А. Г. Ростовцев

О времени жизни общего и персонального открытого ключа

Криптосистемы с открытым ключом часто строятся на основе задачи дискретного логарифмирования в циклической группе вычислимого или трудновычислимого порядка (в группе обратимых элементов кольца, якобиане алгебраической кривой и т. п.). В общем случае эта задача формулируется так: в группе G решить показательное уравнение $a^x = b$.

Защищенные системы связи в этом случае имеют общий для всех участников открытый ключ, каждый пользователь имеет персональный секретный и парный ему открытый ключ. Общим ключом является способ задания группы G и образующая a, персональным открытым ключом является значение b, персональным секретным ключом является логарифм x. Например, в стандартах цифровой подписи РФ и США общим открытым ключом является характеристика поля p и образующая a группы простого порядка r.

Персональные ключи должны периодически меняться в соответствии с установленным порядком по аналогии с симметричными криптосистемами. При смене персонального ключа симметричные криптографические алгоритмы допускают сохранение общего ключа (например, блока подстановок в ГОСТ 28147–89). Что можно сказать о времени жизни общего открытого ключа в криптосистемах с открытым ключом? Можно ли продлить срок безопасной эксплуатации криптосистемы с открытым ключом, периодически меняя персональные ключи всех пользователей и сохраняя общий открытый ключ? Если смена персонального ключа решается в организационном плане несложно, то смена общего открытого ключа сопряжена со значительными неудобствами в эксплуатации. Кроме того, после выработки срока действия общего ключа все цифровые подписи, составленные с его помощью, должны считаться недействительными, а все сеансовые ключи — раскрытыми.

Будем рассматривать только криптографические ограничения на срок жизни ключа.

1. Мультипликативная группа кольца

Задача логарифмирования в группе $\mathbf{F}_p^*$ положена в основу отечественного стандарта подписи, наилучший метод раскрытия персонального секретного ключа — решето числового поля [1], обладающий субэкспоненциальной сложностью $S = O\left(\exp\left(c\sqrt[3]{\ln p \cdot (\ln\ln p)^2}\right)\right)$, где $c \approx 1{,}92$ при условии удачного выбора простого числа p. Аналогичную сложность имеет и метод логарифмирования на основе решета поля алгебраических функций.

«Проблемы информационной безопасности. Компьютерные системы»,
СПб., 1999. № 4. С. 57-61

Метод решета числового поля основан на выборе неприводимого над $\mathbf{Q}$ многочлена $f(X) \in \mathbf{Z}[X]$ небольшой степени n (практически $2 \le n \le 5$) и целого числа $m < p^{1/n}$ такого, что $f(m) \equiv 0 \pmod p$ и m имеет малые простые делители. Неудачный выбор числа p заключается в том, что существует разреженный многочлен $f(X)$ с малыми по абсолютной величине ненулевыми коэффициентами. В основу метода положена цепочка гомоморфизмов колец φ: $\mathbf{Z}[\alpha] \to \mathbf{Z} \to \mathbf{F}_p$, $\varphi(\alpha) \equiv m \pmod p$, и однозначность разложения идеалов в $\mathbf{Z}[\alpha]$ и в $\mathbf{Z}$ на простые множители.

Наиболее трудоемкий этап логарифмирования заключается в составлении базы данных из пар целых рациональных чисел (c, d) таких, что $c + dm$ и норма идеала $(c + d\alpha)$ раскладываются на множители из базы B, не превышающие y, и в решении системы линейных уравнений. При этом база, в которой раскладываются числа $c + dm$, может быть более узкой, чем база, в которой раскладываются нормы идеалов.

Модифицируем метод решета числового поля для случая, когда a и b неизвестны.

1. Создать базу данных размера не менее $\#B$ из пар (c, d).
2. Выразить базисные элементы в виде линейных комбинаций показателей, найденных на шаге 1.
3. Когда нарушитель узнает персональный ключ b, он подбирает произведение $a^k b^l$, имеющее малые простые делители.
4. Ненулевые координаты элемента $a^k b^l$ выражаются через векторы, найденные на шаге 2 (число ненулевых координат этого элемента пренебрежимо мало по сравнению с размером базы данных).
5. Решается система линейных уравнений для ненулевых координат вектора, найденного на шаге 4, и вычисляется логарифм.

