Расширение класса булевых функций, используемых в алгоритмах симметричного шифрования

С.Ю. Корабельщикова[1]

[1]Северный (Арктический) федеральный университет имени М.В. Ломоносова, наб. Северной Двины 17, Архангельск, Россия, 163007

Аннотация. Одним из стандартных криптографических преобразований является сложение по модулю два открытого двоичного текста с ключевой двоичной последовательностью. В данной работе получено описание всех булевых функций от n аргументов, подходящих для использования в криптографических преобразованиях вместо функции сложения по модулю два (будем их называть покомпонентные булевы функции). Также приведен пример их использования в алгоритме криптографического преобразования ГОСТ Р 34.12-2015. В статье предложен алгоритм генерации покомпонентных булевых функций от n переменных при различных значения n и k, где k – номер переменной, значение которой возвращает функция. Для случая n=3 и $k\in\{1,2\}$ представлены все булевы функции, замещающие функцию сложения по модулю 2. В работе предложен метод шифрования на основе покомпонентных булевых функций и генератора псевдослучайной последовательности элементов из поля $GF(2^{n-1})$. Использование булевых функций, возвращающих значение одного из аргументов при повторном применении, расширяет разнообразие промежуточных вариантов раундовых преобразований, дает новый вариативный метод шифрования, в конечном итоге существенно увеличивая криптостойкость шифра.

1. Введение

К настоящему времени многие работы по криптографии были посвящены булевым функциям, изучению тех или иных криптографических свойств булевых функций, а также возможностям их использования в целях защиты информации. Эти вопросы рассматриваются как в научных статьях, так и в ряде учебных пособий для ВУЗов, например, [1,2]. Наиболее полный обзор криптографических свойств булевых функций и имеющихся результатов приведен в [3]. Данная статья является продолжением работы [4], в которой авторами были предложены булевы функции от трех аргументов, замещающие операцию сложения по модулю 2.

Одним из стандартных криптографических преобразований, применяемых в различных алгоритмах симметричного шифрования, является поразрядное сложение по модулю два открытого текста, представленного в двоичном виде, с ключевой двоичной последовательностью. Например, стандарт ГОСТ Р 34.12-2015 [5] - это симметричный шифр, в котором началом последовательности преобразований является преобразование $X[k]$:

$$X[k](a) = k \oplus a, \quad (1)$$

где: k – раундовый ключ, a – открытый текст $k, a \in V_{128}$.

Метод расшифрования заключается в повторном сложении по модулю 2 зашифрованного текста с тем же самым ключом k:

$$D\,X[k](a) = (k \oplus a) \oplus k = a. \quad (2)$$

Для зашифрования и расшифрования информации применяется одна и та же булева функция $F(x, y)=x \oplus y$, обладающая свойством:

$$F(F(x, y), y) = x. \quad (3)$$

Ранее в работе [4] нами было предложено 10 булевых функций от трех переменных, которыми можно заменить сложение по модулю 2. Для них был введен термин – покомпонентные функции. В общем случае, булева функция от n переменных, возвращающая значение первого аргумента, должна удовлетворять условию:

$$F(F(x_1, x_2, \ldots, x_n), x_2, \ldots, x_n) = x_1. \quad (4)$$

На рисунке показано возможное местоположение покомпонентных функций F_j, зависящих от $n+1$ аргумента, в процессе зашифрования 128-разрядного блока информации. Представлены два алгоритма: алгоритм из стандарта ГОСТ Р 34.12-2015 (слева) и предлагаемая авторами модификация (справа).

