достижимые характеристики эффективности случайного кодирования как способа криптографической защиты информации.

В настоящей статье предложена конструкция двоичных линейных кодов, построенных на основе кодов РС над полем **GF**(2^m), обеспечивающих по основным показателям (стойкости защиты информации в отводном канале, скорости передачи, длине кодовых слов) существенно более высокую эффективность случайного кодирования по сравнению с эффективностью систем, построенных на основе кодов Хэмминга и других линейных кодов с большим дуальным расстоянием. Показано также, что при относительно плохом качестве одводного канала ($p \geq 0{,}1$) применение предложенных кодов в системах со случайным кодированием информации обеспечивает меньшую вероятность правильного декодирования в отводе при большей скорости передачи по сравнению с некоторыми другими ранее исследованными кодами.

Разработанные и описанные в статье алгоритмы случайного кодирования и декодирования сообщений с использованием предложенных кодов основываются на дискретном преобразовании Фурье в поле **GF**(2^m), позволяют осуществлять случайное кодирование (декодирование в основном канале) с временной сложностью (измеряемой количеством операций сложения и умножения в поле **GF**(2^m)), пропорциональной длине кодовых слов, и могут быть практически эффективно реализованы в перспективных системах кодовой защиты информации.

Литература: ***1.*** *Wyner A. D. The Wire-Tap Channel // Bell System Techn. J. – 1975. – V. 54. – № 8. – P. 1355–1388.* ***2.*** *Csiszar I., Korner J. Broadcast Channels with Confidential Messages // IEEE Trans. Inform. Theory. – 1978. – V. 24. – № 3. – P. 339–348.* ***3.*** *Коржик В. И., Яковлев В. А. Неасимптотические оценки кодового зашумления одного канала // Проблемы передачи информации. – 1981.– Т.17.– В.4.– С.11-18.* ***4.*** *Maurer U. M. Provable Security in Cryptography: Diss. ETH № 9260. – 1990. – 120 p.* ***5.*** *Коржик В. И., Яковлев В. А. Пропускная способность канала связи с внутренним случайным кодированием // Проблемы передачи информации. – 1992. – Т. 28. – В. 4. – С. 24-34.* ***6.*** *Горицкий В. М. Вероятностная криптография в системах защиты информации: кодовая защита // Электроника и связь. – 1998. – В. 5. – С. 140-145.* ***7.*** *Иванов В. А. О методе случайного кодирования // Дискретная математика. – 1999. – Т. 11. – В. 3. – С. 99–108.* ***8.*** *Чисар И. Почти независимость случайных величин и пропускная способность криптостойкого канала // Проблемы передачи информации. – 1996. – Т. 32. – В. 1. – С. 48–57.* ***9.*** *Алексейчук А. Н. О вероятности безошибочного декодирования в отводном канале с аддитивным шумом, распределенным на конечной абелевой группе // Защита информации: сборник научных трудов Национального авиационного ун-та. – К.: КМУГА, 2001. – С. 9–16.* ***10.*** *Мак-Вильямс Ф. Дж., Слоэн Н. Дж. А. Теория кодов, исправляющих ошибки: Пер. с англ. – М.: Связь, 1979. – 743 с.* ***11.*** *Алексейчук А. Н. Оценки эффективности кодовой защиты дискретных сообщений с использованием линейных кодов с большим дуальным расстоянием // Реєстрація, зберігання і обробка даних. – 2001. – Т. 3 – № 2. – С. 99–106.* ***12.*** *Блейхут Р. Быстрые алгоритмы цифровой обработки сигналов: Пер. с англ. – М.: Мир, 1989. – 448 с.* ***13.*** *Ноден П., Китте К. Алгебраическая алгоритмика: Пер. с франц. – М.: Мир, 1999. – 719 с.*

УДК 681.3.06:519.248.681

ТЕСТИРОВАНИЕ ЧИСЕЛ НА ПРОСТОТУ: ТЕОРИЯ И ПРАКТИКА

Иван Горбенко, Виталий Вервейко
Харьковский национальный университет радиоэлектроники

Анотація: **Наводиться класифікація та огляд основних алгоритмів тестування чисел на простоту, а також їх порівняльний аналіз та рекомендації з побудови практичних засобів.**
Summary: **Classification, review of main primality test algorithms, comparative analysis and recommendation of mean building are given in the article.**
Ключові слова: **Прості числа, тестування чисел на простоту, асиметричні криптосистеми.**

Введение

Задача определения, является ли заданное число простым или составным, есть одной из фундаментальных проблемных задач теории чисел. Поиском решения данной задачи математики занимались много веков, однако до появления криптографии с открытым ключом [1] эффективных алгоритмов проверки чисел на простоту найдено не было.