Сложность шагов 4–6 составляет примерно $O(u^u)$ операций, где $u \approx \dfrac{\ln p}{\ln y}$. Традиционная оценка сложности раскрытия ключа равна $O(u^u y^3)$, сложность вычисления секретного ключа (при выполненных шагах 1–3) не превышает квадратного корня из сложности первого этапа. Например, если первоначально раскрытие персонального ключа требует нескольких лет вычислений, то в случае его смены при составленной базе данных срок безопасной эксплуатации продлится на несколько секунд. Смена образующей по сути ничего не меняет, так как в этом случае шаг 4 нужно выполнить два раза. Поэтому здесь, в отличие от симметричных криптосистем, по окончании срока действия любого персонального ключа нужно менять характеристику поля и, следовательно, ключи всех пользователей независимо от срока их ввода в действие.

В криптографии иногда используется задача логарифмирования в группе трудновычислимого порядка $(\mathbf{Z}/pq\mathbf{Z})^*$ обратимых элементов кольца вычетов по модулю составного числа pq с неизвестным разложением.

Данная задача сводится к разложению общего для всех составного числа. Очевидно, что и в этом случае выработка срока действия любого персонального ключа влечет замену характеристики кольца и, следовательно, замену всех персональных ключей. Попутно отменяются защитные качества всех криптографических алгоритмов, использующих отмененный общий открытый ключ, независимо от начала действия персональных ключей.

Данное свойство криптосистем в сочетании с быстрым падением стойкости — десятичный порядок в год для размера задачи 512 бит — демонстрирует неудобство популярных криптографических алгоритмов, допускающих субэкспоненциальные алгоритмы раскрытия. В частности, цифровая подпись электронных документов, действующая в течение нескольких лет, не может быть реализована средствами ГОСТ Р 34.10–94 при размере задачи менее 1–2 кбит. Кроме того, на сложность логарифмирования существенно влияет способ выбора характеристики поля p. В частности, "мина" может быть заложена, если сначала выбрать многочлен $f(X)$, числа α и m, а затем — простое число p с требуемыми свойствами. Аналогичная "мина" может быть заложена и в задачу разложения. Кроме того, составное число может быть не свободным от квадратов или содержать более двух простых делителей, что значительно снижает стойкость.

В последние годы предпринимаются попытки создать электронные деньги на основе криптографической подписи "вслепую", причем протоколы часто основываются на задаче разложения или логарифмирования в мультипликативной группе конечного поля. Ограничение на срок действия общего ключа ведет к тому, что все электронные монеты независимо от даты выпуска имеют одинаковый срок окончания действия, по истечении которого они не могут быть погашены или обменены, так как все по определению должны считаться фальшивыми. Социальные последствия использования таких "денег" можно предсказать — ситуация с памятным всем обменом купюр покажется невинной шалостью.

2. Общий случай циклической группы вычислимого порядка

Пусть абелева группа G имеет простой порядок r (в противном случае в соответствии с основной теоремой об абелевых группах задача сводится к логарифмированию в подгруппах простых порядков).

Для логарифмирования в группе G наилучшим является алгоритм Полларда [2]. Этот алгоритм использует сжимающее отображение $\varphi: G \to G$, сохраняющее вычислимость логарифма (если логарифм y известен, то логарифм $\varphi(y)$ можно легко найти). Алгоритм предусматривает построение цепочки отображений для начального элемента y, логарифм которого известен или может быть выражен через x. Граф конечен, поэтому существует натуральное число n такое, что выполняется равенство $\varphi^n(y) = \varphi^{2n}(y)$. Отсюда находится логарифм решением несложного уравнения по модулю

порядка группы. Алгоритм Полларда невозможно улучшить за счет увеличения объема памяти [4].

Рассмотрим модификацию алгоритма Полларда при известном общем ключе и неизвестном персональном ключе. Предположим, что память вычислительной модели ограничена значением $O(\sqrt{r})$, что обычно имеет место на практике. Выберем отображение, зависящее только от образующей a и текущего значения аргумента. Например, для одной половины аргументов $\varphi(y) = ay$ и $\log(\varphi(y)) = \log(y) + 1$, для другой половины аргументов $\varphi(y) = y^2$ и $\log(\varphi(y)) = 2\log(y)$. Алгоритм предусматривает выбор случайных начальных значений y_i, логарифмы которых известны, последовательное выполнение n отображений и запоминание троек $\{\varphi^n(y_i), \log(\varphi^n(y_i)), y_i\}$ с сортировкой по первой координате. После того, как нарушитель узнает b, он поочередно выбирает элементы, логарифмы которых являются линейной функцией от b, делает m отображений, сравнивая на каждом шаге результат с базой данных. При совпадении приравниваются логарифмы и находится x.

