соответственно. Условия обратимости этих двух типов выглядят следующим образом: $\exists f \forall q \forall \alpha \forall \delta (f(\bar{\varphi}(\alpha\delta, q)) = \alpha)$ и $\exists f \forall q \forall \alpha \forall \delta (f(\bar{\varphi}(\alpha\delta, q), q) = \alpha)$.

Пример ещё одного типа обратимости автомата A содержится в условии

$$\exists f \forall \alpha \exists \delta \forall q (f(\bar{\varphi}(\alpha\delta, q), q, \psi(\alpha\delta, q), \delta) = \alpha).$$

Это есть условие обратимости степени $\forall\alpha\exists\delta\forall q$ и порядка $\upsilon(q, \alpha, \delta) = (q, \psi(\alpha\delta, q), \delta)$. В нём утверждается возможность восстановления функцией f префикса α входного слова $\alpha\delta$ автомата по его выходному слову $\bar{\varphi}(\alpha\delta, q)$ при известных начальном состоянии q, заключительном состоянии $t = \psi(\alpha\delta, q)$ и суффиксе δ входного слова в предположении, что в автомате для каждого префикса α входного слова суффикс δ этого слова не любой, но свой, и для этого входного слова $\alpha\delta$ начальное состояние q может быть любым.

Каждому типу обратимости с фиксированной задержкой ставится в соответствие класс автоматов, обратимых этого типа. Показано, что граф отношения включения между этими классами представляет собой объединение двадцати девяти решёток, где каждая решётка по определению есть частично упорядоченное множество с точными верхней и нижней гранями для каждой пары его элементов. Доказано, что автомат A обратим типа $(\forall q \forall \alpha \forall \delta, \upsilon(q, \alpha, \delta))$, если и только если

$$\forall q \forall \alpha \forall \delta \forall s \forall \beta \forall \varepsilon (\alpha \neq \beta \Rightarrow (\bar{\varphi}(\alpha\delta, q), \upsilon(q, \alpha, \delta)) \neq (\bar{\varphi}(\beta\varepsilon, s), \upsilon(s, \beta, \varepsilon)),$$

и что для любых символов кванторов $Q_i \in \{\forall, \exists\}$, $i \in \{1, 2, 3\}$, если автомат A обратим типа $(Q_1 x_1 Q_2 x_2 Q_3 x_3, \upsilon(q, \alpha, \delta))$, то

$$Q_1 x_1 Q_2 x_2 Q_3 x_3 Q_1 y_1 Q_2 y_2 Q_3 y_3 (\alpha \neq \beta \Rightarrow (\bar{\varphi}(\alpha\delta, q), \upsilon(q, \alpha, \delta)) \neq (\bar{\varphi}(\beta\varepsilon, s), \upsilon(s, \beta, \varepsilon)),$$

где x_1, x_2, x_3 и y_1, y_2, y_3 — различные переменные из множеств $\{q, \alpha, \delta\}$ и $\{s, \beta, \varepsilon\}$ соответственно, такие, что если x_i есть q, α или δ, то y_i есть s, β или ε соответственно, и если $x_i = \alpha$, то $Q_i = \forall$.

ЛИТЕРАТУРА

1. *Agibalov G. P.* Cryptanalytic concept of finite automaton invertibility with finite delay // Прикладная дискретная математика. 2019. № 44. С. 34–42.
2. *Агибалов Г. П.* Конечные автоматы в криптографии // Прикладная дискретная математика. Приложение. 2009. № 2. С. 43–73.
3. *Tao R.* Finite Automata and Application to Cryptography. N.Y.: Springer, 2009. 406 p.

УДК 519.7 DOI 10.17223/2226308X/12/27

О ВЕРОЯТНОСТЯХ РАЗНОСТНЫХ ТРАЕКТОРИЙ SPONGE-ФУНКЦИИ BASH-F

С. В. Агиевич, А. С. Маслов, Ю. С. Ярошеня

Предлагаются два метода оценки снизу весов разностных траекторий sponge-функции Bash-f. Оценки ограничивают сверху вероятности траекторий и могут использоваться при обосновании стойкости основанных на Bash-f криптографических алгоритмов к разностным атакам. Для полных 24-тактовых траекторий лучшая из оценок ограничивает вероятности величиной 2^{-386}.

Ключевые слова: *sponge-функция, S-блок, разностный криптоанализ, разностная траектория.*

1. Sponge-функции и разностные траектории

Sponge-функция — это бесключевая подстановка, действующая на двоичные слова достаточно большой (по криптографическим меркам) размерности. Располагая sponge-функцией, можно построить целую линейку симметричных криптографических алгоритмов разнообразного назначения — от древовидного хэширования до аутентифицированного шифрования. Эти алгоритмы достаточно просты и имеют высокий потенциал быстродействия на распространённых аппаратных платформах.

