Лекция 8

Криптосистеми основани на задачата за разлагане на прости множители

8.1 Криптосистема на **Cocks-Ellis**

През 1969 Джеймз Елис от британските Government Communication Haedquarters (GCHQ) предлага идеята за несекретно шифриране (non-secret encryption) или това което днес наричаме асиметрична криптография и по-специално идеята за еднопосочна функция. Това е функция, чиято обратна може да бъде намерена лесно от всеки, които притежава някаква допълнителна информация. Същата идея е предложена няколко години по късно от Дифи и Хелман, но подобно на Елис, и те първоначално не намират подходяща реализация. Три години по-късно. през ноември 1973, Джеимз Кокс предлага просто решение, по същество преоткрито няколко години по-късно от Ривест, Шамир и Ейделман [44].

Сега ще опишем системата, предложена от Кокс, с която за пръв път е демонстрирана възможността за асиметрична криптография.

(1) *Инициализация.* за създаване на системата A извършва следните стъпки:

   (a) A тайно избира две големи прости числа p и q, такива че p не дели $q-1$ и q не дели $p-1$. A публикува открития ключ $n = pq$.

   (b) A използва алгоритъма на Евклид за да намери числа r, s, удовлетворяващи

$$pr \equiv 1 \pmod{q-1}, \quad qs \equiv 1 \pmod{p-1}.$$

   (c) B използва оче веднъж алгоритъма на Евклид за да намери u, v, удовлетворяващи

$$up \equiv 1 \pmod{q} \quad vq \equiv 1 \pmod{p}.$$

   Тайният ключ на A е (p, q, r, s, u, v).

(2) *Шифриране.* B иска да изпрати съобщение, което е представено като редица от цели числа $m_1, \ldots, m_t$, където $0 \leq m_i \leq n$. B шифрира тези блокове като

$$c_i = m_i^n \pmod{n}$$

и изпраща шифрираните блокове на A.

(3) *Дешифриране.*

(a) A възстановява изпратените блокове $\bmod p$ и $\bmod q$, пресмятайки

$$a_i = c_i^s \pmod{p} \quad b_i = c_i^r \pmod{q}.$$

(b) A възстановява съобщението от

$$m_i = upb_i + vqa_i \pmod{n}.$$

Теорема 8.1. Криптосистемата на Кокс и Елис е коректно зададена.

Доказателство. От $up \equiv 1 \pmod{q}$ и $vq \equiv 1 \pmod{p}$ получаваме

$$upb_i + vqa_i \equiv a_i \pmod{p}, \quad upb_i + vqa_i \equiv b_i \pmod{q}.$$

Ако докажем, че $m_i \equiv a_i \pmod{p}$ и $m_i \equiv b_i \pmod{q}$, то съгласно Китайската теорема за остатъците ще имаме

$$upb_i + vqa_i \equiv m_i \pmod{n},$$

с което ще е доказано, че дешифрирането работи.

Ако $m_i \neq 0 \pmod{p}$, то работейки по модул p, имаме

$$c_i^s \equiv m_i^{ns} \equiv m_i^{sqp} \pmod{p}.$$

Тогава от $sq \equiv 1 \pmod{p-1}$ следва, че $sq = 1 + k(p-1)$ за някакво цяло число k. От малката теорема на Ферма получаваме

$$c_i^s \equiv m_i^{(1+k(p-1))p} \equiv m_i^p \equiv m_i \pmod{p}.$$

Следователно $m_i \equiv a_i \pmod{p}$. Това сравнение е тривиално изпълнено и когато $m_i \equiv 0 \pmod{p}$.

По подобен начин доказваме, че $m_i \equiv b \pmod{q}$. така от китайската теорема за остатъците следва че дешифрирането наистина възстановява съобщението m_i. □

8.2 RSA

Най-известната и широко използвана асиметрична криптосистема е анонсирана криптосистемае създадена от Рон Ривест, Ади Шамир и Лен Ейделман през 1977 г.[**?**]. мето на системата е абревиатура от имената на създателите ѝ. Системата е сходна с тази на Кокс и Елис. както и при Кокс и Елис сигурността ѝ се основава на трудността на задачата за разлагане на прости множители на числа, за които е известно,

че са произведение на две прости числа. Да отбележим, еч не същецвува формално доказателство, че задачата за разлагане на прости множители е трудна (т.е. не е в P). Такова доказателство няма дори иза специалния случай на задачата, използван в RSA. Не е извстно и доказателство на твърдението, че разлагането на простимножители е необходимо за криптанализа на RSA (достатъчността е очевидна), т.е. че не съществува криптанализ, заобикалящ факторизацията. Най-напред ще опишем системата.

