех раундов сети Фейстеля сложность такой атаки составляет O(2^n), где n – длина блока.
2. Уязвимость к частотному криптоанализу при небольших размерах блока, в результате чего возможно произвести статистический анализ биграмм, триграмм и т.д. [6].
3. На каждом раунде изменяется только половина блока обрабатываемого текста, что приводит к необходимости увеличивать количество раундов для достижения требуемой стойкости [7].
4. При изменении одного бита открытого текста изменяются только биты соответствующего блока закрытого текста. Таким образом, в некоторых случаях подмена битов может быть трудно обнаруживаемой, поскольку каждый блок шифруется независимо от других. Данный недостаток становится особенно критичным при небольшом количестве раундов.

2. Требования к алгоритму на основе сети Фейстеля для увеличения его криптостойкости

Наличие уязвимостей в классической сети Фейстеля приводит к необходимости ее модификации для повышения криптостойкости алгоритма.

Возможны два варианта решения проблемы ненадежности сети Фейстеля при малом количестве раундов:

1. Увеличение числа раундов таким образом, чтобы обеспечить требуемую стойкость. М. Люби и Ч. Ракофф в своей работе [3] доказали, что для сети Фейстеля минимальное количество раундов равняется четырем, а с учетом существующих современных рекомендаций число раундов в алгоритмах блочного шифрования должно быть на два больше минимальной безопасной границы [8].
2. Обработка полного блока за один раунд шифрования, благодаря чему требуемая стойкость может достигаться за сравнительно небольшое число раундов.

Проблему уязвимости к частотному криптоанализу можно решить, выдвинув к алгоритму следующие требования:

1. Использование блоков, длиной не менее 64 бит. Однако в настоящее время большинство современных блочных шифров ориентируются на длину блока 128 бит (такое требование выдвигалось для кандидатов конкурса AES).
2. Использование переменной длины блоков на выходе алгоритма шифрования, что уменьшает вероятность атаки методом частотного криптоанализа.
3. Использование элемента случайности, который будет препятствовать статистическому анализу закрытого текста.

Для решения проблем уязвимости к атакам на основе открытых текстов и путем подмены битов необходимо реализовать алгоритм таким образом, чтобы результат шифрования текущего блока открытого текста зависел от результата шифрования предыдущего блока.

Такой метод позволяет:

1) создать "лавинный эффект", распространяющийся не на один зашифрованный блок, а на все последующие блоки, благодаря чему подмена битов в закрытом тексте будет легко обнаруживаемой;

2) устранить ситуацию, когда одинаковым блокам открытого текста соответствуют одинаковые блоки закрытого текста.

Для обеспечения устойчивости алгоритма к атаке методом грубой силы должно быть вычислительно сложно осуществить полный перебор ключа шифрования. В соответствии с современными вычислительными мощностями, длина ключа не должна быть менее 128 бит. С целью повышения надежности следует выбирать длину ключа, равную 256 бит.

3. Описание модифицированного алгоритма на основе сети Фейстеля с использованием кодов Хэмминга

В соответствии с требованиями, описанными ранее, предложен алгоритм симметричного блочного шифрования на основе сети Фейстеля со следующими характеристиками:

– размер ключа: 256 бит;
– размер блока: 128 бит;
– количество раундов: 8.

3.1. Алгоритм шифрования

Отметим, что в алгоритме присутствует элемент случайности, заключающийся в совершении случайной ошибки в блоке, исправление которой возможно при дешифровании блока благодаря проверочным битам кодов Хэмминга. Добавление к блоку данных кодов Хэмминга зависит от того, в каком бите была установлена ошибка в последний раз при шифровании блока, предшествующего текущему, следующим образом:

1. Порядковый номер бита (нумерация начинается с единицы), в который была установлена ошибка, переводится в двоичную систему счисления. Первые 8 бит получившегося двоичного числа формируют двоичный вектор (для первого шифруемого блока двоичный вектор формируется из первых 8 бит исходного ключа).
2. Если i-й разряд двоичного вектора равен единице, то на i-м раунде шифрования будет происходить добавление к блоку данных кодов Хэмминга и случайной ошибки, иначе – не будет.