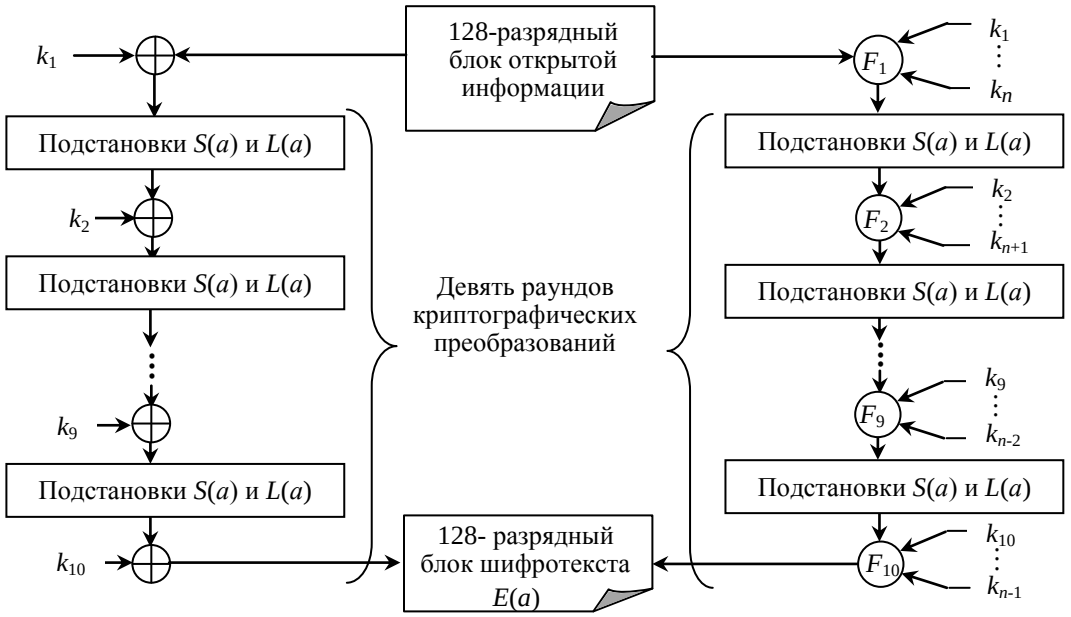

Рисунок 1. Модификация алгоритма шифрования из стандарта ГОСТ Р 34.12-2015 с использованием покомпонентных функций.

В настоящей статье получено описание всех булевых функций от n аргументов, подходящих для использования в симметричных криптографических преобразованиях вместо функции сложения по модулю два. Предложен алгоритм их генерации и метод шифрования на основе покомпонентных булевых функций и генератора псевдослучайной последовательности элементов поля $GF(2^{n-1})$.

2. Алгоритм генерации и свойства покомпонентных функций

Условие (4) означает, что выполняется система уравнений:

$$\begin{cases} F(F(0, x_2, \ldots, x_n), x_2, \ldots, x_n) = 0 \\ F(F(1, x_2, \ldots, x_n), x_2, \ldots, x_n) = 1' \end{cases} \quad (5)$$

откуда следует, что $F(0, x_2, \dots, x_n) = 0$ тогда и только тогда, когда $F(1, x_2, \dots, x_n)$=1 и наоборот, $F(0, x_2, \dots, x_n)$=1 тогда и только тогда, когда $F(1, x_2, \dots, x_n)$=0. Договоримся представлять булеву функцию F вектором значений $\tilde{F}$=(α_0,α_1,…,α_{2^n-1}), записанным в порядке соответствия упорядоченным наборам значений переменных от нулевого к единичному. Тогда полученные условия означают, что если вектор значений покомпонентной булевой функции разделить на две части, то вторая часть будет инверсна первой. Следовательно, справедливо следующее утверждение.

Теорема 1. Число булевых функций от n переменных, удовлетворяющих условию (4), равно $2^{2^{n-1}}$. Булева функция удовлетворяет условию (4) тогда и только тогда, когда вторая половина её вектора значений инверсна первой.

Доказательство.

Докажем первое утверждение. Булеву функцию от n переменных можно задать вектором значений длины 2^n. Поскольку вторая половина вектора значений полностью зависит от первой (инверсна первой), то задать ее можно, указав произвольным образом первую половину вектора значений. Она имеет длину 2^{n-1}, а число двоичных векторов такой длины равно $2^{2^{n-1}}$.