Граф случайного отображения φ представляет собой ориентированный лес, корни которого связаны в циклы, при этом почти все вершины лежат на одном дереве [3]. Случайное дерево при обращении ребер моделируется пуассоновским ветвящимся процессом с параметром $\lambda = 1$. Задача логарифмирования сводится к задаче о встрече на случайном ориентированном дереве.

Определим индуктивно глубину d на дереве с r вершинами. Все листья имеют глубину 0, вершина имеет глубину d, если максимальное расстояние от какого-либо из листьев до вершины равно d. Максимальная глубина равна $O(\sqrt{r})$. Обозначим $P(d)$ вероятность того, что вершина имеет глубину не более d, $p(d)$ — вероятность того, что вершина имеет глубину d. Тогда $p(0) = 1/e$ (e — основание натуральных логарифмов). Имеет место рекуррентное соотношение $P(d) = -1 + e^{-1+P(d-1)}$. После каждого шага от вершины с глубиной d глубина может увеличиваться от $d + 1$ до $O(\sqrt{r})$, вероятность попадания на каждую глубину пропорциональна доле вершин с данной глубиной. Вероятности $P(d)$ и $p(d)$ при больших значениях d равны соответственно $(1 - 2/d)$ и $2/d^2$.

Вероятность того, что после $k > 0$ шагов по графу взятая наугад вершина будет совпадать с вершиной из базы данных, равна

$$p_k = \sum_{d=1}^{O(\sqrt{r})} p_k(d)\,p(d),$$

где $p_k(d)$ — вероятность попасть после шага k на глубину d.

Попасть на глубину d после k шагов можно при следующих чередованиях глубин: $(0, 1, ..., k - 3, k - 2, d)$, $(0, 1, ..., k - 3, k - 1, d)$, ..., $(0, 1, ..., k - 3, d - 1, d)$, ..., $(d - k, ..., d - 1, d)$. Число вариантов чередования глубин

равно $\binom{d}{k}$. Общее число путей длины k на глубину d равно k-й элементарной симметрической функции от d аргументов. Эта симметрическая функция выражается в виде многочлена $H(f_1, \ldots, f_k)$ над $\mathbf{Q}$ от симметрических функций $f_i = \sum p(d)^i$ для $1 \le i \le k$, которые могут быть оценены интегралами. Длина многочлена H определяется числом разбиений числа k и асимптотически равна субэкспоненте $\frac{e^{\pi\sqrt{2k/3}}}{4k\sqrt{3}}$ [5]. Все коэффициенты многочлена H имеют вид $1/s$, где $s \mid (k!)$, их сумма равна нулю.

Для того, чтобы суммарная вероятность равнялась единице, введем нормирование:

$$p_k(d) \approx \frac{p_k'(d)}{\sum_{j=1}^{O(\sqrt{q})} p_k'(j)},$$

где p_k' — аппроксимированное значение вероятности.

Анализ сложности алгоритма с использованием численных методов и указанных аппроксимаций показал, что оптимальными являются значения $m = O(n) = O(\log r)$ или $n = O(1)$, $m = O(\sqrt{r})$. Если доступная память не превышает $O(\sqrt{r})$, то алгоритм имеет ту же асимптотическую сложность, что и обычный алгоритм Полларда.

Таким образом, в криптосистемах с открытым ключом, в которых наилучшим методом раскрытия ключа является алгоритм Полларда, замена персонального ключа при сохранении общего ключа позволяет продлевать срок безопасной эксплуатации. В этом проявляется аналогия с симметричными криптосистемами. К такому классу на сегодняшний день относятся криптосистемы на алгебраических кривых невысокого рода, в частности, на эллиптических кривых при условии, что группа точек не может быть вложена в аддитивную или мультипликативную группу поля при небольших степенях расширения.

Литература

1. D. Gordon. Discrete logarithms in *GF*(*p*) using number field sieve // SIAM Journal on Discrete Mathematics, v. 6, № 1, 1993, pp. 124–138.
2. J. Pollard. Monte Carlo methods for index computation (mod *p*) // Mathematics of Computation, v. 32, № 143, pp. 918–924.
3. В. Ф. Колчин. Случайные отображения. — М.: Наука, 1984.
4. А. Г. Ростовцев, А. П. Буренкова. О невозможности улучшения алгоритма Полларда для шифров, образующих группу // Тезисы докладов конференции “Региональная информатика” РИ–96, С.-Петербург, 1996. С. 121.
5. Г. Эндрюс. Теория разбиений. — М.: Наука, 1982.