Рассмотрим структуру раунда алгоритма шифрования (рис. 3). Пунктирной линией обведены операции, выполнение которых зависит от результата шифрования блока, предшествующего текущему.

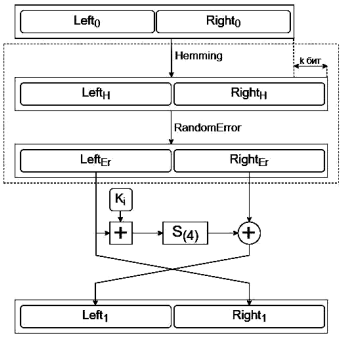

Рис. 3. *Структура раунда шифрования*

В начале раунда могут выполняться следующие операции:

1. Hemming – добавление кодов Хэмминга ко всему блоку данных, где k – количество добавляемых проверочных бит (k = 8);
2. RandomError – добавление случайной ошибки в блок данных (инвертирование случайно выбранного бита блока).

Далее исходный блок либо исходный блок, к которому были применены операции Hemming и RandomError, разбивается на две равные части, над которыми последовательно производятся следующие раундовые операции:

1. Сложение раундового ключа Ki с левым подблоком по модулю 2 в степени размера операндов в битах (операция обозначена символом ⊕ на рис. 1).
2. Применение S-блоков: над результатом сложения выполняются подстановки по таблице четырехбитных подстановок.
3. Сложение по модулю 2: значение результата сложения после подстановок складывается с правым подблоком (операция обозначена символом ⊕ на рис. 1).
4. Перестановка левого подблока и результата последнего сложения. Данная операция не выполняется на последнем раунде шифрования.

3.2. Алгоритм дешифрования

Рассмотрим структуру раунда алгоритма дешифрования (рис. 4). Пунктирной линией обведены операции, выполнение которых зависит от результата дешифрования блока, предшествующего текущему.

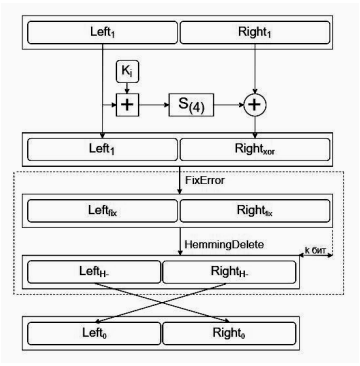

Рис. 4. *Структура раунда дешифрования*

В начале раунда дешифрования выполняются те же раундовые операции, что в раунде шифрования: сложение с раундовым ключом, применение S-блоков, сложение подблоков по модулю 2 и их перестановка.

Рассмотрим операции FixError и HemmingDelete:

1. FixError – исправление ошибки в блоке данных, внесенной при шифровании блока, (инвертирование бита с ошибкой);
2. DeleteHemming – удаление проверочных k бит кодов Хэмминга, добавленных при шифровании блока (k = 8).

Выполнение описанных выше операций зависит от того, в каком бите была исправлена ошибка в первый раз при дешифровании блока, предшествующего текущему, следующим образом:

1. Порядковый номер бита (нумерация начинается с единицы), в котором была исправлена ошибка, переводится в двоичную систему счисления. Первые 8 бит получившегося двоичного числа формируют двоичный вектор (для первого дешифруемого блока двоичный вектор формируется из первых 8 бит исходного ключа).
2. Если i-й разряд двоичного вектора равен единице, то на (7-i)-м раунде шифрования будет происходить добавление к блоку данных кодов Хэмминга и случайной ошибки, иначе – не будет.

3.3. Определение раундовых ключей

Исходный 256-битный ключ K разбивается на два ключа K1 и K2: K1 – первые 128 бит ключа K, K2 – последние 128 бит ключа K. Каждый ключ K1 и K2 разбивается еще на два ключа по тому же принципу (рис. 5).

