1995	6к	10м	16 G
2000	15к	43м	103 G
2005	38к	180м	658 G
2010	96к	750м	4.2 T
2015	239к	3.1G	27 T
2020	549к	13G	173 T

Одан кейін, Р. Л. Ривест кілттердің қауіпсіз ұзындығы туралы өз болжамын жасады.

Кілттердің қауіпсіз ұзындығы туралы Л. Ривесттің болжамы:

Жыл	Жеке	Корпорация	Мемлекет
1990	398	515	1289
1995	405	542	1399
2000	422	572	1512
2005	439	602	1628
2010	455	631	1754
2015	472	661	1884
2020	489	677	2017

Кілттерді ашудың барлық тәсілдері көп компьютерлерді параллель қолдану арқылы іске асады. 129-мәнде сан туралы қысқаша айтайық. 1977 ж. белгілі математик Мартин Гарднер ScietificAmerican-ға шифрды ашу үшін 129 мәнді санды көбейткішке жіктеу шифрын жариялады.

N=11 43 81 62 57 57 88 88 67 66 92 35 77 99 76 14 66 10 01 02 18 29 67
21 24 36 25 62 56 18 42 93 57 06 93 52 45 73 38 97 83 05 97 12 35
63 95 87 05 0589890 7514 75 99 29 00 26 879543541

1600 машина қатысқан, қарапайым жеке компьютерден бастап. GrayC90 станциясына дейінгі, уақытша «суперкомпьютер»

ұйымдастырылған. 5000-ға жуық мипсолет қажет ететін тапсырма нәтижесінде 1993 жылдың тамызынан 1994 жылдың сәуіріне дейін шешілген болатын. Жалпы Internet барлық машиналары N-ді 3 сағатта факторизациялауға болатындығын ескеруіміз керек.

3.4. Қалдықтар туралы Қытай теоремасының қолдануымен жылдам шифрларды шешу

RSA (Ron Rivest, Adi Shamir, Leonard Adleman) дешифрлау және қолтаңба амалдары, яғни C берілгенде төмендегіні есептеу кезінде:

$$M = C^d \pmod N$$

қалдықтар туралы Қытай теоремасының көмегімен жылдам орындалуы мүмкін, себебі пайдаланушы $N = p \times q$ модулінің көбейткішін біледі.

$i = 1,2,3,...,k$ үшін p_i өзара қарапайым жай сандар делік, яғни

$$i \neq j \text{ үшін } EYOБ(p_i, p_j) = 1.$$

$i = 1,2,3,...,k$ үшін u_j белгілі болғанда, қалдықтар туралы Қытай теоремасы $0, p-1$ аралығында жалғыз бүтін u саны бар делік, мұндағы $p = p_1, p_2,..., p_k$ мынадай $u = u_i \pmod{p_i}$ қалдықтар туралы Қытай теоремасы есептеуді

$$M = C^d \pmod{p \cdot q}$$

екі бөлікке келесідей түрде бөлуге болатыны туралы айтады:

$$M_1 = C^d \pmod p$$
$$M_2 = C^d \pmod q.$$

Одан кейін M ақырғы мәні қалдықтар туралы Қытай теоремасын пайдаланып есептеледі. Бұл сияқты есептеуге арналған 2 алгоритм бар: бір негізбен жұмыс істейтін алгоритм-SRC және аралас негізбен жұмыс жасау алгоритмі - MRC. Енді осы алгоритмдерді қысқаша

сипаттайық. $u_1, u_2, ..., u_k$ және $p_1, p_2, ..., p_k$ алгоритмдерін қарастырамыз. SRC алгоритмі $u-$ді қосындылау көмегімен есептейді

$$u = \sum_{i=1}^{k} u_i c_i P_i (\mathrm{mod}\, P),$$

мұнда $$P_i = p_1 p_2 ... p_{i-1} p_{i+1} ... p_k = P / P_i,$$

ал $c_i P_i$ модулі бойынша P_i-нің мультипликациялық инверсиясы, яғни

$$c_i P_i = 1 (\mathrm{mod}\, p_i),$$

бұдан кейін SRC-ті RSA дешифрлауға пайдалана отырып, біз алдымен мынаны есептейміз:

$$M_1 = C^d (\mathrm{mod}\, p)$$
$$M_2 = C^d (\mathrm{mod}\, q).$$

Алайда Ферма теоремасын көрсеткіштерге қолданғанда, бізге тек келесілерді есептеу қажеттілігі анықталады:

$$M_1 = C^{d_1} (\mathrm{mod}\, q)$$
$$M_2 = C^{d_2} (\mathrm{mod}\, p),$$

мұнда

$$d_1 = d \bmod (p-1)$$
$$d_2 = d \bmod (q-1).$$

Бұл біршама үнемділікті қамтамасыз етеді, себебі $d_1, d_2 < d,$ шын мәнінде d_1 және d_2 мәндері $d-$дан екі еседей кіші. SRC алгоритміне сәйкес біз $M-$ді есептеуге мына қосындыны пайдаланамыз:

$$M = M_1 c_1 pq / p + M_2 c_2 pq / p (\mathrm{mod} N) = M_1 c_1 q M_2 c_2 p (\mathrm{mod} N),$$