Большинство криптосистем, базирующихся на асимметричной криптографии, используют в качестве одного или нескольких параметров простые числа. К ним относятся криптографические системы, использующие преобразования в кольцах (RSA, Рабина), полях Галуа (Эль-Гамаля), группах точек эллиптических кривых и др. Стойкость этих криптосистем зависит от правильности построения простых чисел различной длины (от 128 до 2048 и более бит). В течение последних 25 лет были разработаны ряд эффективных вероятностных алгоритмов для решения этой задачи и, наконец, в 2002 году предложен детерминированный алгоритм полиномиальной сложности для проверки чисел на простоту.

Настоящая статья содержит классификацию и обзор основных алгоритмов тестирования чисел на простоту, а также их сравнительный анализ и рекомендации по построению практических средств проверки целых чисел на простоту.

I Классификация алгоритмов тестирования чисел на простоту

Простым числом называется число p, которое не имеет нетривиальных делителей, т. е. делится только на 1 и на само себя. Задача тестирования чисел на простоту числа формулируется следующим образом. Пусть задано целое число $GF(p)$, необходимо определить, является ли n простым числом. Алгоритмы, которые предназначены для решения этой задачи, называют тестами на простоту. Если алгоритм принимает вход n, то говорят, что n проходит или выдерживает тест на простоту.

Как показал проведенный анализ, все алгоритмы тестирования на простоту можно разделить на три больших класса:

- вероятностные алгоритмы с односторонней ошибкой;
- алгоритмы с вероятностным временем выполнения;
- детерминированные алгоритмы.

Если n проходит тест вероятностным алгоритмом с односторонней ошибкой, то n является простым не достоверно, а только с некоторой вероятностью $\delta < 1$. В случае алгоритма с вероятностным временем выполнения, если число n проходит тест, то оно достоверно является простым, однако число шагов алгоритма, а, следовательно, и время выполнения алгоритма, зависят от некоторого случайного числа. Детерминированный алгоритм распознает простоту числа n с вероятностью, равной 1 и с заранее известным числом шагов.

Дополнительно алгоритмы тестирования на простоту можно классифицировать по сложности выполнения:

- экспоненциальные – алгоритмы с оценкой сложности порядка $O(c^{\log n})$ для некоторой константы $c > 1$;
- субэкспоненциальные – $O(c^{(\log n)^{\nu}(\log\log n)^{1-\nu}})$ для $c > 1$ и $0 < \nu < 1$;
- квазиполиномиальные – $O((\log n)^{c\log\log n})$ для $c > 0$;
- полиномиальные – $O(\log^c n)$ для $c \geq 1$.

Экспоненциальные и субэкспоненциальные алгоритмы являются алгоритмами факторизации и в случае, если число n – составное, находят его нетривиальный делитель. Однако эти алгоритмы исключены из рассмотрения, т. к. они предназначены для решения более сложной задачи и обладают существенно большей вычислительной сложностью, чем тесты на простоту.

II Вероятностные алгоритмы с односторонней ошибкой

Исторически первыми алгоритмами с полиномиальной сложностью выполнения стали вероятностные алгоритмы с односторонней ошибкой.