Докажем второе утверждение теоремы. Значения $F(0, x_2, \dots, x_n)$ находятся в первой половине вектора $\tilde{F}$, и в том же порядке во второй половине вектора $\tilde{F}$ находятся значения $F(1, x_2, \dots, x_n)$. Из условий, полученных выше, следует, что $F(0, x_2, \dots, x_n) = \overline{F(1, x_2, \dots, x_n)}$, то есть вторая половина вектора $\tilde{F}$ инверсна первой.

Теорема доказана.

Пример 1. При n=2 имеем $2^{2^{2-1}}$=4 булевых функции, возвращающих первый аргумент. Перечислим их векторы значений. Первую половину вектора значений задаём произвольным образом, вторую достраиваем инверсно первой: 0011, 0110, 1001, 1100.

Получили следующие 4 функции: $F(x, y) = x$, $F(x, y) = x+y$, $F(x, y) = x \leftrightarrow y$ и $F(x, y) = \bar{x}$.

Пример 2. При n=3 имеем $2^{2^{3-1}}$=16 булевых функций, возвращающих первый аргумент. Их векторы значений: 00001111, 00011110, 00101101, 00111100, 01001011, 01011010, 01101001, 01111000, 10000111, 10010110, 10100101, 10110100, 11000011, 11010010, 11100001, 11110000.

Учитывая требования к криптографическим функциям, сделаем выводы из примеров 1 и 2. Заметим, что первая и последняя функции являются отрицанием друг друга. Аналогично связаны вторая и предпоследняя функции, и так далее. Таким образом, если мы собираемся использовать в алгоритме шифрования несколько покомпонентных функций, будем выбирать их только по одной из каждой пары.

Также очевидно, что некоторые из полученных функций содержат фиктивные переменные, что недопустимо для криптографических функций. Применив к выбранным функциям алгоритм определения фиктивности переменных $x_2, \dots, x_n$, мы получим все булевы функции, удовлетворяющие условию (4) и существенно зависящие от всех переменных.

Определение 1. Покомпонентной (n, k) функцией, где $1 \le k \le n$, будем называть булеву функцию $F(x_1, x_2, \dots, x_n)$, не содержащую фиктивных переменных и возвращающую при повторном применении k-тый аргумент, то есть удовлетворяющую условию:

$$F(x_1, x_2, \dots, x_{k-1}, F(x_1, x_2, \dots, x_n), x_{k+1}, \dots, x_n) = x_k \quad (6)$$

Сформулируем алгоритм генерации всех покомпонентных функций $F(x_1, x_2, \dots, x_n)$, существенно зависящих от n переменных и возвращающих при повторном применении переменную x_k. Входные данные алгоритма: n – число переменных, k – номер переменной, значение которой возвращает функция.

Алгоритм

Шаг 1. Ввод n, k.

Шаг 2. Вычисляем 2^{n-1} и генерируем всевозможные двоичные векторы длины 2^{n-1}. Для каждого вектора выполняем шаг 3.

Шаг 3. (проверка на фиктивность всех переменных). Для проверки используем алгоритм из [6]. Если проверка по всем переменным прошла успешно (все переменные существенные), то выбираем следующий вектор, выполняем шаг 3.

Иначе удаляем рассматриваемый вектор и выбираем следующий вектор, выполняем шаг 3.

Шаг 4. (дописывание инверсных частей) Выполняем для всех векторов, оставшихся после шага 3.

Разобьем вектор на 2^{k-1} частей и после каждой части допишем инверсную.

Включаем полученный вектор в ответ. Переходим к следующему вектору, шаг 4.

В предложенном алгоритме проверка на фиктивность переменных проводится по половине вектора значений. Если проверка прошла успешно, то и для полного вектора значений, сгенерированного в шаге 4, все переменные будут существенными.

Пример 3. Приведем пример работы алгоритма при $n=3$, $k=2$.