мұнда $c_1 = q^{-1} (\mathrm{mod}\, p)$ және $c_2 = p^{-1} (\mathrm{mod}\, q)$. Бұдан

$$M = M_1 (q^{-1} \bmod p) q + M_2 (p^{-1} \bmod q) p (\mathrm{mod} N)$$

дәлелдеу үшін атап көрсетеміз:

$$M(\mathrm{mod}\, p) = M_1 \cdot 1 + 0 = M_1$$
$$M(\mathrm{mod}\, p) = 0 + M_2 \cdot 1 = M_2\text{'}$$

екінші жағынан MRC алгоритмінің ақырғы мәнін есептейді, ол үшін алдымен мәндердің триагулярлық кестесін есептеп алады:

$$u_{11}$$
$$u_{21}u_{22}$$
$$u_{31}u_{32}u_{33}$$
$$\ldots\ldots\ldots\ldots$$
$$\ldots\ldots\ldots\ldots$$
$$u_{k1}u_{k2}\ldots\ldots u_{kk}$$

Мұнда u_{i1} мәнінің бірінші бағаны бұл u_i шамасының берілгендері, яғни

$$u_{i1} = u_i$$

өзге бағандардың мәні келесі рекурсия көмегімен алдыңғы бағандар мәндерінен біртіндеп есептеледі.

$$u_{i,j+1} = (u_{ij} - u_{jj})c_{ij} (\mathrm{mod}\, p_i),$$

мұнда c_{ij} - p_i модулі бойынша p_j мәнінің мультипликациялық инверсиясы болып табылады, яғни

$$c_{ij} p_i = 1(\mathrm{mod}\, p_i).$$

Мысалы, u_{32} былай есептеледі:

$$u_{32} = (u_{31} - u_{11})c_{13} (\mathrm{mod}\, p_3),$$

мұндағы c_{ij} p_2-ң модулі бойынша инверсиясы.

u бойынша ақырғы мәні есептеледі

$$u = u_{11} + u_{22}p_1 + u_{33}p_1p_2 + ... + u_{kk}p_1p_2 + ... + p_{k-1}$$

Ол p модулі бойынша қалдықты ақырғы есептеуді талап етпейді. RSA дешифрлауға MRC алгоритмін қолдана отырып, алдымен мынаны есептейміз:

$$M_1 = C^{d_1} (\mathrm{mod}\, q)$$
$$M_2 = C^{d_2} (\mathrm{mod}\, q)$$

Мұнда d_1 және d_2 алдыңғы алгоритмдегі сандар. Бұл жағдайда триангулярлық кесте айтарлықтай кішкентай және мыналардан тұрады:

M_{11}
$M_{21}\ M_{22}$

мұнда $M_{11} = M_1$, $M_{21} = M_2$ және $M_{22} = (M_{21} - M_{11})(p^{-1} \bmod q)(\mathrm{mod}\, q)$ болады.

Бұдан кейін M келесі формула бойынша есептеледі:

$$M = M_1 + [(M_2 - M_1) \cdot (p^{-1} \bmod q) \cdot \bmod q] \cdot p$$

Бұл қатынас дұрыс, себебі

$$M(\mathrm{mod}\, p) = M_1 + 0 = M_1$$
$$M(\mathrm{mod}\, q) = M_1 + (M_1 - M_2) \cdot 1 = M_2$$

MRC алгоритмі SRC алгоритміне қарағанда екі себепке байланысты қолайлы болып келеді:

- Ол жалғыз инверсияны есептеуді талап етеді: $p^{-1} \bmod q$

- Ол N модулі бойынша қалдықты ақырғы есептеуді талап етпейді.

$p^{-1} \bmod q$ кері мәні алдын ала есептеліп, сақталуы мүмкін. Алайда, p және q есептеу тәртібі PKCS-1 ашық кілті бар ұсынылған криптографиялық стандартта біздің белгілеуімізге кері екенін ескерейік. Жеке қолданылатын кілттің мәнін қамтитын деректер құрылымында келесі айнымалылар бар:

exponent 1 INTEGER, - $d \bmod(p-1)$
exponent 2 INTEGER, - $d \bmod(q-1)$
coefficient INTEGER, - $(inverceofq) \bmod p$.