Одним из первых алгоритмов этого класса стал тест Лемана. Этот тест использует свойства малой теоремы Ферма, которая утверждает, что если n – простое число, то для всех $0 < x < n$ выполняется условие:

$$x^{n-1} \equiv 1 (\mathrm{mod}\, n). \quad (1)$$

Суть теста заключается в следующем: если для случайного $0 < x < n$ выполняется условие $x^{\frac{n-1}{2}} \equiv \pm 1 (\mathrm{mod}\, n)$, то число n является простым с вероятностью 2^{-1}. Числа, которые удовлетворяют условию (1), но не являются простыми, называют псевдопростыми числами Ферма по основанию x. При повторении теста k раз, вероятность ошибки уменьшается до 2^{-k}. Однако некоторые составные числа

являются псевдопростыми по любому основанию. Это числа Кармайкла, наименьшим из которых является $561 = 3 \cdot 11 \cdot 17$. В [2] показано, что существует бесконечное множество таких чисел. Поэтому этот тест на практике не используют.

Следующим алгоритмом этого класса является тест Соловея-Штрассена [3]. Данный тест является усилением теста Лемана и использует следующее утверждение: число n является простым, если для случайного $0 < x < n$ выполняется

$$НОД(x,n) = 1 \text{ и } \left(\frac{x}{n}\right) \equiv x^{\frac{n-1}{2}} (\mathrm{mod}\, n). \quad (2)$$

Вероятность ошибки для этого теста также составляет не более 2^{-1}, а при повторении теста k раз, вероятность ошибки уменьшается до 2^{-k}. Нечетные n, удовлетворяющие условию теста (2) и не являющиеся простыми, называют псевдопростыми числами Эйлера по основанию x. Сложность теста составляет $O(\log^3 n)$.

Еще более сильным тестом является тест Рабина-Миллера. Этот метод тестирования предложил Миллер в работе [4], а Рабин на его основе разработал вероятностный алгоритм тестирования [5]. Суть этого теста заключается в следующем. Пусть тестируемое число n представлено в виде $n-1 = 2^s t$, где t – нечетное. Тогда для случайного элемента $x \in \mathbf{Z}_n$ ($\mathbf{Z}_n$ – область целых чисел) рассчитывается последовательность $x^t, x^{2t}, x^{4t}, ..., x^{2^s t}$. С учетом (1) последний член этой последовательности равен 1, а каждый предыдущий является квадратным корнем из последующего. Так как для простого n $\mathbf{Z}_n$ является полем, то корнем из 1 может быть только 1 или -1. Т. е. либо вся последовательность состоит из единиц, либо существует -1, после которой следуют единицы. Это условие равносильно следующему:

$$\text{или } x^t \equiv 1 (\mathrm{mod}\, n), \text{ или существует } 0 \le j < s \text{ такое, что } x^{2^j t} \equiv -1 (\mathrm{mod}\, n). \quad (3)$$

Вероятность ошибки теста при одном значении x не превышает 2^{-2} и при повторении k раз для различных значений x она уменьшается до 2^{-2k}. Числа, которые удовлетворяют условию (3), но не являются простыми называют сильно псевдопростыми числами по основанию x. Сложность этого теста составляет $O(\log^3 n)$.

Следует отметить, что при условии справедливости расширенной гипотезы Римана (РГР), Миллер доказал [4], что любое нечетное составное число n не является сильно псевдопростым, по крайней мере, для одного основания $x < 2\ln^2 n$. Таким образом, при условии справедливости РГР на основании теста Рабина-Миллера может быть построен детерминированный полиномиальный алгоритм, причем его сложность определяется как $O(\log^5 n)$.

Заметим, что все приведенные выше тесты являются последовательными формальными усилениями один другого. При этом самым слабым тестом является тест Лемана, а самым сильным – тест Рабина-Миллера. Другими словами, все числа, являющиеся сильно псевдопростыми по основанию x, являются псевдопростыми числами Эйлера по основанию x, которые в свою очередь являются псевдопростыми числами Ферма по тому же основанию. Обратное утверждение не верно.