$2^{3-1}=4$, поэтому генерируем все двоичные векторы длины 4:
0000, 0001, 0010, 0011, …, 1111.

После шага 3 из 16 векторов останутся только 10 векторов, прошедших проверку:
0001,0010,0100,0110, 0111, 1000, 1001, 1011, 1101, 1110.

Выполняя шаг 4, разбиваем каждый вектор на 2 части, и к каждой части дописываем инверсию. Получим следующий результат:

(3,2) покомпонентные функции		(3,2) покомпонентные функции	
0001	00110110	1000	10010011
0010	00111001	1001	10010110
0100	01100011	1011	10011100
0110	01101001	1101	11000110
0111	01101100	1110	11001001

Покажем, что мы действительно получили функции, возвращающие вторую компоненту. Возьмем, например, $F_1(x_1, x_2, x_3)$, $\tilde{F}_1$=(00110110). Покажем, что при произвольных значениях x_1 и x_3 и при $x_2=a$ функция F_1 возвращает значение a.

x_1, x_3	$F_1(x_1, 0, x_3)$	$F_1(x_1, F_1(x_1, 0, x_3), x_3)$	$F_1(x_1,1,x_3)$	$F_1(x_1,F_1(x_1,1,x_3),x_3)$
00	0	0	1	1
01	0	0	1	1
10	0	0	1	1
11	1	0	0	1

Если в алгоритме взять $n=3$ и $k=1$, то первые шаги алгоритма будут такими же, как в примере 3. На шаге 4 оставшиеся 10 векторов не разбиваем (так как $2^{k-1}=2^0=1$), и дописав к ним инверсную часть, получим векторы значений *(3,1)* функций из примера 2, за исключением шести функций, содержащих фиктивные переменные.

Все покомпонентные *(n, k)* функции являются сбалансированными, то есть принимают значения 0 и 1 одинаково часто. Однако, вероятность замены либо сохранения символа открытого текста у них разная. Так, функция из примера 3 с вектором значений $\tilde{F}_1$=(00110110), изменяет значение открытого текста a лишь в 2 случаях из 8, то есть имеет вероятность замены символа 25%. Для *(3,1)* покомпонентных функций укажем эти вероятности, а также минимальные дизъюнктивные нормальные формы (МДНФ), в таблице 1.

Лучшими характеристиками (50% на 50%) обладают функции 4 и 7. В общем случае, идеальные характеристики вероятностей переходов имеют те булевы функции, которым соответствуют сбалансированные векторы, оставшиеся после шага 3 приведенного выше алгоритма.

Таблица 1. *(3, 1)* покомпонентные функции.

№ п/п	Покомпонентные функции		Вероятности переходов, %	
	$\tilde{F}$	МДНФ	0→0 1→1	0→1 1→0
1.	00011110	$\bar{x}$yz∨x$\bar{y}$∨x$\bar{z}$	75	25
2.	00101101	$\bar{x}$y$\bar{z}$ ∨x$\bar{y}$∨xz	75	25
3.	01001011	xy∨x$\bar{z}$∨x$\bar{y}$z	75	25
4.	01101001	$\bar{x}\bar{y}$z∨$\bar{x}$y$\bar{z}$∨x$\bar{y}\bar{z}$ ∨xyz	50	50
5.	01111000	$\bar{x}$y∨$\bar{x}$z∨x$\bar{y}\bar{z}$	25	75
6.	10000111	xy∨xz∨$\bar{x}\bar{y}\bar{z}$	75	25
7.	10010110	$\bar{x}\bar{y}\bar{z}$∨$\bar{x}$yz∨x$\bar{y}$z∨xy$\bar{z}$	50	50
8.	10110100	$\bar{x}$y∨$\bar{x}\bar{z}$∨x$\bar{y}$z	25	75
9.	11010010	$\bar{x}\bar{y}$ ∨$\bar{x}$y∨x$\bar{z}$	25	75
10.	11100001	$\bar{x}\bar{y}$∨$\bar{x}\bar{z}$∨xyz	25	75

