

*А. А. Варфоломеев, К. Г. Когос, А. М. Коренева, В. М. Фомичев*

# О СЛОЖНОСТИ РЕАЛИЗАЦИИ НЕКОТОРЫХ АЛГОРИТМИЧЕСКИХ МЕТОДОВ КРИПТОАНАЛИЗА С ПОМОЩЬЮ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛЕНИЙ

**Метод согласования**

Приведем основные результаты по методу согласования [1, 2], рассмотренному в условиях различных математических моделей распределенных вычислений (РВ). В отчете по НИР дано более детальное обоснование оценок среднего времени реализации метода и размера требуемой памяти вычислителя.

В кластерной модели РВ, обозначаемой 1.1РВ, каждый из $2^k$ вычислителей имеет одинаковые производительность и память и допускается возможность активного обмена данными между вычислителями. Время определения ключа может быть сокращено до $2^k$ раз по сравнению с однопроцессорной вычислительной системой, если время пересылки данных между вычислителями не слишком велико, а размер требуемой памяти вычислителя составляет $2^{n/2-k}$ ячеек, в которые записываются элементы $V_{n/2}$ — опробуемые ключи. Надежность метода равна 1.

В модели 1.2РВ процессом вычислений управляет координатор, который раздает задания отдельным узлам сети и объединяет результаты вычислений, при этом допускается существенно большее число участников. Величина сокращения трудоемкости алгоритма согласования по сравнению с алгоритмом полного опробования ключей не больше, чем $2^p$ раз, где $2^p$ — число вычислителей, и определяется, в первую очередь, соотношением величин $\tau_o$ (время обращения к памяти) и $\max\{\tau_з,\tau_п\}$ (времена записи и пересылки данных соответственно). Координатору достаточно иметь $2^{n/2}$ ячеек, в которые записываются элементы $V_{n/2}$.

В модели 1.3РВ, объединяющей подходы 1.1РВ и 1.2РВ, система использует $2^p$ участников, $\rho \le m$, и кластерную подсистему координатора. Трудоемкость $T(m)$ алгоритма в целом определяется величиной порядка $\max\{2^{m-\rho}(\tau_з+\tau_п),2^{n-m-\rho}(\tau_р+\tau_п),2^{m-k}\tau_o,2^{n-m-k}\tau_o\}$. Минимум трудоемкости достигается при $m=[n/2]$ и имеет величину порядка $\tau 2^{n/2-k}$, что может быть меньше в несколько раз, чем при кластерных вычислениях. Коэффициент сокращения определяется соотношением скоростей шифрования, пересылки данных и обращения к памяти. Кластерному вычислителю достаточно иметь $2^{n/2-k}$ ячеек, в которые записываются элементы $V_{n/2}$.

Оптимизация процесса опробования ключей заключается в таком распределении вычислительного задания между участниками, при котором вычислительная нагрузка на все узлы сети примерно одинакова, в этом случае каждый участник затратит приблизительно одинаковое время на выполнение вычислительного задания. Рассмотрена также модель с постоянным числом

участников и разной производительностью их ЭВМ. Сложность опробования и вероятность использования всех ключей одинаковы.

Рассмотрим пример. Пусть число участников $N = 10^6$, соотношение производительностей составляет $\frac{\pi_1}{\pi_2} = 10$. Пусть две группы имеют одинаковое число участников по $5 \cdot 10^5$ в каждой. Общая производительность сети составляет: $\pi_\Sigma = 5 \cdot 10^5(\pi_1+\pi_2) = 55 \cdot 10^5 \cdot \pi_2$. Оптимальное разбиение опробуемого ключевого множества имеет вид:

$$N_1^1 = [0,...,\frac{K}{55 \cdot 10^4} - 1],...,N_1^{N_1} = [\frac{(5 \cdot 10^5 - 1) \cdot K}{55 \cdot 10^4},...,\frac{10 \cdot K}{11} - 1];$$

$$N_2^1 = [\frac{10 \cdot K}{11},...,\frac{10 \cdot K}{11} + \frac{K}{55 \cdot 10^5} - 1],...,N_2^{N_2} = [\frac{10 \cdot K}{11} + \frac{K}{55 \cdot 10^5},...,K - 1].$$