Самым новым алгоритмом этого класса является QFT – Quadratic Frobenius Test [6]. В отличие от рассмотренных выше алгоритмов, которые для определения простоты n проверяют структуру Абелевой группы $\mathbf{Z}_n^*$, QFT рассматривает ее квадратичное расширение, т. е. рассматривается кольцо $\mathbf{Z}_n[x]/(f(x))$, где $f(x)$ – полином второй степени вида $f(x) = x^2 - bx - c$. Коэффициенты полинома выбираются таким образом, чтобы полином был неприводимым, когда n – простое. В этом случае кольцо является конечным полем порядка n^2. В основе теста лежат следующие три условия:

$$\begin{aligned} & x^{\frac{n+1}{2}} (\mathrm{mod}(x^2 - bx - c), n) \in \mathbf{Z}_n; \\ & x^{\frac{n+1}{2}} \equiv -c (\mathrm{mod}(x^2 - bx - c), n); \\ & \text{и } x^t \equiv 1 (\mathrm{mod}(x^2 - bx - c), n), \end{aligned} \quad (4)$$

или $x^{2^j t} \equiv -1(\mathrm{mod}(x^2 - bx - c), n);$ для некоторого $0 \le j \le s-2$,

где $n^2 - 1 = 2^s t$. Составные числа, которые удовлетворяют условию (4), называются псевдопростыми Фробениуса. Вероятность ошибки для этого теста не превышает 7710^{-1} и при повторении теста k раз для различных значений b и c уменьшается до 7710^{-k}. В [7] разработана более сильная версия теста для чисел вида $n \equiv 1(\mathrm{mod}\,4)$, вероятность ошибки, которой на одном цикле не более $131040^{-1} \approx 2^{-17}$. Сложность теста составляет $O(\log^3 n)$.

III Алгоритмы с вероятностным временем выполнения

Следующим важным этапом в развитии тестов на простоту стало появление алгоритмов с вероятностным полиномиальным временем выполнения. Основными алгоритмами этого класса являются тесты с использованием эллиптических кривых. В 1985 году Ленстра разработал алгоритм факторизации с использованием эллиптических кривых. После этого был предпринят ряд попыток построить аналогичный, но более простой алгоритм для тестирования на простоту. Первым это удалось сделать Голдвассеру и Килиану [8]. В этом алгоритме для расчета порядка эллиптической кривой использовался алгоритм Скуфа. Он выполнялся за случайное полиномиальное время, но при условии выполнения некоторых правдоподобных, но не доказанных утверждений из аналитической теории чисел. Другой алгоритм был предложен в работе [9] и использовал гиперэллиптические кривые второго рода. Сложность этого алгоритма также является полиномиальной, но уже без использования каких либо недоказанных гипотез. Оба эти алгоритма хотя и имели полиномиальную сложность, но слишком высокую и поэтому практического использования не нашли.

Следующим тестом этого класса стал алгоритм, предложенный Аткином и реализованный Морейном [10] и известный как ECPP – Elliptic Curve Primality Proving. Этот тест подобен тесту Голдвассера-Килиана, однако имеет несколько отличий, главным из которых является применение эллиптических кривых с более простым методом вычисления порядка – эллиптических кривых с комплексным умножением. В 2002 году была реализована еще более эффективная версия теста с использованием для вычисления порядка кривой алгоритма SEA [11]. В основе алгоритма лежит следующее утверждение [8]: если существует несингулярная эллиптическая кривая $y^2 = x^3 + Ax + B(\mathrm{mod}\, n)$ порядка $u = s \cdot r$, где r – простое, $r > (\sqrt[4]{n} + 1)^2$, $s \ge 2$ и точка $P = (x, y)$, принадлежащая этой кривой, такая, что $uP = 0$, а $sP = 0$, то n – простое. Исходя из этого утверждения, доказательство простоты n можно свести к доказательству простоты r, причем $r < n$. Применяя выше рассмотренное утверждение k раз доказательство простоты n можно свести к доказательству простоты небольшого числа r_k , которое может быть выполнено некоторым детерминированным алгоритмом, например, пробным делением. Из теоремы Хасэ о порядке эллиптической кривой следует, что количество итераций в худшем случае $k < \log_2 n$ (при $s = 2$), однако число попыток построения подходящей эллиптической кривой на каждой итерации является случайным. Следует сказать, что в процессе доказательства, элементы последовательности r_i, $i = \overline{0,k}$, являются вероятно простыми, поэтому они проверяются при помощи какого либо алгоритма с односторонней ошибкой. Все элементы последовательности являются доказанными простыми числами как только доказывается простота r_k.