(1) *Инициализация на системата.* За да създаде двойка ключове (открит и таен ключ) A извършва следните стъпки:

- A избира две големи прости числа p и q, $p < q$, с дължина поне 512 бита и пресмята произведението им $n = pq$.
- A пресмята $\varphi(n) = (p-1)(q-1)$.
- A генерира случайно цяло число $e \in \mathbb{N}$, $1 < e < \varphi(n)$, за което $\gcd(e, \varphi(n)) = 1$.
- A пресмята $d \in \mathbb{N}$, $1 < d < \varphi(n)$, такова че

$$ed \equiv 1 \pmod{n}.$$

Числото n наричаме *модул* на криптосистемата, а e и d – съответно *шифрираща* и *дешифрираща експонента.* Двойката (n, e) е откритият (публичният) ключ на системата,а d е нейният таен ключ. Макар да неса необходими при дешифрирането, простите числа p и q, както и $\varphi(n)$, трябва да бъдат пазени в тайна и въобще е най-добре да се унищожат след създаването на системата. Често шифриращата експонента се избира по специален начин за увеличаване на бързодействието на системата. Типични стойности за e в този случай са $e = 3, 17, 65537$.

(2) *Шифриране.* Да приемем, че B иска да изпрати съобщение на A. Съобщението е представено като цяло число m, $0 \leq m < n$; ако съобщението е с по-голяма дължина, то се разбива на редица от числа, ненандхвърлящи n. Криптотекстът се получава по правилото

$$c = m^e \pmod{m}.$$

(3) *Дешифриране.* За да дешифриракриптотекста A пресмята

$$m' = c^d \pmod{m}.$$

Тук m' е цяло число в интервала $[0, n)$.

Теорема 8.2. *Нека $n \in \mathbb{Z}^+$ и нека e, d са цели числа удовлетворяващи $1 \leq e, d \leq \varphi(n)$ и $ed \equiv 1 \pmod{\varphi(n)}$. Ако $m \in \mathbb{Z}$ и $c \equiv m^e \pmod{n}$, то $m \equiv c^d \pmod{n}$. В случай, че $m < n$, то $m = c^d \pmod{n}$.*

Доказателство. Нека нито p нито q дели m. Тогава съгласно условието съществува такова цяло число j, за което $ed = j\varphi(n) + 1$. От Теоремата на Ойлер

$$c^d \equiv (m^e)^d \equiv m^{j\varphi(n)+1} \equiv m(m^{\varphi(n)})^j \equiv m \pmod{n}.$$

Ако p дели m, но q не дели m, имаме

$$c^d \equiv m^{ed} \equiv^{j\varphi(n)+1} \equiv (m^{q-1})^{j(p-1)} m \equiv m \pmod{q}.$$

Същото сравнение е очевидно в сила и *modp*. Следователно $c^d \equiv m \pmod{n}$.

Теоремата е тривиално изпълнена, ако p и q едновременно делят n. □

Пример 8.3. Нека $p = 11$, $q = 19$. Тогава $n = 209$ и $\varphi(n) = 180$. Да изберем $e = 7$, $\gcd(7, 180) = 1$. С помощта на разширения алгоритъм на Евклид получаваме $d = -77 = 103$ (по модул 180). Да шифрираме съобщението $m = 5$. Имаме

$$5^1 \equiv 5 \pmod{209}, 5^2 \equiv 25 \pmod{209}, 5^4 \equiv -2 \pmod{209},$$

откъдето $5^7 \equiv 5 \cdot 25 \cdot (-2) \equiv -41 \pmod{209}$. Следователно $c = 168$.

За дешифрирането пресмятаме

$$-41^1 \equiv -41, (-41)^2 \equiv 9, (-41)^4 \equiv 81, (-41)^8 \equiv 82, (-41)^{16} \equiv 36,$$
$$(-41)^{32} \equiv 42, (-41)^{32} \equiv 92 \pmod{209}.$$

Сега

$$\begin{aligned} (-41)^{103} &\equiv (-41)^{64} \cdot (-41)^{32} \cdot (-41)^4 \cdot (-41)^2 \cdot (-41)^1 \pmod{209} \\ &\equiv 92 \cdot 42 \cdot 81 \cdot 9 \cdot (-41) \pmod{209} \\ &\equiv 5 \pmod{209}, \end{aligned}$$

което и трябваше да се получи.