Если опробуются ключи с разной вычислительной сложностью, то для обеспечения равномерной нагрузки на каждый вычислительный узел вычислитель с номером $i$ должен опробовать такое число ключей $X$, чтобы $\frac{\sum_{l=1}^{X} c_l}{\pi_i} \le \frac{c_\Sigma}{\pi_\Sigma}$, где $\pi_\Sigma = \Sigma_{i=1}^{N} \pi_i$ — общая производительность сети, $c_\Sigma = \Sigma_{j=1}^{K} c_j$ — суммарная сложность всех ключей. В случае переменной вероятности использования ключа начинать опробование следует с наиболее вероятных ключей (за один такт работы сети должно быть опробовано подмножество неопробованных ключей с максимальной вероятностью). Поэтому оптимальное распределение ключей среди участников следует выполнить, например, обходя ключи в порядке убывания вероятностей, при этом порядок номеров соответствующих участников описывается маршрутом, чередующим обход от 1 до $N$ и обратный обход от $N$ до 1. Таким образом, у каждого участника должно оказаться по $K_i$ ключей.

**Метод последовательного опробования**

Метод последовательного опробования (ПО) элементов ключа реализует более экономное опробование по сравнению с методом полного опробования для некоторых криптосистем [3,4]. Метод не универсален, он применим лишь к тем системам уравнений, где некоторые уравнения из левой части зависят не от всех переменных. Реализация метода не требует значительной памяти, надежность метода равна 1.

В работе построена наилучшая по трудоемкости схема ПО с использованием РВ для решения систем уравнений общего вида, в левой части которых найдется функция, не зависящая от некоторой переменной. Пусть $2^{[n]}$ — решетка всех подмножеств множества $\{1,\ldots,n\}$. Тогда вычислительная сложность метода ПО определяется для системы уравнений общего вида парой цепей $(C,\psi_F(C))$ в решетках $2^{[m]}$ и $2^{[n]}$ соответственно, где $C$ — максимальная $F$-приведенная цепь в $2^{[m]}$, при которой выражение трудоемкости принимает наименьшее значение. При больших $n,m$ поиск такой цепи является в общем случае сложной вычислительной задачей.

Построен алгоритм решения одного класса систем уравнений треугольно-ступенчатого вида:

$$\begin{cases} f_1(x_1,...,x_p) = a_1 \\ \quad ... \\ f_q(x_1,...,x_p) = a_q \\ f_{q+1}(x_1,...x_{2p}) = a_{q+1} \\ \quad ... \\ f_{2q}(x_1,...,x_{2p}) = a_{2q} \\ \quad ... \\ f_{hq}(x_1,...,x_{hp}) = a_{hq} \end{cases}.$$

При реализации алгоритма в условиях РВ первые $s$ этапов вычислительного процесса выполняет координатор, остальные этапы выполняют $N$ участников. Средняя трудоемкость алгоритма с использованием РВ оценивается величиной:

$$T_{\text{по/РВ}} = k^p \frac{k^{s(p-q)} - 1}{k^{p-q} - 1} + k^p \frac{k^{h(p-q)} - k^{s(p-q)}}{N(k^{p-q} - 1)}.$$

Таким образом, среднее время решения системы уравнений треугольно-ступенчатого вида с использованием РВ с числом процессоров $N < k^{q + h(\rho - q)/2}$ может быть сокращено приблизительно в $N$ раз по сравнению с однопроцессорной вычислительной системой.

Построены алгоритмы эффективного применения метода ПО в условиях широкого класса математических моделей РВ. В частности, рассмотрены математические модели, где вычислители обладают различной производительностью, а также схемы, где распределение ключевого множества не является равномерным.

### Метод Хеллмана и ему подобные методы

Оригинальный метод опробования ключей шифрсистем был предложен М. Хеллманом в 1980 г. Метод состоит из предварительного этапа (построение таблиц) и оперативного этапа (нахождение ключа) [5].

Напомним кратко суть метода. Преобразование открытого текста $M_0$ в шифртекст $C_0$ на ключе $k_0$ будем обозначать как $C_0 = E_{(k_0)}(M_0)$. $L$ — размер открытого и шифрованного текстов, $n$ — размер ключа. Обозначим через $R = R_{(L,n)}$ отображение (редукции): $V_L \to V_n$. Через $f(k)$ обозначим функцию вида $f(k) = R(E_k(M_0))$. На предварительном этапе случайно выбираются $m$ стартовых точек $S\rho_1, \ldots, S\rho_m$ из множества возможных ключей. Для выбранного значения $t$ вычисляются значения матрицы размера $(m \times t)$ по правилу $K_{(i,\,0)} = S\rho_i$. $(i = 1,\ldots, m)$. $K_{(i,j)} = f(K_{(i,\,j-1)})$ $(j = 1,\ldots, t)$. Только пары $(S\rho_i, E\rho_i)$, $(i = 1,\ldots, m)$, хранятся в памяти, отсортированные по $E\rho_i$. Оперативный этап использует полученные пары. Применяя редукцию $R$ к левой и правой частям соотношения $C_0 = E_{(k_0)}(M_0)$, получим равенство: $R[C_0] = R[E_{(k_0)}(M_0)]$, правая часть которого равна $f(k_0)$ по определению функции $f(k)$. Обозначим левую часть через $Y_1 = R(C_0) = R(E_k(M_0)) = f(k_0)$. Значение $Y_1$ ищется среди значений конечных точек $E\rho_i (i=1, \ldots, m)$ в хранимых в памяти парах $(S\rho_i, E\rho_i)$, $(i = 1,\ldots, m)$. Если $Y_1$ равен некоторому $E\rho_i$, то искомый ключ $k_0$ может быть равен $K_{(i,\,t-1)}$. Этот ключ не хранится в памяти, и для его нахождения и проверки используется его стартовая точка. Если $Y_1$ не совпадает ни с одной из конечных точек, то поиск переносится на ключи $(t - 2)$-го столбца матрицы. И так далее.