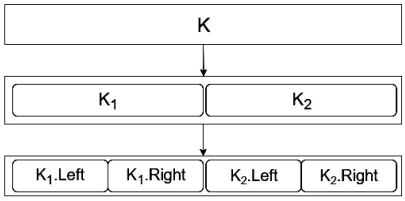

Рис. 5. *Разбиение исходного ключа*

В итоге получается четыре возможных раундовых ключа. Для алгоритма шифрования

- K1.Left – применяется на раундах 0, 4;
- K1.Right – применяется на раундах 1, 5;
- K2.Left – применяется на раундах 2, 6;
- K2.Right – применяется на раундах 3, 7.

Для алгоритма дешифрования раундовые ключи применяются в обратном порядке.

На каждом раунде шифрования к ключам K1 и K2 применяется операция Hemming, если она была применена к обрабатываемому блоку данных. Это необходимо для того, чтобы размер раундового ключа совпадал с размером левого подблока при их сложении в процессе каждого раунда алгоритма шифрования.

Аналогичным образом на каждом раунде дешифрования к ключам K1 и K2 применяется операция HemmingDelete.

Перед дешифрованием каждого блока необходимо провести процедуру подготовки раундовых ключей: применить к ключам K1 и K2 операцию Hemming в том количестве, в котором она была применена при шифровании текущего блока.

Данное количество определяется следующим образом:

1. Порядковый номер бита (нумерация начинается с единицы), в котором была исправлена ошибка в первый раз при дешифровании блока, предшествующего текущему, переводится в двоичную систему счисления. Первые 8 бит получившегося двоичного числа формируют двоичный вектор (для первого дешифруемого блока двоичный вектор формируется из первых 8 бит исходного ключа).
2. Количество операций Hemming, применяемых к ключам K1 и K2, равняется количеству единиц в двоичном векторе.

4. Анализ модифицированного алгоритма на основе сети Фейстеля с использованием кодов Хэмминга

4.1. Анализ времени выполнения алгоритма

Проведено сравнение времени выполнения модифицированного алгоритма с временем выполнения алгоритма на основе сети Фейстеля с той же образующей функцией, но без внесения избыточности и элемента случайности.

В качестве входных данных для алгоритмов использовались файлы размером 512 Кб. Измерения производились на компьютере со следующими характеристиками:

- Количество ядер процессора: 4;
- Тактовая частота процессора: 2,3 ГГц;
- Оперативная память: 4 Гб.

В ходе тестирования было измерено время выполнения каждого алгоритма в зависимости от количества раундов. Для каждого алгоритма и каждого количества раундов проводилось не менее 10 измерений.

По результатам измерений было определено среднее, минимальное и максимальное время выполнения алгоритмов шифрования в зависимости от количества раундов (табл. 1).

Таблица 1. *Результаты измерений времени выполнения алгоритмов шифрования*

Количество раундов	Сеть Фейстеля			Модифицированный алгоритм		
	t_{avg} (с)	t_{min} (с)	t_{max} (с)	t_{avg} (с)	t_{min} (с)	t_{max} (с)
1	0,64	0,59	0,69	1,42	1,38	1,51
2	1,24	1,17	1,32	2,24	2,2	2,31
3	1,66	1,61	1,7	3,2	3,14	3,33
4	2,14	2,05	2,29	4,09	3,99	4,19
5	2,6	2,49	2,75	5,33	5,23	5,63
6	3,07	3,02	3,19	6,4	6,3	6,6
7	3,48	3,4	3,57	7,32	7,29	7,6
8	4,03	3,97	4,19	7,79	7,6	8,07

На рис. 6 в виде графика изображены средние значения времени выполнения алгоритмов шифрования в зависимости от количества раундов. В результате анализа полученных данных было установлено, что в среднем время выполнения модифицированного алгоритма шифрования в 2 раза больше, чем у классической сети Фейстеля без внесения избыточности и элемента случайности.