Пример 8.4. Ето едни ммалко по-големи параметри, но все още далеч под изискванията за сигурност (p и q са с дължина около 32 бирта, а n е около 64 бита).

$$\begin{aligned} p &= 3\ 336\ 670\ 033 \\ q &= 9\ 876\ 543\ 211 \\ n &= 32\ 954\ 765\ 761\ 773\ 295963 \\ \varphi(n) &= 32\ 954\ 765\ 748\ 560\ 082720 \\ e &= 1031 \\ d &= 31\ 963\ 885\ 304\ 131\ 991 \end{aligned}$$

Сега ще направим няколко общи бележки по параметрите на системата. Пър вият въпрос, който възниква, е генерирането на големи прости числа. С тази задача ще се занимаем по-детайлно в следващите лекции. Засега ще отбележим само, че генерираните прости числа трябва да са случайни, а не от специален вид (напр. прости числа на Мерсен) или взети от някоя таблица. Трябва да се използва генератор на случайни числа, с който да се получи редица от цифри (десетични или двоични). След това трябва да се провери дали числото, предствено от тази редица от цифри е просто. За да се избегнат тривиалности, тази редица завършва с нечетна цифра (или с 1, ако записът е двоичен). Въпросът за това, на какъв точно тест подлагаме това число, отлагаме за по-късно. Сега само ще проверим, че простите числа са

достатъчно гъсто разположени и броят на тестовете, които ще трябва да извършим, не е неразумно голям.

От теоремата за простите числа (PNT) имаме, че

$$\lim_{x\to\infty} \frac{\pi(x)}{x/\log x} = 1,$$

откъдето можем да приемем, че броят на простите числа по-малки или равни на x е приблизително $x/\log x$. Така броят на t-цифрените прости числа е приблизително

$$\frac{10^t}{\log 10^t} - \frac{10^{t-1}}{\log 10^{t-1}},$$

вероятността едно случайно избрано нечетно t-цифрено число да е просто е

$$\frac{\frac{10^t}{\log 10^t} - \frac{10^{t-1}}{\log 10^{t-1}}}{\frac{10^t - 10^{t-1}}{2}} = \frac{20t-22}{9t(t-1)(\log 10)}.$$

За $t = 100$ това дава вероятност от ≈ 0.00864, за $t = 150$ – вероятност от ≈ 0.00643 и за $t = 200$ – вероятност от ≈ 0.00482. Така ако искаме да генерираме 200-цифрено просто число, то очакваният брой опити е малко по-голям от 200. За $t = 310$ (число с 1024 двоични цифри има дължина 309) тази вероятност е приблизително ≈ 0.0028 (т.е. очакваме веднъж на 357 опита да получаваме просто число). С други думи, простите числа са достатъчно много и не трябва да пробваме твърде дълго за да попаднем на такова (ако изборът ни е наистина случаен).

Числата p и q трябва да бъдат "далеч" едно от друго. Ако допуснем, че $p < q$ и че $q - p$ е "малко", т.е. тези числа са почти равни, то $(p+q)/2$ е малко по-голямо от $\sqrt{n}$:

$$\frac{(q+p)^2}{4} - n = \frac{(q-p)^2}{4},$$

където дясната страна е точен квадрат. Можем да разложим n чрез следния прост алгоритъм: за всяко $x > \sqrt{n}$ образуваме числото $x^2 - n$ и проверяваме дали то е точен квадрат. Ако това е така (например, $x^2 - n = y^2$), получаваме $p = x + y$, $q = x - y$. Нека $n = 97343$, $\sqrt{n} = 311.998$. Тогава $312^2 - n = 1$ и $x = 312$, $y = 1$, откъдето $p = 313$, $q = 311$. За да избегнем възможността от такава атака е достатъчно да изберем p и q "далеч" едно от друго. Това се постига лесно, ако при генерирането им осигурим разлика в дължините им от няколко бита.