Пример 4. При *n*=4 из 256 сгенерированных на шаге 2 векторов проверку на отсутствие фиктивных переменных проходят только 220. Из них сбалансированы 58 векторов. При *n*=4 сбалансирован и проходит проверку на отсутствие фиктивных переменных, например, вектор 11010100. Значит, *(4,1)* покомпонентная функция с вектором значений 11010100**00101011** имеет вероятность переходов 50% на 50%. Такую же вероятность переходов имеют *(4,2)* покомпонентная функция с вектором значений 1101**0010**0100**1011**, *(4,3)* покомпонентная функция с вектором значений 11**00**01**10**01**10**00**11**, *(4,4)* покомпонентная функция с вектором значений 1**0**1**0**0**1**1**0**0**1**1**0**0**1**0**1**.

Пусть r – неотрицательное число, меньшее n. Булева функция f от n переменных называется r-устойчивой, если любая её подфункция, полученная фиксацией не более r переменных, является сбалансированной.

Теорема 2. Для покомпонентной *(n,k)* функции $F(x_1, x_2, \dots, x_n)$ следующие условия эквивалентны:

1. $F(x_1, x_2, \dots, x_n)$ имеет вероятность замен 50%;
2. вектор значений функции $F(x_1, x_2, \dots, x_{k-1}, 0, x_{k+1}, \dots, x_n)$ сбалансирован;
3. $F(x_1, x_2, \dots, x_n)$ является 1-устойчивой.

Доказательство.

Докажем 1⇒2. Длина вектора значений функции $F(x_1, x_2, \dots, x_{k-1}, 0, x_{k+1}, \dots, x_n)$ равна 2^{n-1}. Пусть вектор значений функции $F(x_1, x_2, \dots, x_{k-1}, 0, x_{k+1}, \dots, x_n)$ имеет t единиц. Нужно доказать, что $t = 2^{n-2}$, то есть, что число единиц составляет ровно половину от длины вектора. Так как выполняется (6), то вектор значений функции $F(x_1, x_2, \dots, x_{k-1}, 1, x_{k+1}, \dots, x_n)$ имеет t нулей. Тогда общее число замен равно $2t$. По условию, вероятность замен 50%, то есть $2t = 2^{n-1}$, откуда $t = 2^{n-2}$.

Докажем 2⇒3. Отметим, что именно вектор значений функции $F(x_1, x_2, \dots, x_{k-1}, 0, x_{k+1}, \dots, x_n)$ мы получаем на шаге 3 предложенного выше алгоритма. Так как по условию он сбалансирован, то сбалансирован будет и инверсный к нему вектор $F(x_1, x_2, \dots, x_{k-1}, 1, x_{k+1}, \dots, x_n)$.

Рассмотрим теперь функцию $F(0, x_2, \dots, x_{k-1}, x_k, x_{k+1}, \dots, x_n)$. Ее можно разбить на две подфункции: $F(0, x_2, \dots, x_{k-1}, 0, x_{k+1}, \dots, x_n)$ и $F(0, x_2, \dots, x_{k-1}, 1, x_{k+1}, \dots, x_n)$, причем вторая будет инверсна первой. Значит, объединенный вектор $F(0, x_2, \dots, x_{k-1}, x_k, x_{k+1}, \dots, x_n)$ имеет равное число нулей и единиц. Аналогично доказывается, что сбалансирован вектор значений функции $F(1, x_2, \dots, x_{k-1}, x_k, x_{k+1}, \dots, x_n)$. По той же причине сбалансированы векторы значений

подфункций, которые получены фиксацией одной из переменных $x_2, \dots, x_{k-1}, x_{k+1}, \dots, x_n$. Значит, $F(x_1, x_2, \dots, x_n)$ является 1-устойчивой.