Рис. 6. *Результаты измерений времени выполнения алгоритмов шифрования*

Аналогичные измерения были проведены для алгоритмов дешифрования (табл. 2).

Таблица 2. *Результаты измерений времени выполнения алгоритмов дешифрования*

Количество раундов	Сеть Фейстеля			Модифицированный алгоритм		
	t_{avg} (с)	t_{min} (с)	t_{max} (с)	t_{avg} (с)	t_{min} (с)	t_{max} (с)
1	0,7	0,57	0,92	1,41	1,39	1,51
2	1,27	1,17	1,47	2,25	2,22	2,32
3	1,67	1,62	1,71	3,25	3,17	3,33
4	2,14	2,08	2,21	4,1	3,99	4,22
5	2,58	2,5	2,71	5,36	5,24	5,69
6	3,06	3	3,22	6,4	6,33	6,68
7	3,5	3,44	3,62	7,44	7,3	7,61
8	4,21	4	4,64	8,53	7,93	9,21

На рис. 7 в виде графика изображены средние значения времени выполнения алгоритмов дешифрования в зависимости от количества раундов. В результате анализа полученных данных было установлено, что в среднем время выполнения модифицированного алгоритма дешифрования в 2 раза больше, чем у классической сети Фейстеля без внесения избыточности и элемента случайности.

Рис. 7. *Результаты измерений времени выполнения алгоритмов дешифрования*

4.2. Анализ объема зашифрованного текста

Была определена зависимость от количества раундов увеличения объема входного файла после применения к нему алгоритма шифрования с внесением избыточности. Для каждого количества раундов проводилось не менее 10 измерений. Объем входного файла составлял 100 Кб.

По результатам измерений был определен средний, минимальный и максимальный объем входного файла после применения к нему алгоритма шифрования с внесением избыточности (табл. 3).

Таблица 3. *Результаты измерений объема выходного файла*

Количество раундов	Модифицированный алгоритм		
	V_{avg} (Кб)	V_{min} (Кб)	V_{max} (Кб)
1	106,25	106,25	106,25
2	108,52	108,47	108,57
3	110,98	110,91	111,12
4	113,41	113,07	113,53
5	116,19	116,04	116,57
6	118,52	118,43	118,83
7	120,11	120,07	120,57
8	121,45	120,84	121,6

На рис. 8 в виде графика изображены средние значения объема выходного файла после применения к нему алгоритма шифрования с внесением избыточности в зависимости от количества раундов. В результате анализа полученных данных было установлено, что для 8 раундов шифрования объем выходного файла увеличивается в 1,21 раз.

Рис. 8. *Результаты измерений среднего объема зашифрованного файла*

4.3. Анализ криптостойкости алгоритма

Проведен анализ криптостойкости алгоритма шифрования с применением кодов Хэмминга и элемента случайности относительно уязвимостей классической сети Фейстеля, определенных ранее.

4.3.1. Частотный криптоанализ

В классической схеме Фейстеля совпадающие блоки открытого текста шифруются одинаково, в результате чего после выполнения процедуры шифрования закрытый текст сохраняет частотные характеристики открытого текста, что в свою очередь может дать криптоаналитику определенную полезную информацию. Особенно опасной данная ситуация становится, если при шифровании текст разбивался на блоки малой длины (от 8 до 32 бит), так как становится возможным произвести атаку методом частотного криптоанализа, основываясь на частоте встречаемости символов или групп символов.

Для анализа стойкости модифицированного алгоритма к данной уязвимости был выбран произвольный текст, содержащий 20 совпадающих 128-битных блоков данных, затем данный текст был зашифрован с использованием классической сети Фейстеля и с использованием модифицированного алгоритма, после чего был выполнен поиск совпадающих блоков в каждом из полученных закрытых текстов. Для обеспечения корректности анализа размер блока данных для обоих алгоритмов шифрования равнялся 128 бит.