Генерирането на двойката (e, d) е лесно с помощта на разширения алгоритъм на Евклид. Както вече отбелязахме, шифриращата експонента често не е случайна, а се избира така, че шифрирането да е бързо. Операцията повдигане на степен по модул m може да бъде извършена ефективно с приблизително $2\log_2 m$ умножения (Appendix X.) Ще отбележим, че вероятността случайно избран открит текст да има общ делител с n, което би довело до случайно разлагане на n (като пресметнем $\gcd(m, n)$) е много малка. Тя е равна на

$$\frac{n - \varphi(n)}{n} = \frac{1}{p} + \frac{1}{q} - \frac{1}{pq} < \frac{1}{p} + \frac{1}{q}.$$

Ако изберем p и q да са с дължина 512 бита, то тази вероятност е $< 1/2^{512}$. Да отбележим, че такава е вероятността за случайно отгатване на прост делител на n.

При всеки избор на параметри за RSA съществуват открити текстове, които се шифрират в себе си. Те са решение на сравнението $x^e \equiv x \pmod{n}$, или, еквивалентно на системата

$$x^e \equiv x \pmod{p}, x^e \equiv x \pmod{q}.$$

Не трудно да се покаже, че броят на откритите текстове, които се шифрират в себе си е равен на $(1+\gcd(e-1,p-1))(1+\gcd(e-1,q-1))$. Тъй като $\gcd(e-1,p-1) \geq 2$ (от $\gcd(e,\varphi(n)) = 1$ следва, че e е нечетно), то поне девет открити текста се шифрират в себе си. Този брой зависи от избора на e. Така един особено лош избор е

$$e = \frac{\varphi(n)}{2} + 1.$$

Тогава имаме $(1+\gcd(e-1,p-1))(1+\gcd(e-1,q-1)) = pq$, т.е. за тази стойност на e всеки открит текст се шифрира в себе си.

Пример 8.5. за $p = 5$, $q = 11$, $n = 55$, $\varphi(n) = 40$, $e = 7$, $d = 23$ имаме девет открити текста, които се шифрират в себе си. Точно четири от тях не се делят нито на 5, нито на 11. Това са 1, 21, 34, 54.

Вече отбелязахме, еч не е известно дали задачата за разбиването на RSA (разбирана като задачата за намиране на тайния ключ d при зададени n и e) е еквивалентна на задачата за разлагане на n на прости множители. Сега ще покажем, че знанието на d дава ефективен метод за разлагане на n.

Теорема 8.6. Дадена е криптосистема *RSA* с публичен ключ (n,e). Ако е известна дешифриращата експонента d, то числото n може ефективно да бъде разложено на множители.

Доказателство. Съществува цяло число k, за което

$$ed = 1 + k(p-1)(q-1).$$

Нека $ed - 1 = 2^s t$, t – нечетно число. Тогава

$$a^{ed-1} = a^{2^s t} \equiv 1 \pmod{n}$$

за всяко a, за което $\gcd(a,n) = 1$. Броят на елементите $a \in \mathbb{Z}_n^*$, за които съществува цяло число $i \in \{1,\ldots,s\}$, за което

$$a^{2^{i-1}t} \not\equiv \pm 1 \pmod{n}, a^{2^i t} \equiv 1 \pmod{n}$$

е поне $\frac{1}{2}|\mathbb{Z}_n^*| = \frac{1}{2}\varphi(n)$. за тези a числото $\gcd(a^{2^{i-1}t} - 1, n)$ е нетривиален делител на n (т.е. това е p или q), тъй като

$$0 \equiv a^{2^i t} - 1 = (a^{2^{i-1}t} - 1)(a^{2^{i-1}t} + 1) \pmod{n}$$

и никой от двата множителя вдясно не се дели на n. следователно,при случаен избор на a, $\gcd 9a, n) = 1$, вероятността $\gcd(a^{2^{i-1}t} - 1, n)$ да е нетривиален делител на n е $1/2$. Очакваният брой опити да попаднема на такова a е 2. □

Съгласно теорема 8.6 система, в която се използва общ модул, е несигурна. В такава система всеки участник би могъл да получи лесно дешифриращите експоненти на всички останали потребители. наистина, ако допуснем, че имаме система с модул n, който се използва от A с двойка ключове (e_A, d_A) и от B с двойка ключове (e_B, d_B), то A може да използва d_A за да получи разлагането на $n = pq$, след това да пресметне $\varphi(n) = (p-1)(q-1)$ и накрая да пресметне d_B като $d_B = e_B^{-1} \pmod{\varphi(n)}$, използвайки разширения алгоритъм на Евклид.