Одним из достоинств этого алгоритма является возможность создания сертификата простоты – некоторой последовательности промежуточных вычислений алгоритма, которая может быть проверена за детерминированное полиномиальное время, значительно меньшее, чем требуется для доказательства. Сложность алгоритма доказательльства оценивается [11] как $O((\log n)^{6+\varepsilon})$, а сложность алгоритма проверки – как $O(\log^4 n)$.

IV Детерминированные алгоритмы

Детерминированные алгоритмы тестирования на простоту известны уже более 2-х тысячелетий. Первым из известных нам алгоритмов является решето Эратосфена, который определяет простоту n по следующему критерию: число n является простым тогда и только тогда, когда оно не делится ни на одно из предыдущих ему простых чисел. На самом деле достаточно проверить только делимость на все простые $p_i \le \left\lfloor \sqrt{n} \right\rfloor$. Такую

проверку называют алгоритмом пробного деления. Следующий алгоритм с оценкой времени $O(\sqrt[3]{n})$ был предложен в 1974 году [12]. Однако сложность всех этих алгоритмов является экспоненциальной, и они не могут быть применены для проверки простоты чисел, используемых в криптографических преобразованиях.

Одним из лучших детерминированных алгоритмов тестирования на простоту на сегодня является APR [13]. Алгоритм APR использует теорию циклических расширений кольца $\mathbf{Z}_n$ порядка s и использует сложный математический аппарат. В работе [14] были предложены модификация алгоритма, известная как APR-CL, в которой были упрощены вычисления, используемые в алгоритме. Сложность алгоритма составляет $O((\log n)^{c\log\log\log n})$. Существует также версия этого алгоритма с вероятностным временем выполнения, которая обладает сравнимой сложностью, однако более эффективно реализуется на ЭВМ.

Важнейшим теоретическим достижением в области тестирования на простоту стала разработка детерминированного алгоритма полиномиальной сложности. Таким алгоритмом стал AKS [15]. В основании алгоритма лежит утверждение, что если выполняется

$$(x-a)^n \equiv x^n - a(\mathrm{mod}\, x^r - 1, n), \tag{5}$$

для некоторого небольшого числа r и всех $1 \le a \le s$, то n – некоторая степень простого числа. Число r выбирается из следующего условия: пусть q – наибольший простой делитель $r-1$, тогда r считается подходящим, если r является простым числом, $n^{(r-1)/q} \bmod r \notin \{0,1\}$ и $\binom{q+s-1}{s} \ge n^{2\lfloor\sqrt{r}\rfloor}$. В работе [16] была уменьшена вычислительная сложность алгоритма путем уменьшения требований к r: r считается подходящим, если n является первообразным элементом по модулю r, $n^{(r-1)/q} \bmod r \notin \{0,1\}$ и $\binom{\varphi(r)+s-1}{s} \ge n^{\lfloor\sqrt{r}\rfloor}$. Временная сложность алгоритма оценивается как $O(\log^6 n)$.

Существует ряд тестов, основанных на попытке разложения чисел $n-1$ и $n+1$. Этот тест применим только для чисел специального вида, таких, у которых в разложении $n-1$ или $n+1$ есть большой простой делитель $R > \sqrt{n}$. Суть теста $n-1$ заключается в следующем: пусть $n-1 = R \cdot S$, $R > S$ и R – простое, если существует небольшое целое $a > 1$ такое, что

$$a^{n-1} \equiv 1(\mathrm{mod}\, n) \text{ и } НОД(a^S - 1, n) = 1, \tag{6}$$

то n – простое. Тест $n+1$ несколько сложнее: пусть $n+1 = R \cdot S$, $R > S$ и R – простое, если существует последовательность Лукаса $V[i]$ с параметрами P и Q, такими, что

$$НОД(P,Q) = 1,\ \left(\frac{P^2 - 4Q}{n}\right) = -1,$$
$$V[(n+1)/2] \equiv 0(\mathrm{mod}\, n) \text{ и} \tag{7}$$
$$НОД\left(\frac{S}{2}, n\right) = 1,$$