Частотный анализ закрытого текста, полученного в результате выполнения классической схемы Фейстеля, показал, что закрытый текст так же, как и открытый, содержит 20 совпадающих 128-битных блоков.

Частотный анализ закрытого текста, полученного в результате выполнения модифицированного алгоритма, показал, что закрытый текст не содержит ни одного повторяющегося блока данных.

Такая ситуация обусловлена, во-первых, тем, что длина блока не является фиксированной, а изменяется от 128 до 192 бит, поэтому разбить закрытый текст на блоки нужного размера, не зная ключ шифрования – трудоемкая задача. Во-вторых, при шифровании каждого блока данных (в случае применения кодов Хэмминга) искажались случайные биты, причем максимальное количество таких бит для одного блока – 8. Вероятность того, что для различных блоков данных будет установлено равное количество ошибочных бит на одних и тех же позициях, очень мала.

Таким образом, можно сделать вывод, что разработанный алгоритм шифрования с применением кодов Хэмминга и элемента случайности является устойчивым к частотному криптоанализу.

4.3.2. Лавинный эффект

В классической сети Фейстеля при изменении одного бита открытого текста изменяются только биты соответствующего блока закрытого текста, следовательно, возникает угроза, связанная с возможностью необнаружимой подмены битов закрытого текста, что особенно опасно, если число раундов шифрования невелико.

Для анализа стойкости модифицированного алгоритма к данной проблеме зашифруем текст, используя модифицированный алгоритм и классическую сеть Фейстеля. Открытый текст (три 128-битных блока): «AAAAAAAAAAAAAAAABBBBBBBBBBBBBBBBCCCCCCCCCCCCCCCC».

Выполнив шифрование двумя алгоритмами, получим на выходе две строки. В этих строках изменим произвольный бит во втором блоке. Дешифруем строки с ошибкой.

После дешифрования, используя классическую сеть Фейстеля, получили следующую строку:

«AAAAAAAAAAAAAAAABBBFBBBBBBBBFBBBCCCCCCCCCCCCCCCC».

В результате дешифрования были искажены только 4 бита данных одного блока.

После дешифрования, используя модифицированный алгоритм, получили следующую строку:

«AAAAAAAAAAAAAAAABBBBBBB@BBB□□□□□r□□K□□<□□-)□c□□8□□8□U□>□□□>4□3».

В результате дешифрования были искажены символы множества бит, причем не только второго, но и третьего блока данных.

Такая ситуация возникает по причине того, что на этапе шифрования после применения кодов Хэмминга уже была внесена случайная ошибка, которую впоследствии удается успешно обнаружить и исправить, не на-

рушая процесс дешифрования. Однако если в тексте будут искажены более одного бита, исправить ошибку с помощью кодов Хэмминга становится невозможным, а, значит, последующий процесс дешифрования будет выполняться некорректно не только для текущего блока, но и для всех последующих. Следовательно, лавинный эффект в модифицированном алгоритме присутствует и достаточно ярко выражен.

Таким образом, можно сделать вывод, что при использовании разработанного алгоритма шифрования с применением кодов Хэмминга и элемента случайности подмена бит в зашифрованном тексте становится более очевидной и заметной. Однако для того чтобы исказить весь текст, злоумышленнику достаточно будет исказить несколько бит начальных блоков зашифрованного текста.

4.3.3. Линейный и дифференциальный криптоанализ

Исследования, проводимые Винсентом Рименом, Джоаном Дэймоном и др. [9] показали, что внедрение в алгоритмы шифрования некоторых случайных параметров позволяет повысить криптостойкость алгоритмов к линейному и дифференциальному криптоанализу.

Таким образом, можно сделать предположение, что разработанный алгоритм шифрования с применением кодов Хэмминга и элемента случайности более устойчив к линейному и дифференциальному криптоанализу, чем классическая сеть Фейстеля.

Заключение

В результате проведенного исследования был предложен модифицированный алгоритм на основе сети Фейстеля с использованием кодов Хэмминга и элемента случайности, и были выявлены его достоинства и недостатки.