Сега ще предположим, че системата използва общ модул, но опонентът не е потребител от системата (т.е. той не знае нито една дешифрираща експонента). Нека A, който е законен потребител, изпраща едно и също съобщение на двама различни участници в системата с двойки ключове (e_1, d_1) и (e_2, d_2). Опонентът вижда две шифрирани съобщения:

$$c_1 \equiv m^{e_1} \pmod{n}, c_2 \equiv m^{e_2} \pmod{n}.$$

Тъй като разполага с e_1 и e_2 опонентът може да изчисли

$$\begin{aligned} t_1 &= e_1^{-1} \pmod{e_2}, \\ t_2 &= \frac{t_1 e_1 - 1}{e_2}. \end{aligned}$$

Ясно е, че числото t_2 е цяло. Сега съобщението m може да се възстанови от следното изчисление:

$$\begin{aligned} c_1^{t_1} c_2^{-t_2} &\equiv m^{e_1 t_1} m^{-e_2 t_2} \pmod{n} \\ &\equiv m^{1+e_2 t_2} m^{-e_2 t_2} \pmod{n} \\ &\equiv m \pmod{n}. \end{aligned}$$

Дотук показахме, че знанието на тайната експонента d позволява да се намерят простите делители на n. Оказва се, че знанието на $\varphi(n)$ също позволява ефективното разлагане на модула на системата.

Теорема 8.7. Нека е дадена криптосистема *RSA* с модул n, за който е известна стойността на функцията на Ойлер $\varphi(n)$. Тогава простите множители на n могат да бъдат пресметнати ефективно.

Доказателство. От $\varphi(n) = (p-1)(q-1) = n - (p+q) + 1$, можем да намерим

$$s := p + q = n - \varphi(n) + 1.$$

Сега p и q са корени на уравнението $x^2 - sn + n = 0$ и могат да се определят ефективно от

$$p = \frac{s + \sqrt{s^2 - 4n}}{2}, \quad q = \frac{s + \sqrt{s^2 + 4n}}{2}.$$

□

Случаите, когато $\varphi(n)$ има само малки прости делители, трябва да бъдат избягвани. Да допуснем, че всички прости делители r на $\varphi(n)$ са по-малки от някакво цяло число k (не много голямо). Най-високата степен на r, която дели $\varphi(n)$ е $\lfloor \log_r n \rfloor$. Сега

можем да генерираме всички кандидати v за $\varphi(n)$, повдигайки криптотекста на степен $\frac{v+1}{e}$, когато това число е цяло.

За да се избегнат такива ататки трябва да се използват само сигурни прости числа. Едно просто число p се счита за сигурно, ако $\frac{p-1}{2}$ също е просто число; такива числа са, например, 83, 107, $10^{100} - 166157$.

Вече беше отбелязано, че с цел ускоряване на шифрирането RSA се използва с малки шифриращи експоненти e. Това може да доведе до допълнителни проблеми със сигурността. да предположим, че имаме трима потребители с различни модули n_1, n_2, n_3 и обща шифрираща експонента $e = 3$. Нека е изпратено едно и също открито съобщение m, шифрирано с трите различни открити ключа: $(3, n_1)$, $(3, n_2)$, $(3, n_3)$. Опонентът вижда три криптограми:

$$\begin{aligned} c_1 &\equiv m^3 \pmod{n_1}, \\ c_2 &\equiv m^3 \pmod{n_2}, \\ c_3 &\equiv m^3 \pmod{n_3}. \end{aligned}$$

Можем да считаме, че $\gcd(n_i, n_j) = 1$ за $i \neq j$. В противен случай получаваме тривиално разлагане на някои от модулите и разбиване на поне две от системите. С помощта на Китайската теорема за остатъците той намира решение x_0 на системата

$$\begin{aligned} x &\equiv c_1 \pmod{n_1}, \\ x &\equiv c_2 \pmod{n_2}, \\ x &\equiv c_3 \pmod{n_3}, \end{aligned}$$

за което $0 \leq x_0 < n_1 n_2 n_3$. Имаме $x_0 \equiv m^3 \pmod{n_1 n_2 n_3}$, но доколкото $0 \leq m^3 < \min n_1^3, n_2^3, n_3^3 \leq n_1 n_2 n_3$, то $x_0 = m^3$. Сега m може да бъде пресметнато като $m = \sqrt{x_0^3}$.