Докажем 3⇒1. Из условия следует, что функция $F(x_1, x_2, \dots, x_{k-1}, 0, x_{k+1}, \dots, x_n)$ является сбалансированной, то есть принимает 2^{n-2} раз значение 1 и 2^{n-2} раз значение 0. Также функция $F(x_1, x_2, \dots, x_{k-1}, 1, x_{k+1}, \dots, x_n)$ является сбалансированной, то есть принимает 2^{n-2} раз значение 1 и 2^{n-2} раз значение 0. Тогда число замен равно 2^{n-1}, что означает вероятность замен 50%.

Теорема доказана.

3. Варианты шифрования на основе покомпонентных булевых функций

Один из вариантов шифрования с применением *(3,1)* покомпонентных булевых функций был предложен нами в работе [4], и схематически изображен на рисунке 1. Однако, для использования покомпонентных *(n,k)* функций при шифровании, требуется одновременно иметь или генерировать *n –1* ключевую двоичную последовательность, что не всегда удобно. Вместо этого можно генерировать одну псевдослучайную последовательность (ПСП) элементов поля $GF(2^{n-1})$. Для этого можно использовать, например, генератор ПСП на основе регистров сдвига с линейными обратными связями (РСЛОС).

Пусть $F(x_1, x_2, \dots, x_n)$ – покомпонентная *(n,1)* функция, *a* –двоичный открытый текст, *K* – ключевая ПСП элементов поля $GF(2^{n-1})$. Тогда зашифрование текста *a* выполняется поэлементно согласно формуле:

$$E(a) = F(a, K). \quad (7)$$

Для расшифрования используем те же функцию $F(x_1, x_2, \dots, x_n)$ и ключевую последовательность *K*:

$$D(E(a)) = F(F(a, K), K) = a. \quad (8)$$

Организация вычислений во многом зависит от представления элементов поля $GF(2^{n-1})$, поэтому рассмотрим этот вопрос более подробно. Операции над элементами конечного поля $GF(2^m)$ легко производить, когда они образуют таблицу индексов, где степеням примитивного элемента сопоставлены *m*-мерные двоичные векторы. Такое представление удобно также при разбиении элементов поля на круговые классы, используемые, например, для нахождения примитивных многочленов или в алгоритмах получения числа помехоустойчивых кодов [7].

Рассмотрим алгоритм построения таблицы индексов поля $GF(2^m)$, где $m \leq 30$. Входные данные: *m* – степень расширения, *f(x)* – примитивный многочлен степени *m* над *GF(2)*. По окончании работы алгоритма в памяти программы содержатся все 2^m-1 ненулевых векторов длины *m* в определенном порядке, а именно по степеням примитивного элемента α – корня многочлена *f(x)*.

Для оптимизации скорости вычислений будем хранить каждый вектор коэффициентов в 32-битном целочисленном типе данных. На позиции *i*-го бита в двоичной записи числа будет храниться *i*-ый коэффициент вектора. За счет этого, умножение вектора на α будет осуществляться побитовым сдвигом влево. Поиск остатка от деления на примитивный многочлен *f(x)* будем выражать через операцию побитового сложения по модулю 2. Таким образом, для хранения таблицы индексов $GF(2^m)$ потребуется порядка $4 \cdot 2^m$ байт памяти.

Заметим, что вычисление каждого следующего вектора коэффициентов производится последовательно. Таким образом, заполняются строки 0, 1, 2, …, 2^m-3, 2^m-2. Для того, чтобы вычислять таблицу с использованием параллельных технологий, разобьем все строки на *k* последовательных блоков. Для вычисления *j*-ого блока понадобится вычислить вектор коэффициентов, стоящий первым в этом блоке. Для этого нет необходимости находить все предыдущие векторы. Используем алгоритм бинарного возведения в степень для значительного ускорения вычислений.