Достоинства алгоритма:

1. Алгоритм является устойчивым к частотному криптоанализу.
2. Алгоритм обладает лавинным эффектом, что снижает риск не обнаружимой подмены битов закрытого текста.
3. Алгоритм устойчив к атаке грубой силы.
4. Простота программной и аппаратной реализации.
5. Возможность распараллеливания вычислений, в частности, для вычисления помехоустойчивых кодов Хэмминга.
6. Возможность интеграции в другие алгоритмы блочного шифрования, основанные на сети Фейстеля (например, ГОСТ 28147-89, IDEA и др.).

Недостатки алгоритма:

1. Время выполнения алгоритма шифрования в среднем в 2 раза больше, чем у классической сети Фейстеля.
2. Время выполнения алгоритма дешифрования в среднем в 2 раза больше, чем у классической сети Фейстеля.
3. Объем зашифрованного файла в среднем увеличивается в 1,21 раз.
4. Алгоритм обладает лавинным эффектом, что упрощает искажение закрытого текста: для искажения всего текста достаточно исказить несколько бит начальных блоков.

Следует отметить, что в разработанном алгоритме в качестве раундовых операций использовались две операции: сложение по модулю и S-блоки. Эти операции могут быть заменены на другие, более криптографически стойкие операции, которые окажут положительное влияние на криптостойкость алгоритма.

Список литературы

1. *Feistel H.* Cryptography and Computer Privacy, Scientific American, May 1973. Vol. 228. № 5. P. 15–23.
2. *Дроздова Е.С.* История создания и описание конструкции Фейстеля. СПб.: Информационная безопасность, проектирование и технология элементов и узлов компьютерных систем. СПб.: НИУ ИТМО, 2013. Вып. 1. С. 70–74.
3. *Luby M., Rackoff C.* How to construct pseudorandom permutations from pseudorandom functions. SIAM Journal of Computing, April 1988. Vol. 17.
4. *Naor M., Reingold O.* On the construction of pseudo-random permutations: Luby-Rackoff revisited. Journal of Cryptology, 1999. Vol. 12.
5. *Patarin J.* Security of Random Feistel Schemes with 5 or more Rounds – Crypto'2004, Versailles Cedex, France, 2004.

6. *Mekhazina T., Zidani A.* BAT algorithm for Cryptanalysis of Feistel cryptosystems – International Journal of Intelligent Systems and Applications in Engineering, aug. 2004.
7. *Баричев С.Г., Гончаров В.В., Серов Р.Е.* Основы современной криптографии. М.: Горячая линия–Телеком, 2002. 175 с.
8. *Молдовян А.А. и др.* М75. Криптография: скоростные шифры. СПб.: БХВ-Петербург, 2002. 496 с.
9. *Rijmen V., Daemen J., Preneel B., Bosselares A., Win E.* The chipher Shark Katholieke Universiteit Leuven, ESAT-COSIC K. Mercierlaan 94, B-3001 Heverlee, Belgium.

Modification of algorithms based on the Feistel network by redundancy introduction using Hamming codes

E. I. Aleksandrova, A. P. Shkaraputa
Perm State University; 15, Bukireva st., Perm, 614990, Russia
kaaate11@gmail.com, shkaraputa@psu.ru

The article revealed the merits and demerits of the classical Feistel network; on the basis of those, requirements for algorithms based on the Feistel network were put forward to enhance their cryptostability. In accordance with these requirements, a modified algorithm based on the Feistel network was proposed using Hamming codes and an element of randomness; analysis was performed for the main characteristics of the algorithm (execution time, volume of encrypted text, cryptostability) relative to the classical Feistel network. The analysis revealed that the modified algorithm is more cryptographically stable than the classical Feistel network; however, the execution time for the modified algorithm is twice as long as the time of execution of the classical Feistel network.

Keywords: *encryption; Feistel network; Hamming codes.*