Вероятность успеха нахождения ключа близка по порядку к 1, если на предварительном этапе генерируется множество таблиц и для каждой таблицы выбрана уникальная функция редукции $R$. Метод Хеллмана может быть обобщен и применен для «инвертирования» произвольной однонаправленной функции $f:\{0,1\}^n \to \{0,1\}^n$.

«Метод характерных точек» развивает метода Хеллмана. Существенно то, что цепочки ключей на предварительном этапе строятся не на фиксированную длину, а до тех пор, пока не найден ключ (точка) с характерной структурой. Экономия трудоемкости заключается в том, что во время оперативного этапа не производится поиск по таблицам.

«Радужный метод» позволяет на практике сократить время оперативного этапа восстановления неизвестного параметра в 2 раза по сравнению с классическим методом Хеллмана. Суть метода в использовании различных функций при построении цепочек (таблиц) ключей в матрице метода Хеллмана.

Для обоих этапов указанных методов можно использовать РВ. Основным способом распараллеливания вычислений является выбор начальных точек цепочек ключей. При

неограниченной памяти координатор хранит результаты предварительного этапа и рассылает каждому участнику порцию данных цепочек (начальную и конечную точки) для обработки на оперативном этапе. Общее время обоих этапов существенно сокращается при РВ за счет увеличения общей производительности.

При нестационарной модели вычислений координатор с помощью периодических опросов участников вычислений определяет производительность каждого из участников и время, которое следует выделить на решение задачи. После этого координатор формирует и отправляет участнику порцию пар точек и таблиц цепочек для обработки.

При ограниченной памяти ЭВМ у координатора часть участников вычислений следует задействовать для получения пар точек, используемых в цепочках предварительного этапа.

В задаче определения прообраза для однонаправленной функции важную роль играют вероятности использования каждого прообраза. Однонаправленные функции (функции хеширования) используются в парольных системах, где пароли и являются элементами области определения функции. Различные пароли имеют разную вероятность использования, например, более короткие — более вероятны. В условиях неравновероятности прообразов методы типа Хеллмана требуют особенностей получения цепочек (строк) для таблиц и их использования на оперативном этапе. Каждой цепочке может быть поставлена в соответствие суммарная вероятность прообразов ее составляющих. Чем выше суммарная вероятность, тем предпочтительнее обработать цепочку на оперативном этапе в первую очередь.

Существенную роль в достижении выигрыша играет координатор вычислений, который контролирует равномерную загрузку всех участников сети. При стационарной модели это делать проще, при нестационарной модели обработки имеется необходимость получения информации о готовности участников вычислений и их характеристиках.

**Вывод**

Сложность реализации метода согласования, метода последовательного опробования и метода Хеллмана для опробования ключей при определенных условиях может быть существенно сокращена с помощью распределенных вычислений.

## СПИСОК ЛИТЕРАТУРЫ:

1. *Фомичёв В. М.* Методы дискретной математики в криптологии. М.: ДИАЛОГ-МИФИ, 2010.
2. *Шнайер Б.* Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. М.: ТРИУМФ, 2002.
3. *Алферов А. П., Зубов А. Ю., Кузьмин А. С., Черемушкин А. В.* Основы криптографии. М.: Гелиос АРВ, 2001.
4. *Брассар Ж.* Современная криптология. Перевод с английского. М.: Полимед, 1999.
5. *Saran N., Doganaksoy A.* Choosing Parameters to Achieve a Higher Success Rate for Hellman Time Memory Trade Off Attack // Int. Conference on Availability, Reliability and Security. 2009. P. 504–509.