Една очевидна предпазна мярка срещу тази атака е добавянето на известен брой случайни цифри към всяко съобщение. разбира се, можем да решим да избягваме малки шифриращи експоненти. Следва да отбележим, че при използването на RSA за цифров подпис може безпроблемно да се вземат малки шифриращи експоненти. Широко разпространен избор е

$$e = 65537 - 2^{2^4} + 1.$$

Ще отбележим също, че малко d компрометира системата. В този случай е възможна т.нар. итерирана атака срещу RSA. При получен криптотекст $c_0 = c$ пресмятаме последователно

$$\begin{aligned} c_1 &= c_0^e \bmod n, \\ c_2 &= c_1^e \bmod n, \\ \dots &\quad \dots \\ c_i &= c_{i-1}^e \bmod n, \\ \dots &\quad \dots \end{aligned}$$

докато намерим c_i, което е смислено. Вероятността за успех на такава атака е пренебрежимо малка, когато $p-1$ и $q-1$ имат големи прости делители p', q', а $p'-1$ и $q'-1$ също имат големи прости делители.

8.3 Вариант на Рабин

Към настоящия момент не е известен криптанализ на RSA, който да е различен от разлагането на n на прости множители. От друга страна не е известно дали хаистина криптанализът на RSA е еквивалентен на разлагането на n. Рабин [43] предлага вариант на RSA, чийто криптанализ е доказуемо еквивалентен на задачата за разлагане на прости множители (когато n е произведение на две различни прости прости числа). В този вариант всеки потребител използва шифриращa експонента $e = e_U = 2$. Тъй като $\gcd(e, \varphi(n)) = 2$, шифрирането не е биективно изображение. Наистина, ако $c \equiv m^2 \ (n)$, където $n = pq$, то за възстановяването на m трябва да решим системата

$$\begin{aligned} x^2 &\equiv c \pmod{p} \\ x^2 &\equiv c \pmod{q}. \end{aligned}$$

Всяко от тези сравнения има по две решения, съответно $\pm u$ и $\pm v$, и съгласно Китайската теорема за остатъците системата има четири решения $\pm au \pm bv$, където a и b удовлетворяват

$$\begin{aligned} a &\equiv 1 \pmod{p} \qquad & a &\equiv 0 \pmod{p} \\ b &\equiv 0 \pmod{q} \qquad & b &\equiv 1 \pmod{q}. \end{aligned}$$

Така дешифрирането се свежда до решаване на сравнения от вида

$$x^2 \equiv c \pmod{p}, \tag{8.1}$$

където p е просто число. за $p \equiv 3 \pmod 4$ това сравнение има очевидното решение $x = c^{\frac{p+1}{4}} \pmod{p}$. наистина лесно се проверява, че

$$x^2 \equiv (c^{\frac{p+1}{4}})^2 \equiv c^{\frac{p+1}{2}} \equiv (c^{\frac{p-1}{2}}) \cdot c \equiv c \pmod{p}.$$

В случая, когато $p \equiv 1 \pmod 4$, не съществува бърз детерминистичен алгоритъм, който да решава (8.1). В този случай можем да посочим ефективан вероятностен алгоритъм.

Да означим с Q множеството от квадратичните остатъци, а с N – множеството на квадратичните неостатъци подул p. Нека означим с r и s решенията на (8.1). Тогава $r + u$ и $s + u$ са решенията на

$$(x - u)^2 \equiv c \pmod{p}. \tag{8.2}$$

Ако u пробягва $\{0, 1, \ldots, p-1\} \setminus \{-s\}$, то $\dfrac{r+u}{s+u} \pmod{p}$ пробягва $\{0, 1, \ldots, p-1\} \setminus \{1\}$. За половината от допустимите стойности на u елементът $\dfrac{r+u}{s+u}$ ще бъде в $Q \cup \{0\}$, а за другата половина – този елемент ще е в $\mathbb{N}$. Ако е налице първата алтернатива, то

- или $r + u, s + u \in Q$,
- или $r + u, s + u \in N$,
- или $u = -r$.

