УДК 519.7, 004.832.25 DOI 10.17223/2226308X/8/54

ПРИМЕНЕНИЕ АЛГОРИТМОВ РЕШЕНИЯ ПРОБЛЕМЫ БУЛЕВОЙ ВЫПОЛНИМОСТИ К КРИПТОАНАЛИЗУ ХЭШ-ФУНКЦИЙ СЕМЕЙСТВА MD[1]

И. А. Богачкова, О. С. Заикин, С. Е. Кочемазов, И. В. Отпущенников, А. А. Семёнов

Задачи поиска коллизий криптографических хэш-функций семейства MD рассматриваются как варианты задачи о булевой выполнимости (SAT). Для построения SAT-кодировок алгоритмов MD4 и MD5 использована система Transalg автоматической трансляции алгоритмических описаний дискретных функций в булевы уравнения. Полученные кодировки оказались существенно экономнее известных аналогов. В построенные SAT-кодировки хэш-функций добавлены дополнительные условия, кодирующие известные разностные атаки на данные функции. Время решения SAT-задач, кодирующих поиск одноблоковых коллизий для функции MD4, составило в среднем менее 1 с на ПК средней производительности. Для решения SAT-задач, кодирующих поиск двухблоковых коллизий для функции MD5, использованы параллельные SAT-решатели и вычислительный кластер. В результате был выделен класс двухблоковых коллизий для MD5 с 10 первыми нулевыми байтами. Построено несколько десятков коллизий такого типа. Рассмотрена также задача обращения хэш-функции MD4 (поиск прообраза для фиксированного хэша). В процессе решения данной задачи разработана техника, использующая так называемые «переменные переключения». Использование переменных переключения позволило найти новые дополнительные условия (типа «условий Доббертина»), учёт которых ускорил решение проблемы обращения 39-шагового варианта MD4 в сотни раз.

Ключевые слова: *криптографические хэш-функции, коллизии для хэш-функций, алгоритмы MD4, MD5, задача о булевой выполнимости, SAT.*

Хэш-функциями называются отображения вида $\chi : \{0,1\}^* \to \{0,1\}^c$, где c — некоторая натуральная константа. Хэш-функции используются в целом ряде разделов Computer Science для ускорения работы с данными. В криптографии хэш-функции являются основой большого числа различных криптосистем и протоколов. В отличие от «обычных» хэш-функций, к криптографическим хэш-функциям предъявляются дополнительные условия, требующие, чтобы задачи, связанные с обращением этих функций, были вычислительно трудными. Более точно, трудной должна быть собственно задача обращения: по известному значению хэша y найти вход x (обычно некоторой фиксированной длины), такой, что $\chi(x) = y$. Стандартным является также требование сложности задачи поиска коллизий. Поскольку константа c фиксирована, то для произвольного $n > c$ отображение $\chi : \{0,1\}^n \to \{0,1\}^c$ не может быть биективным. Если при этом χ определена всюду на $\{0,1\}^n$, то существуют такие различные $x_1, x_2 \in \{0,1\}^n$, что $\chi(x_1) = \chi(x_2)$. В этом случае говорят, что пара сообщений x_1, x_2 образует коллизию. Итак, еще одно требование к криптографическим хэш-функциям состоит в том, что задача поиска коллизий должна быть вычислительно трудной.

Одной из наиболее распространённых основ для криптографических хэш-функций является конструкция Меркля — Дамгарда [1, 2]. В соответствии с данной конструкцией двоичный вход x разбивается на блоки фиксированной длины n (этот параметр за-

[1]Работа выполнена при частичной поддержке грантов РФФИ (№ 14-07-31172 мол-а, 14-07-00403 а и 15-07-07891 а); стипендий Президента РФ СП-3667.2013.5, СП-1184.2015.5; Совета по грантам Президента РФ для гос. поддержки ведущих научных школ (НШ-5007.2014.9).

даётся в спецификации алгоритма). При необходимости исходное сообщение дополняется незначащей информацией для получения слова, длина которого кратна n. Пусть x — входное слово, разбитое на N блоков, каждый из которых имеет длину n:

$$x = M_1|\dots|M_N$$

(подразумевается конкатенация блоков). Процесс построения хэша $y = \chi(x)$ — это итеративная процедура вычисления «функции сжатия» (compression function):

$$\chi_i = f(\chi_{i-1}, M_i), \quad i \in \{1, \dots, N\}, \quad \chi_N = y.$$

Для произвольного $i \in \{1, \dots, N\}$ слово χ_i имеет длину c, которая задаётся в спецификации алгоритма; там же задаётся начальное значение (Initial Value) χ_0. Функция сжатия f — это обычно сложная функция, представляемая в виде суперпозиции «раундовых» функций.