то n – простое. Эти тесты практически не применяются в чистом виде, однако они могут быть использованы для эффективного построения простых чисел. Также они могут применяться совместно с тестом ECPP для повышения его эффективности: доказательство простоты числа n, для которого удалось найти разложение $n-1$ или $n+1$, удовлетворяющее условиям (6) или (7) соответственно, сводится к доказательству простоты числа R без сложных вычислений для построения эллиптической кривой. Существует и другой вид теста $n-1$. Этот вид применим, если удалось факторизовать часть $F > \sqrt{n}$ числа $n-1$. Если существует небольшое целое $a > 1$ такое, что

$$a^{n-1} \equiv 1(\mathrm{mod}\, n) \text{ и } НОД(a^{\frac{n-1}{q}} - 1, n) = 1 \tag{8}$$

для всех $q \mid F$, то n – простое. Этот тест эффективно применим к другим числам n – числам, разложение $n-1$ которых имеет много небольших простых делителей. Последние исследования в этой области показывают, что для работы полиномиального теста на простоту достаточно факторизовать часть $F > n^{3/10}$.

Задача построения простых чисел в общем случае сводится к задаче тестирования на простоту. Однако, в некоторых случаях она может быть решена намного проще, хотя при этом сужается (иногда значительно) множество простых чисел, которые могут быть построены таким образом (по сравнению с генерацией случайного числа с последующим тестированием его на простоту). Существует ряд алгоритмов, позволяющих эффективно строить простые числа с заданными свойствами, которые основаны на свойствах рассмотренных выше алгоритмов и некоторых других, например, арифметических прогрессиях [17].

V Практика тестирования на простоту

Как было показано выше, теоретически существует детерминированный алгоритм с полиномиальной сложностью – AKS, определяемый (5). Поэтому для решения задачи тестирования на простоту достаточно эффективно реализовать этот алгоритм. Однако при его реализации появляется ряд серьезных трудностей, одной из которых является быстрое возрастание порядка полинома r. В табл. 1 показана зависимость порядка полинома от длины тестируемого числа n.

Таблица 1 - зависимость порядка полинома от длины тестируемого числа

Длина n, бит	32	64	128	256	512	1024
AKS	>65000	>262000	>1048000	>4194000	>16777000	>67108000
AKS-2	>4000	>16000	>65000	>262000	>1047000	>4192000

Анализ показывает, что вместе с возрастанием степени полинома r возрастает и требуемое количество циклов проверки s. Даже при применении умножения с использованием быстрого преобразования Фурье, тестирование 32-х битового числа алгоритмом AKS-2 занимает более 1,5 часов на ЭВМ с процессором Celeron 800, что значительно дольше, чем пробное деление. Таким образом, алгоритм AKS является значительным теоретическим достижением, однако на практике он не применим.

В работе [15] приводится гипотеза, которая позволяет существенно упростить этот тест. Гипотеза утверждает, что число n является простым тогда и только тогда, когда выполняется условие

$$(x-1)^n \equiv x^n - 1 (\mathrm{mod}\, x^r - 1, n), \tag{9}$$

для любого простого r, такого, что $n^2 \bmod r \neq 1$. Эта гипотеза не является доказанной, но в работе [18] проверена для всех чисел меньше 10^{10}, т. е. может использоваться для быстрой проверки на простоту небольших чисел без пробного деления. Авторы статьи провели дополнительные эксперименты с числами больших длин. В ходе эксперимента была произведена проверка 10^7 64-х битовых и 10^6 128-битовых чисел, сформированных случайным образом. В результате эксперимента все простые и составные числа были распознаны правильно.

Основными тестами, используемыми в настоящее время, являются тесты с односторонней ошибкой. Наиболее широкое применение получил тест Рабина-Миллера, поэтому интересно именно его сравнить с новым тестом QFT. Проведенные исследования показывают, что программная реализация алгоритма QFT примерно в 3,5 раза медленнее, чем реализация алгоритма Рабина-Миллера, однако один цикл QFT дает вероятность ошибки, сравнимую с 6-ю циклами алгоритма Рабина-Миллера. Дополнительно следует отметить, что в работе [19] исследована зависимость вероятности ошибки теста Рабина-Миллера от длины тестируемого числа и показано, что для 1024 битового нечетного числа для достижения вероятности ошибки 2^{-80} достаточно всего трех циклов. Возможно, аналогичная зависимость существует и для алгоритма QFT, но такие исследования только начались. Также для теста Рабина-Миллера нет необходимости реализовывать значительно более сложную математику работы с полиномами. Таким образом, в настоящее время наиболее целесообразным является использование теста Рабина-Миллера, однако необходимо учитывать некоторые новые сведения.