Обычная sponge-функция F преобразует слово $X \in \{0,1\}^n$ в слово $Y \in \{0,1\}^n$, выполняя d тактов (итераций)

$$X_i = F_i(X_{i-1}), \quad i = 1, 2, \ldots, d,$$

с $X_0 = X$ и $Y = X_d$. Здесь F_i — тактовые подстановки.

Слова X_i интерпретируются как векторы над двоичным полем $\mathbb{F}_2$. Преобразования F_i представляют собой композиции линейных и нелинейных преобразований этих векторов. Нелинейные преобразования, как правило, действуют локально на подвекторы $x \in \mathbb{F}_2^m$, где m невелико. Действие задаётся подстановками $S\colon \mathbb{F}_2^m \to \mathbb{F}_2^m$, которые в криптографии принято называть *S-блоками*.

В разностном криптоанализе рассматривается обработка не одного, а сразу двух слов: X и X'. Пусть $\Delta X = X \oplus X'$ — их сумма, ΔX_i — сумма слов, полученных в результате применения i тактов к X и X', $i = 1, 2, \ldots, d$. Поскольку сложение по модулю два равносильно вычитанию, суммы ΔX_i являются также *разностями* (отсюда и название — «разностный криптоанализ»). Ненулевые биты разностей будем называть *активными*.

Последовательность разностей ΔX, ΔX_1, ΔX_2, …, ΔX_r называется (r-тактовой разностной) *траекторией*. Траектории с нулевой входной разностью ΔX тривиальны (состоят из нулей), мы их далее не рассматриваем. Пусть траектория порождается случайными X и X' с фиксированной разностью ΔX. Возможность достаточно точного прогнозирования ΔX_r при r, близких к d, является предпосылкой для разностных атак. При обосновании стойкости F требуется оценить точность прогнозов и, в частности, получить оценки сверху для вероятностей траекторий.

Мы строили разностную траекторию, суммируя промежуточные результаты обработки входных слов. Можно поступить по-другому. Представим себе вероятностную машину M, которая работает исключительно с разностями, последовательно применяя к ним преобразования $F_1, \ldots, F_r$, точнее, их композиционные элементы. Линейные преобразования применяются к разностям (или их фрагментам) так же, как если бы речь шла о первоначальных словах. Остаётся рассмотреть действие S-блока S на фрагмент разности Δx. Если $\Delta x = 0$, то разность Δy на выходе S также нулевая. Если же $\Delta x = \alpha \neq 0$, то M выбирает Δy случайным образом в соответствии с вероятностным распределением

$$\mathsf{P}\{\Delta y = \beta \mid \Delta x = \alpha\} = \frac{1}{2^m} \sum_{u \in \mathbb{F}_2^m} \mathbf{I}\{S(u \oplus \alpha) \oplus S(u) = \beta\},$$

индуцируемым S (здесь $\mathbf{I}\{\mathcal{E}\}$ — индикатор наступления события $\mathcal{E}$).

Случайный выбор Δy мотивируется случайным выбором входов F, которые порождают разностную траекторию. Следует понимать, что M только моделирует разностные траектории. Но это моделирование достаточно точно, точность считается удовлетворительной в практических случаях.

Подача на вход S ненулевой разности Δx называется *активацией* S-блока. Активация является точкой ветвления — после неё выходная разность Δy определяется неоднозначно. Если активация произошла во время обработки разности ΔX_{i-1}, то речь идёт о нескольких вариантах следующей разности ΔX_i. Во время обработки ΔX_i могут происходить новые активации, которые приводят к новым ветвлениям, и так далее. Другими словами, M на фиксированном входе ΔX порождает несколько вариантов траекторий. Число вариантов, как правило, экспоненциально быстро растёт с увеличением длины траектории r.

Пусть во время генерации траектории $\Delta X, \Delta X_1, \ldots, \Delta X_r$ произошло s_A активаций S-блоков и p — произведение соответствующих вероятностей $\mathsf{P}\{\Delta y = \beta \mid \Delta x = \alpha\}$. Величина p и есть интересующая нас вероятность траектории.

Далее вместо p будет удобнее использовать характеристику $w_A = -\log_2 p$, которую называют *весом* траектории [1, 2]. Мы получим оценки снизу для веса траекторий sponge-функции Bash-f.

2. Sponge-функция Bash-f

Sponge-функция Bash-f введена в работе [3]. Функция преобразует двоичные слова длины $n = 1536$, называемые *состояниями*. Состояние разбивается на 24 подслова длины 64. Слова записываются в матрицу размера 3×8. Строки и столбцы матрицы называются *плоскостями*: горизонтальными и вертикальными.