Бұдан шығатыны, $p^{-1} \bmod q$ орнына $(q^{-1} \bmod p)$ пайдаланамыз. M_1 және M_2 жоғарыдай анықталады делік. p,q және M_1, M_2 орындарын ауыстырғанда алатынымыз.

$$M = M_2 + [(M_1 - M_2)(q^{-1} \bmod p) \bmod p] \cdot q$$

Бұл да дұрыс, өйткені

$$M(\bmod q) = M_2 + 0 = M_2$$
$$M(\bmod p) = M_2 + (M_1 - M_2) \cdot 1 = M_1$$

MRC алгоритмі SRC алгоритміне осы талап етілген еді.

p және q сандары $(k/2)$ биттік екілік сандар, ал $d-$ның үлкендігі N сияқты, ол өз кезегінде $k-$биттік бүтін сан болып табылады, енді MRC алгоритмі көмегімен RSA дешифрлау үшін биттік амалдардың жалпы санын есептейік. d_1 және $d_2, (p^{-1} \bmod q)$ шамалары алдын ала есептелген, ал дәрежеге шығару бинарлық әдіс негізінде орындалады деп алып, көбейту амалдарының санын бағалаймыз:

- $M_1 : 3/2(k/2) \cdot (k/2) -$ биттік көбейту амалдарын есептейміз;
- $M_2 : 3/2(k/2) \cdot (k/2) -$ биттік көбейту амалдарын есептейміз;
- M : есептеу, бір $(k/2)$-биттік алып тастау, екі $(k/2)$-биттік көбейту және бір $k-$биттік қосу амалы.

Көбейту амалдарында k^2 тәртібінде биттік амалдар, ал алу тәртібіндегі $k-$амалдар бар деп алып, биттік амалдардың жалпы санын біз былай есептейміз:

$$2\frac{3k^3}{4}(k/2)^2 + 2(k/2)^2 + (k/2) + k = \frac{3k^3}{8} + \frac{k^2+3k}{2}.$$

Екінші жағынан, CRT қолданбай алгоритмдер $(3/2)k, k-$биттік көбейту амалдарын пайдалана отырып, $M = C^d (\bmod N)$ тікелей есептейтін еді, ал ол $(3k^2/2)-$биттік амалдарды талап етеді. Бұдан

шығатыны, одан да жоғары тәртіп мүшелерін ескере отырып, біз былайша қортындылаймыз: CRT-ға негізделген алгоритм шамамен төрт есе жылдам болады.

13-кестеде қалдықтар туралы Қытай теоремасын пайдаланып, RSA шифрлау және дешифрлаудың жылдамдығы көрсетіледі.

13-кесте.

RSA-CRT алгоритмінің жұмыс істеу жылдамдығы

Жұмыс режимі	Модульдің ұзындығы, бит		
	96	88	64
Шифрлау	0.31 мс	0.26 мс	0.19 мс
Дешифрлау	1.01 мс	081 мс	0.85 с

3.5. Эль-Гамаль криптожүйесі

1985 жылы Эль-Гамальдің ұсынған алгоритмі ақпаратты шифрлау үшін және электрондық цифрлы қолтаңба үшінде қолдануға болатын еді.

Эль-Гамаль алгоритмінің қауіпсіздігі дискретті логарифмнің күрделі есептеуіне негізделген.

Құпия кілті және ашық кілті таңдау үшін P үлкен жай сан және Q үлкен бүтін сан алынады $(Q, P), P$ және Q, A және B пайдаланушыларға таратылып беріледі.

1. A және B пайдаланушылар өздерінің X құпия кілттерін таңдап алады.

Ол үшін X кездейсоқ бүтін сан таңдап алынады, мұнда $X < P$. Құпия кілті құпиялы түрде сақталуы қажет.

2. A және B пайдаланушылар

$$Y = Q^x mod\ P$$

есептеп шығарады. Y – ашық кілт.

3. Егер A пайдаланушы B пайдаланушыға M ақпаратты шифрлап жібергісі келсе, онда K кездейсоқ бүтін санын алады. $1 < K < P - 1$ болатындай $\big(K(P-1)\big) = 1$ яғни K және $(P-1)$ өзара жай сан болуы тиіс.

4. A пайдаланушы

$$a = Q^k mod\ P,$$
$$b = Y^k MmodP$$

есептеп шығарады.

Мұндағы a және b шифрланған мәтін болады. Бұл (a, b)-ы B пайдаланушыға жібереді.

5. B пайдаланушы (a, b) шифрланған мәтінді ашу үшін

$$M = b/a^x\ mod\ P(3.1)$$

есептеп шығарады,
яғни

$$a^x \equiv Q^{kx}\ mod\ P,$$
$$b/a^x \equiv Y^k\ M/a^x \equiv Q^{kx}\ M/Q^{kx} \equiv M\ (mod\ P)$$

тең болғандықтан (3.1) теңдік дұрыс екенін білдіреді.