Многие криптографические системы в настоящее время используют тест Рабина-Миллера по двадцати-сорока фиксированным основаниям. Существует методика построения сильно псевдопростых чисел по заданным основаниям. В качестве примера такого числа можно привести следующее:

80383745745363949125707961434194210813883768828755814583748891752229\
74273765333652186502336163960045457915042023603208766569966760987284\

043965408232928738791850869166857328267761771029389697739470167082 30\
428687109997439976544144845341155872450633409279022275296229414984 23\
068816854043264575340183297861112989606448452161916528725975349 01

Это число является сильно псевдопростым по 46 основаниям, но не является простым. Кроме того, это число не имеет малых простых делителей (<10000000), т. е. обычно предшествующий тесту Рабина-Миллера тест пробного деления также ничего не выявит. Это позволяет центру генерации общесистемных параметров криптографических систем с открытым ключом произвести фальсификацию простых чисел, которая не будет выявлена в клиентских машинах в случае детерминированного теста. Использование же чисел, которые не являются простыми, значительно ослабит стойкость большинства криптосистем. Таким образом, можно сформулировать главное требование к реализации теста Рабина-Миллера: основания теста должны выбираться случайно, равновероятно и независимо.

Чтобы полностью избавиться от проблем с вероятностями и возможностью обмана следует применять тесты с достоверной проверкой простоты. В настоящее время существует два эффективных теста такого класса: APR-CL и ECPP. Оба метода используют сложную математическую базу и достаточно сложны в реализации. Для сравнения с APR-CL была реализована комбинированная версия теста ECPP с тестами $n-1$ и $n+1$. Результаты тестирования этих алгоритмов по времени выполнения для чисел различной длины приведены в табл. 2. Тестирование производилось на ЭВМ с процессором Celeron 800 MHz, 128 MB RAM.

Таблица 2

Длина n, бит	128	192	256	512	1024	2048
APR-CL	0,32 с	0,98 с	1,65 с	24 с	335 с	6774 с
ECPP	0,06 с	0,10 с	0,20 с	1,9 с	31 с	637 с

Как видно из таблицы, алгоритм APR-CL эффективно применим для тестирования на простоту чисел меньше 2^{1024}. Тест ECPP является более эффективным, но главным его преимуществом является то, что он формирует сертификат простоты, который может быть значительно быстрее и проще проверен, чем сформирован. Для больших чисел проверка может производиться более чем в 10 раз быстрее, чем формирование. Кроме того, в случае использования этого теста в системе формирования общесистемных параметров для криптосистем, использующих преобразования в группах точек эллиптических кривых, реализация этого теста значительно упрощается, т. к. наиболее сложной его частью является именно построение эллиптической кривой. Использование же в таких системах теста APR-CL вряд ли целесообразно – для этого придется реализовывать другую сложную математическую базу и совершенно другие принципы проверки. В случае необходимости повторной проверки на простоту тестом APR-CL необходимо заново выполнять тест в полном объеме.

Заключение

К настоящему времени разработан ряд эффективных тестов проверки “больших” чисел на простоту, позволяющих строить простые числа для асимметричных криптографических алгоритмов. Как показал проведенный анализ, практически ни один из алгоритмов не является эффективным при самостоятельном использовании. Алгоритмы достоверного тестирования слишком сложны для проверки большого объема чисел, которые нужно протестировать при генерировании случайного простого числа, и могут использоваться только для подтверждения простоты вероятно простого числа. Вероятностные же алгоритмы с односторонней ошибкой тестирования могут пропускать некоторые составные числа.

Таким образом, в качестве эффективного алгоритма построения достоверно простых чисел для использования в криптографических приложениях, требующих повышенной стойкости, можно предложить следующий алгоритм.