Функция Bash-f построена как многократная композиция преобразований усложнения и перемешивания. За усложнение отвечает преобразование $S3$, которое применяется к тройке 64-разрядных слов. Тройки соответствующих друг другу битов этих слов (*вертикальные линии*) одновременно подвергаются действию фиксированного трёхбитового S-блока. Одновременность достигается формульным заданием $S3$ с помощью логических операций AND, OR, NOT и сложения XOR.

За перемешивание в Bash-f отвечают преобразования $L3$ и P. Первое преобразование также применяется к тройке 64-разрядных слов. Слова суммируются по правилу XOR, циклически сдвигаются, снова суммируются и т. д. Всего выполняется шесть сложений и четыре сдвига. Преобразование P переставляет местами 24 обрабатываемых слова.

Выполняется $d = 24$ такта. На каждом такте сначала к вертикальным плоскостям применяется $L3$, затем $S3$, затем плоскости перемешиваются с помощью P. Дополнительно, для разрушения однородностей, к 24-му слову добавляется тактовая константа.

S-блок $S\colon \mathbb{F}_2^3 \to \mathbb{F}_2^3$ выбран так, что для любого ненулевого $\alpha \in \mathbb{F}_2^3$ существует в точности четыре ненулевых $\beta \in \mathbb{F}_2^3$, для которых

$$\mathsf{P}\{\Delta y = \beta \mid \Delta x = \alpha\} = \frac{1}{4}.$$

Таким образом, вес w_A и число активных S-блоков s_A связаны соотношением $w_A = 2s_A$.

Пусть $s_A(r)$ — минимальное число активных S-блоков на нетривиальных траекториях Bash-f длины r и $w_A(r) = 2s_A(r)$ — минимальный вес траекторий.

3. Результаты

Мы получили оценки снизу для величины $w_A(24)$ двумя различными методами. Первый метод почти полностью автоматизирован, он реализуется компьютерной программой и, как всякий автоматизированный метод, имеет перспективы быстрого улучшения и точной верификации результатов. Второй метод почти полностью ручной, он

реализуется прямыми расчётами на основе известных свойств $L3$ и P и, как всякий ручной метод, опережает автоматизированный, если не требует вычислений большого объема. Методы появились в ходе разработки Bash-f, в настоящей работе они усиливаются.

В [1, 2] получены аналогичные оценки для величины $w_A(24)$, только применительно к sponge-функции Keccak-f (точнее, Keccak-f[1600]). Эта функция является ядром известного семейства алгоритмов хэширования SHA-3. В Keccak-f также выполняется $d = 24$ такта, длина состояния несколько больше: $n = 1600$.

Оценки представлены в таблице. Как видим, на сегодняшний день оценки для Bash-f лучше оценок для Keccak-f.

Оценки снизу для $w_A(24)$

Год, источник	Bash-f		Keccak-f
	метод 1	метод 2	
2012 [1]			296
2015 (разработка Bash-f)	300	324	
2017 [2]			368
2019 (настоящая работа)	386	372	

Опишем суть наших методов. Будем работать с характеристиками $s_A(r)$.

Метод 1. Оценивание $s_A(r)$ предполагает перебор разностных траекторий длины r. Прямой перебор затруднён уже при $r = 3$. Для сокращения перебора использовано сжатое описание траектории, фиксирующее только частичную информацию об их разностях. Такую частичную информацию мы назвали *проекцией*.

Использовались следующие проекции:

1) `avl` — число активных S-блоков (вертикальных линий);
2) `avp` — число активных (т. е. хотя бы с одним активным битом) вертикальных плоскостей;
3) `avls` — число активных S-блоков для каждой из вертикальных плоскостей (упорядоченный набор из восьми чисел).

Пусть $[\mathtt{p}](i)$ — проекция `p` разности i-го такта; $[\mathtt{p}_1, \ldots, \mathtt{p}_k](i, \ldots, j)$ — набор проекций $([\mathtt{p}_u](v))$, $1 \leqslant u \leqslant k$, $i \leqslant v \leqslant j$.

Некоторые проекции однозначно определяют другие. Например, сумма чисел в $[\mathtt{avls}](i)$ совпадает с $[\mathtt{avl}](i)$, количество ненулевых чисел в $[\mathtt{avls}](i)$ — с $[\mathtt{avp}](i)$. В этом смысле `avls` можно считать расширением проекций `avl` и `avp`. Очевидно, что сумма

$$[\mathtt{avl}](1) + \ldots + [\mathtt{avl}](r) \qquad (\star)$$

представляет собой число активных S-блоков подразумеваемой r-траектории, а минимум суммы $(\star)$ по всем допустимым наборам проекций и есть искомое значение $s_A(r)$.