Наиболее известными криптографическими хэш-функции, базирующимися на конструкции Меркля — Дамгарда, являются представители семейств MD [3] и SHA. В хэш-функциях MD4 и MD5 $c = 128$, а длина одного блока входного сообщения равна 512 битам. На сегодняшний день, насколько можно судить из открытых источников, даже задача обращения MD4 не имеет эффективного решения для случайных 512-битных входов. Мы не рассматриваем здесь ситуации, связанные с подбором паролей, когда относительно короткий пароль или его «дубликат» можно восстановить по хэшу полным перебором или с использованием различных стратегий пространственно-временного компромисса, таких, например, как Rainbow-метод.

С другой стороны, хорошо известны примеры успешного построения коллизий для функций MD4 и MD5. Первыми работами, в которых продемонстрирован метод, позволяющий устойчиво порождать коллизии для этих функций, являются X. Wang с соавторами [4, 5]. В дальнейшем «метод Wang» неоднократно совершенствовался и модифицировался. К сожалению, мы не можем здесь перечислить все соответствующие ссылки, так как это займёт слишком много места. Метод Wang представляет собой разновидность разностной атаки и может рассматриваться как развитие идей дифференциального криптоанализа [6] (по крайней мере, именно так его позиционирует сама X. Wang). Суть метода Wang заключается в случайном выборе некоторого сообщения с последующей его модификацией, цель которой состоит в «подгонке» получаемой пары сообщений под дифференциальный путь. В [7] предложено использовать алгоритмы решения проблемы булевой выполнимости (SAT) для автоматизации этапа подгонки. С использованием SAT-решателя minisat в [7] довольно эффективно удавалось находить одноблоковые коллизии для хэш-функции MD4. Задача поиска двухблоковых коллизий для MD5 оказалась тем не менее весьма сложной (текст работы [7] не позволяет точно определить ни время, за которое удавалось находить коллизии для MD5, ни число найденных коллизий). Как это ни странно, но результаты работы [7] не получили дальнейшего развития. Это особенно удивительно в свете интенсивного развития технологий решения SAT-задач, наблюдаемого в последние годы [8].

Следующий шаг в использовании SAT-подхода для поиска коллизий хэш-функций семейства MD был сделан, по всей видимости, только в 2014 г. авторами настоящей работы. Полученные в этом направлении результаты опубликованы в [9]. Дадим краткое их описание. Во-первых, отметим основной недостаток работы [7], который состоит в том, что для построения SAT-кодировок использовались узкоспециальные средства кодирования схемных представлений булевых функций. В [9] для этой цели использована система Transalg [10] автоматической трансляции в SAT алгоритмов вычисления

дискретных функций. Полученные кодировки оказались существенно экономнее кодировок, построенных в [7]. Время поиска одноблоковых коллизий для MD4 составило менее 1 с в среднем в сравнении с 10 мин в [7]. При помощи параллельных SAT-решателей в [9] построены семейства двухблоковых коллизий для хэш-функции MD5. Более того, были выделены коллизии специального вида, а именно начинающиеся с 10 нулевых байт (10 примеров таких коллизий приведены в приложении к [9]).

Помимо перечисленного, рассмотрена задача обращения хэш-функции MD4. Ранее к этой задаче также применялся SAT-подход [11]. Однако, как уже отмечалось, до сих пор никому не удалось осуществить успешное обращение полнораундовой версии MD4. В этом смысле рекордное значение равно 39 шагам алгоритма (число шагов в неослабленной версии MD4 равно 48). Атака на версию MD4 с 39 шагами, описанная в [11], требовала около 8 часов работы решателя minisat. При этом в SAT-кодировку добавлялись дополнительные ограничения на некоторые переменные сцепления. Эти ограничения выбирались похожими на ограничения, описанные в [12]. Мы разработали новую технику генерации дополнительных ограничений такого рода. Основная идея состоит в следующем. Пусть C — КНФ, кодирующая задачу обращения некоторой функции χ, и X — множество переменных, фигурирующих в C. Предположим, что к C требуется добавить ограничения, задающие некоторый предикат над переменными из множества $\tilde{X}$, $\tilde{X} \subseteq X$. Пусть $R(\tilde{X})$ — формула, задающая данный предикат. Введём в рассмотрение новую переменную $u \notin X$. Рассмотрим формулу $C \wedge (\overline{u} \vee R(\tilde{X})$. Очевидно, что ограничение $R(\tilde{X})$ учитывается, когда переменная u принимает значение 1. При $u = 0$ ограничение $R(\tilde{X})$ не активно. Переменные типа u называются переменными переключения. В результате варьирования различных значений переменных переключения можно найти набор дополнительных ограничений, учёт которых позволит существенно повысить скорость решения SAT-задачи, кодирующей обращение рассматриваемой функции. При этом для работы с переменными переключения можно использовать различные техники «обучения», встроенные в современные SAT-решатели. В частности, можно рассматривать значения переменных переключения как «assumptions» [13]. На данном этапе мы применили довольно простой механизм перебора значений переменных переключения. В результате найдены ограничения, отличающиеся от приведённых в [11]. Использование новых ограничений сократило время обращения 39-шаговой версии MD4 с нескольких часов до нескольких секунд. В ближайшем будущем мы планируем развивать технику работы с переменными переключения в направлении перебора существенно более широких (чем на текущий момент) множеств дополнительных ограничений.