1. Формирование случайного нечетного числа n.
2. Проверка n методом пробного деления на простые числа, меньшие 65535. Если проверка не проходит, перейти к шагу 1.
3. Проверка десятикратным тестом Рабина-Миллера (допускается использование фиксированных оснований, т. к. на следующем шаге будет производиться доказательство простоты). Если проверка не проходит, перейти к шагу 1.
4. Формирование сертификата простоты комбинированным алгоритмом ECPP, $n-1$ и $n+1$. Если сертификация не проходит, перейти к шагу 1.

На шаге 1 не рекомендуется вместо формирования нового случайного числа применять увеличение предыдущего на 2, так как это приведет к различным вероятностям появления разных простых чисел. Эта

вероятность зависит от количества нечетных составных чисел, находящихся пред простым числом, и может отличаться в сотни раз для разных простых чисел.

На выходе вышеприведенного алгоритма будет получено гарантированно простое число с соответствующим ему сертификатом, который может быть проверен с небольшими вычислительными затратами. Многие криптосистемы требуют построения сильных простых чисел. Алгоритмы построения таких чисел также могут быть модифицированы с учетом вышеприведенного алгоритма.

Вместе с тем как перспективные необходимо рассматривать алгоритмы, определяемые (5) и (9) соответственно. Основными проблемными задачами для них являются теоретическое обоснование алгоритма (9) и уменьшение сложности (5).

Литература: ***1.*** *R. L. Rivest, A. Shamir, and L M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. Comm. ACM, 1978.* ***2.*** *Alford W. R., A. Granville and C. Pomerance. There are infinitely many Carmichael numbers. Ann. Math., 140, 703-722, 1994.* ***3.*** *R. Solovay and V. Strassen. A fast Monte-Carlo test for primality. SIAM Journal on Computing, 6:84–86, 1977.* ***4.*** *G. L. Miller. Riemann's hypothesis and tests for primality. J. Comput. Sys. Sci., 13:300–317, 1976.* ***5.*** *M. O. Rabin. Probabilistic algorithm for testing primality. J. Number Theory, 12:128-138, 1980.* ***6.*** *J. Grantham. A Probable prime test with high confidence. J. Number Theory 72, pp. 32–47, 1998.* ***7.*** *S. Müller. A probable prime test with very high confidence for n ≡ 1 mod 4. Proceeedings of Asiacrypt 2001, Springer Verlag LNCS.* ***8.*** *S. Goldwasser and J. Kilian. Almost all primes can be quickly certified. In Proceedings of Annual IEEE Symposium on Foundations of Computer Science, pages 316–329, 1986.* ***9.*** *L. M. Adleman and M. A. Huang. Recognizing primes in random polynomial time. In Proceedings 19th STOC (1986), pp. 462–469. New-York City, 1987.* ***10.*** *A. O . L. Atkin and F. Morain. Elliptic curves and primality proving. Math. Comp., 61:29–68, 1993.* ***11.*** *F. Morain. Computing the cardinality of CM elliptic curves using torsion points. 2002.* ***12.*** *R. S. Lehman. Factoring large integers. Math. Comp., 28:637–646, 1974.* ***13.*** *L M. Adleman, C. Pomerance and R. S. Rumely. On distinguishing prime numbers from composite numbers. Ann. Math., 117:173–206, 1983.* ***14.*** *H. Cohen and H. W. Lenstra. Primality testing using Jacobi sums. Math. Comp., 42:297–330, 1984.* ***15.*** *M. Agrawal, N. Kayal and N. Saxena. PRIMES is in P. Preprint; available from http://www.cse.iitk.ac.in/primality.pdf, 2002.* ***16.*** *D. J. Bernstein. An exposition of the Agrawal-Kayal-Saxena primality-proving theorem. Preprint; available from http://cr.yp.to/papers.html#aks, 2002.* ***17.*** *P. Mihailescu. Fast Generation of Provable Primes using Search in Arithmetic Progressions, Proceedings CRYPTO 94, LNCS, vol. 939, Springer 1994, pp. 282–293.* ***18.*** *R. Bhattacharjee and P. Pandey. Primality testing. Technical report. IIT Kanpur, 2001.* ***19.*** *I. B. Damgård, P. Landrock and C. Pomerance. Average case error estimates for strong prime test. Math. Of Comp., 61:177–194, 1993.*