Оценивание $s_A(r)$ выполняется в три этапа по схеме метода ветвей и границ. На первом (подготовительном) этапе рассчитываются характеристики линейного преобразования $L3$, которые позволяют отсеивать недопустимые наборы проекций на третьем этапе. На втором этапе перебираются проекции $[\mathtt{avl}, \mathtt{avp}](1, \ldots, r)$ и вычисляется минимум суммы $(\star)$. При этом наборы проекций, на которых он достигается, сохраняются. На третьем этапе перебираются расширенные проекции $[\mathtt{avls}](1, \ldots, r)$, соответствующие проекциям, сохранённым на втором этапе. Если для некоторых i и j не существует

допустимого набора $[\mathtt{avls}](i,\ldots,j)$, соответствующего $[\mathtt{avl},\mathtt{avp}](i,\ldots,j)$ из сохранённого набора, то набор $[\mathtt{avl},\mathtt{avp}](i,\ldots,j)$ помечается как недопустимый и вычисления возвращаются ко второму этапу для уточнения минимума.

Метод 2. Во втором методе оценивается снизу число активных S-блоков на четырёх тактах. Использованы свойства преобразований $L3$, $S3$, P, заложенные при проектировании Bash-f [3]. Особую важность имеет следующий факт: если на выходе $L3$ получена вертикальная плоскость с малым числом активных линий, то на вход была подана плоскость с большим числом активных линий. Базовые свойства преобразований выступают в роли аксиом, из которых аналитически выводятся теоремы — новые свойства, имеющие отношение к оцениванию.

Затем просматриваются допустимые конфигурации, представляющие собой пары «число активных вертикальных линий на входах $L3$ на втором такте, проекция $[\mathtt{avp}](2)$». Для каждой конфигурации оценивается снизу проекция $[\mathtt{avl}](1)$ (как будто бы от второго такта мы возвращаемся к первому) и проекции $[\mathtt{avl}](3,4)$. В итоге получена оценка

$$s_A(4) = \min\left([\mathtt{avl}](1) + [\mathtt{avl}](2) + [\mathtt{avl}](3) + [\mathtt{avl}](4)\right) \geqslant 31,$$

откуда $s_A(24) \geqslant 6s_A(4) \geqslant 186$.

ЛИТЕРАТУРА

1. *Daemen J. and Van Assche G.* Differential propagation analysis of Keccak // FSE'2012. LNCS. 2012. V. 7549. P. 422–441.
2. *Mella S., Daemen J., and Van Assche G.* New techniques for trail bounds and application to differential trails in Keccak // IACR Trans. Symmetric Cryptology. 2017. No. 1. P. 329–357.
3. *Agievich S., Marchuk V., Maslau A., and Semenov V.* Bash-f: another LRX sponge function // Математические вопросы криптографии. 2017. Т. 8. № 2. С. 7–28.

УДК 519.7 DOI 10.17223/2226308X/12/28

КРИПТОАНАЛИЗ ШИФРСИСТЕМЫ ACBF[1]

И. В. Боровкова, И. А. Панкратова

Рассматривается асимметричная шифрсистема на булевых функциях с функциональным ключом. Предлагаются атаки с известным открытым текстом для двух подмножеств ключевых параметров.

Ключевые слова: *криптосистема ACBF, векторные булевы функции, асимметричная криптосистема, криптоанализ.*

Рассматривается шифрсистема ACBF (Asymmetric Cryptosystem on Boolean Functions) [1]. Она строится на основе двух операций — перестановки и отрицания, — применяемых к переменным и координатам обратимой векторной булевой функции. Открытый ключ $f(x)$ получается по формуле $f(x) = \pi_2(g^{\sigma_2}(\pi_1(x^{\sigma_1})))$, где $g\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ — биективная векторная булева функция; функции g и g^{-1} известны; $\sigma_1, \sigma_2 \in \mathbb{F}_2^n$ — операции отрицания; $\pi_1, \pi_2 \in \mathbb{S}_n$ — операции перестановки. Закрытый ключ — функция f^{-1}.

Ключевыми параметрами криптосистемы являются элементы любого непустого подмножества $J \subseteq \{\pi_1, \pi_2, \sigma_1, \sigma_2\}$; операции из $\{\pi_1, \pi_2, \sigma_1, \sigma_2\}\setminus J$ считаются тождественными. Таким образом, существует 15 различных подмножеств ключевых параметров.

[1]Работа поддержана грантом РФФИ, проект № 17-01-00354.