ЛИТЕРАТУРА

1. *Merkle R. A.* Certified digital signature // LNCS. 1990. V. 435. P. 218–238.
2. *Damgard I. A.* A design principle for hash functions // LNCS. 1990. V. 435. P. 416–427.
3. *Rivest R. L.* The MD4 Message Digest Algorithm // LNCS. 1991. V. 537. P. 303–311.
4. *Wang X., Lai X., Feng D., et.al.* Cryptanalysis of the hash functions MD4 and RIPEMD // LNCS. 2005. V. 3494. P. 1–18.
5. *Wang X. and Yu H.* How to break MD5 and other hash functions // LNCS. 2005. V. 3494. P. 19–35.
6. *Biham E. and Shamir A.* Differential cryptanalysis of DES-like cryptosystems // J. Cryptology. 1991. V. 4. No. 1. P. 3–72.
7. *Mironov I. and Zhang L.* Applications of SAT solvers to cryptanalysis of hash functions // LNCS. 2006. V. 4121. P. 102–115.

8. *Biere A., Heule V., van Maaren H, and Walsh T.* Handbook of Satisfiability. Amsterdam: IOS Press, 2009.
9. *Богачкова И. А., Заикин О. С., Кочемазов С. Е., Отпущенников И. В., Семёнов А. А.* Задачи поиска коллизий для криптографических хеш-функций семейства MD как варианты задачи о булевой выполнимости // Вычислительные методы и программирование. 2015. Т. 16. С. 61–77.
10. *Отпущенников И. В., Семёнов А. А.* Технология трансляции комбинаторных проблем в булевы уравнения // Прикладная дискретная математика. 2011. № 1. С. 96–115.
11. *De D., Kumarasubramanian A., and Venkatesan R.* Inversion attacks on secure hash functions using SAT solvers // LNCS. 2007. V. 4501. P. 377–382.
12. *Dobbertin H.* The first two rounds of MD4 are not One-Way // Proc. 5th Intern. Workshop Fast Software Encryption. London, UK: Springer Verlag, 1998. P. 284–292.
13. *Nadel A. and Ryvchin V.* Efficient SAT solving under assumptions // LNCS. 2012. V. 7317. P. 242–255.

УДК 519.178 DOI 10.17223/2226308X/8/55

ВЫЧИСЛЕНИЕ ВЕРХНЕЙ ОЦЕНКИ ВЕРШИННОЙ ЦЕЛОСТНОСТИ ГРАФА НА ОСНОВЕ МИНИМАЛЬНЫХ СЕПАРАТОРОВ

В. В. Быкова, Ю. И. Кириллов

Рассматривается трудно вычисляемый числовой параметр графа, называемый вершинной целостностью и используемый в анализе и синтезе отказоустойчивых сложных технических систем. Для нахождения данного параметра необходимо знание всех сепараторов исходного графа. Предлагается алгоритм, который ограничивается построением и анализом только всех минимальных сепараторов. Поэтому алгоритм даёт верхнюю оценку вершинной целостности графа. Вычислительная сложность предлагаемого алгоритма полиномиально зависит от числа вершин и числа минимальных сепараторов графа. Результаты экспериментов показали, что вычисленные оценки являются хорошими и часто достижимыми.

Ключевые слова: *алгоритмы на графах, вершинная целостность графа, минимальные сепараторы.*

Все последние десятилетия проблеме исследования мер целостности графов уделяется особое внимание, поскольку она тесно связана с вопросами анализа и синтеза сложных технических систем, для которых отказоустойчивость является важнейшим показателем качества функционирования. Вершинная целостность — одна из детерминированных мер целостности графа [1].

Пусть $G = (V, E)$ — простой связный граф с множеством вершин V и множеством рёбер E, при этом $n = |V| \geqslant 1$ — порядок графа G. Под вершинной целостностью (*Vertex Integrity*) графа G понимается числовой параметр $I(G)$, вычисляемый по формуле

$$I(G) = \min_{S \subseteq V} \{|S| + w(G - S)\}, \tag{1}$$

где $w(G - S)$ — порядок наибольшей компоненты связности графа $G - S$, который получается из G удалением всех вершин, входящих в $S \subseteq V$. Множество S, для которого достигается равенство в (1), принято называть I-множеством. В неполном связном графе G всякое I-множество является сепаратором этого графа [2]. Доказано [3], что задача вычисления вершинной целостности графа NP-трудная. Ввиду высокой вы-