Ако е налице втората алтернатива, то

или $r+u \in Q, s+u \in N$,
или $r+u \in N, s+u \in Q$.
Ако u приема с равна вероятност всяка стойност от $\{0,1,\ldots,p-1\} \setminus \{-s\}$, то всяка от алтернативите се случва с вероятност $\frac{1}{2}$. Тъй като

$$\prod_{a \in Q}(x-a) \equiv x^{\frac{p-1}{2}} \pmod{p},$$

то с вероятност $\frac{1}{2}$ полиномът

$$\gcd((x-u)^2-c, x(x^{\frac{p-1}{2}}-1))$$

е линеен. Този полином е равен на
- $x-u-r$, ако $u+r \in Q, u+s \in N$, или
- $x-u-s$, ако $u+r \in N, u+s \in Q$.

Ако $u=-s$, то $(x-u)^2-c = x^2+2sx$ и в този случай $\gcd((x-u)^2-c, x(x^{\frac{p-1}{2}}-1)) = x$. Във всички останали случаи най-големият общ делител е константа или полином от степен 2. При $\frac{p+1}{2}$ избора на u стигаме до разлагане на n и, следователно, вероятността за разлагане е $\frac{p+1}{2p}$.

Сега ще разгледаме проблема с неоднозначността на дешифрирането. Ще приемем, че простите числа p и q са от вида $4k+3$ (в този случай нямаме проблеми с решаванетона $x^2 \equiv c \pmod{p}$). Въпросът е да определим, кое е истинското решение на $x^2 \equiv c \pmod{n}$, т.е. това, което съответства на открития текст m. Отбелязахме, че четирите решения имат вида $\pm au \pm bv$, където u е решение на $x^2 \equiv c \pmod{p}$, а v е решение на $x^2 \equiv c \pmod{q}$. Без ограничение на общността имаме следните възможности:

- $au+bv \in Q \mod p, au+bv \in Q \ (q)$;
- $-au-bv \in N \mod p, -au-bv \in N \ (q)$;
- $au-bv \in Q \mod p, au-bv \in N \ (q)$;
- $-au+bv \in N \mod p, -au+bv \in Q \ (q)$.

Тук използвахме, че $(-1/p) = (-1/q) = -1$. Сега за символа на Якоби имаме

$$\left(\frac{au+vb}{pq}\right) = \left(\frac{-au-vb}{pq}\right) = 1.$$

При това едно двете числа $au+vb$ и $-au-bv$ е в интервала $(0,n)$,другото е в интервала $(n/2,n)$. Можем да определим откритите текстове като тези решения m' на сравнението, за които

$$\left(\frac{m'}{n}\right) = 1 \text{ и }) < m' < \frac{n}{2}.$$

Ако съобщението е със специална структура, напр. текст на английски език, то проблем не съществува, тъй като с огромна вероятност само едно от четирите решения ще води до смисле тексът. Ако съобщенията нямат специална структура, то

едно възможно решение е да допълним съобщението с определен брой нули преди шифрирането. така дешифрираното събщение трябва да завършва с правилния брой нули.

Сега ще докажем интересния факт, че разбиването на варианта на Рабин е еквивалентно на разлагането на n на прости множители.

Теорема 8.8. Нека $n = pq$, p и q – прости числа,. Нека Нека A е алгоритъм, който намира решение на $x^2 \equiv c \pmod{n}$ във $F(n)$ стъпки за всяко c, което е квадрат по модул n. Тогава съществува вероятностен алгоритъм, който разлага n в (очакван брой) $2(F(n) + 2\log n)$ стъпки.

Доказателство. Избираме случайно число m, $0 < m < n$, и решаваме сравнението $x^2 \equiv m^2 \pmod{n}$ във $F(n)$ стъпки с помощта на алгоритъма A. Нека k е е дно от четирите решения на $x^2 \equiv m^2 \pmod{n}$. Всяка от следните възможности се реализира с вероятност 1/4:

1) $k \equiv m \pmod{p}, k \equiv m \pmod{q}$;

2) $k \equiv m \pmod{p}, k \equiv -m \pmod{q}$;

3) $k \equiv -m \pmod{p}, k \equiv m \pmod{q}$;

4) $k \equiv -m \pmod{p}, k \equiv -m \pmod{q}$.

В случай 2) имаме $\gcd(k-m, n) = p$, а в случай 3) – $\gcd(k-m, n) = q$. Следователно пресмятането на $\gcd(k-m, n)$ намира разлагането с вероятност 1/2. Това пресмятане изисква $2\log n$ стъпки. така при всеки избор за m ще извършваме $F(n) + 2\log n$ стъпки като вероятността за успех е 1/2. Очакваният брой опити до намиране на разлагането на n е два, което е и твърдението на теоремата. □