Была протестирована работа программы построения таблицы индексов поля $GF(2^m)$, где *m*=26, 27, 28, 29 и 30. Вычисления производились на 4 ядрах на кластере САФУ, имеющем 20

вычислительных узлов, на каждом из которых установлено 2 десятиядерных процессора Intel Xeon и 64 ГБ ОЗУ. Таблица содержит данные по времени работы при различных *m* и при различном числе потоков.

Таблица 2. Время выполнения последовательного и параллельного алгоритмов (в секундах).

Потоки \ m	**26**	**27**	**28**	**29**	**30**
Последовательный алгоритм	0,507с	1,003с	2,013с	4,034с	8,290с
1 поток	0,535с	1,077с	2,143с	4,283с	8,869с
2 потока	0,311с	0,624с	1,247с	2,495с	5,103с
3 потока	0,242с	0,473с	0,945с	1,910с	3,877с
4 потока	0,205с	0,403с	0,794с	1,597с	3,283с

Из представленной таблицы можно сделать вывод, что программа показывает ускорение примерно в 2,5 раза при параллельной реализации на 4 потока.

4. Заключение

Покомпонентные *(n,k)* функции расширяют режимы стандартных криптографических преобразований, в частности, ГОСТ Р 34.12-2015. Их использование вместо операции сложения по модулю 2 увеличивает возможности выбора раундовых преобразований для симметричных шифров. Дальнейшие исследования будут направлены на внедрение метода шифрования с использованием покомпонентных функций в программно-аппаратный комплекс.

5. Литература

[1] Токарева, Н.Н. Симметричная криптография. Краткий курс: учеб. пособие – Новосибирск: Новосибирский государственный университет, 2012. – 232 с.
[2] Селезнева, С.Н. Мультипликативная сложность некоторых функций алгебры логики // Дискретная математика. – 2014. – Т. 26, № 4. – С. 100-109.
[3] Городилова, А.А. От криптоанализа шифра к криптографическому свойству булевой функции // Прикладная дискретная математика. – 2016. – Т. 3, № 33. – С. 4-44.
[4] Vasilishin, I.I. Using component-wise function in cryptographical transformation algorithm from Russian national standard GOST R 34.12-2015 / I.I. Vasilishin, S.Yu. Korabelshchikova // CEUR Workshop Proceedings. – 2018. – С. 392-398.
[5] ГОСТ Р 34.12-2015. Информационная технология. Криптографическая защита информации. Блочные шифры – М.: Стандартинформ, 2015. – 25 с.
[6] Яблонский, С.В. Введение в дискретную математику: Учеб. пособие для вузов – М.: Наука, 2005. – 384 с.
[7] Мельников, Б.Ф. Алгоритмы получения числа помехоустойчивых кодов общего и специального вида / Б.Ф. Мельников, С.Ю. Корабельщикова // Информатизация и связь. – 2019. – Т. 1. – С. 55-60.

An extension of the class of Boolean functions used in symmetric cipher algorithms

S.Y. Korabelshchikova[1]

[1]Northern (Arctic) Federal University named after M.V. Lomonosov, Severnaya Dvina Emb. 17, Arkhangelsk, Russia, 163007

Abstract. One of the standard cryptographic transformations is the addition modulo two of a binary plaintext with a key binary sequence. In this paper, we have obtained a description of all Boolean functions of n arguments, suitable for use in cryptographic transformations instead of the addition modulo two function (we will call them component-wise Boolean functions). An example of their use in the algorithm of cryptographic transformation GOST R 34.12-2015 is also given. The paper proposes an algorithm for generating component-based Boolean functions of n variables at different values of n and k, where k is the number of the variable, the value of which is returned by the function. For the case when n=3 and k=1 or 2, all Boolean functions replacing the addition modulo two function are presented. The paper proposes an encryption method based on component-wise Boolean functions and a pseudorandom sequence generator of elements of the $GF(2^{n-1})$ field. The use of Boolean functions, that return the value of one of the arguments when reused, expands the variety of intermediate variants of round transformations and gives a new variable encryption method, consequently significantly increasing the cryptographic resistance of the cipher.