Мысал $P\ =\ 11, Q\ =\ 2, X\ =\ 8$ деп аламыз, X- құпия кілті.

$$Y = Q^x\ mod\ P = 2^8\ mod\ 11 = 256\ mod\ 11 = 3.$$

Сонымен Y ашық кілт $Y\ =\ 3$ болды.

$M = 5$ болсын делік. $K = 9$ кездейсоқ санын алайық, яғни Е.Ү.О.Б. $(K, P - 1) = 1$ тексерейік. Демек Е.Ү.О.Б.$(9, 10)\ = 1$ орындалады, (a, b) сандарын есептейміз:

$$a = Q^k\ mod\ P = 2^9\ mod\ 11\ = 512\ mod\ 11 = 6,$$
$$b = Y^k MmodP = 3^9 *\ 5\ mod\ 11\ = 19683 * 5\ mod\ 11 = 9.$$

$(a, b)\ =\ (6, 9)$ шифр мәтінді аламыз.

Енді шифрланған мәтінді ашайық. M ақпаратты X құпия кілтті қолдану арқылы есептеп шығарамыз:

$$M = b/a^x\ mod\ P\ = 9/6^8\ mod\ 11.$$

$M = 9/6^8\ mod\ 11$ теңдеуін келесі түрге келтіруге болады:

$$6^8 * M \equiv 9 \, mod \, 11$$

немесе

$$1679676 * M \equiv 9 \, mod \, 11.$$

Бұл салыстыруды шешу арқылы $M = 5$ екенін табамыз.

3.6. Месси – Омур шифрлау алгоритмі

Месси – Омур шифрлау алгоритмінде шифрлаудың ашық кілті p жай саны болып табылады және бұл сан үшін $(p-1)$-ң үлкен жай бөлгіші табылады, ал құпия кілт тіпті болмайды. Хабардың шифрлануы және ашылуы жіберуші мен қабылдаушының бір-бірімен хабарласу кезінде өтеді.

Жіберуші A:

1. $(p-1)$ мен өзара жай кез келген e санын таңдайды, $0 < e < p-1$ және $d = e^{-1} (\mathrm{mod}\, p-1)$ есептейді.
2. $C = M^e (\mathrm{mod}\, p)$ есептеп, M алғашқы ашық мәтінді шифрлайды және C криптограммасын B қабылдаушыға жібереді.

C криптограмманы B қабылдаушы қабылдап:

1. $(p-1)$ мен өзара жай кез келген $e', 0 < e' < p-1$ санын таңдайды және $d' = (e')^{-1} (\mathrm{mod}\, p-1)$ есептейді.
2. $C' = C^e (\mathrm{mod}\, p)$ есептейді және осы C' қайта A жіберушіге береді.

Жіберуші A C' – ті d дәрежесіне шығарады:

$$M' = C'^d \, (\mathrm{mod}\, p)$$

және M' B қабылдаушыға жібереді.

A қабылдаушы $M = M'^{d'} \, (\mathrm{mod}\, p)$ есептей отыра, алғашқы ашық мәтінді алады.

3.7. Полига-Хеллманның шифрлау алгоритмі

Полига-Хелманның шифралу алгоритмі RSA шифрлау алгоритміне ұқсас. Бұл симметриялы емес алгоритм болып табылады, себебі шифрлау және ашу үшін әртүрлі кілттер қолданылады.

Сонымен қатар, бұл алгоритмді ашық кілтті криптожүйелер класына жатқызуға болмайды, себебі шифрлау және шифрді ашу кілттері бірін-бірі арқылы оңай алынады. Екі кілтті де (шифрлау және дешифрлау) құпияда сақтау керек.

RSA алгоритміндей C криптограммасы және M ашық мәтіні келесі қатынастардан анықталады:

$$C = M^e \bmod n$$
$$M = C^d \bmod n,$$

мұнда $e * d = 1$ (кейбір құрама санның модулі бойынша).

RSA алгоритмінен өзгешелігі – бұл алгоритмде n саны екі үлкен жай сандар арқылы анықталмайды, n саны құпия кілттің бір бөлігі болуы қажет. Егер кімде-кім e және n мәндерін білсе, ол d мәнін есептей алады.

Қарсылас e және d мәндерін білмесе, оған

$$e = \log_p C(\bmod n)$$

мәнді есептеуге тура келеді.

Бұл есеп қиын екені белгілі. Полига-Хеллманның шифрлау алгоритмі АҚШ-та және Канадада патенттелген.

3.8. Аралас шифралау әдісі

Ашық кілтті криптожүйелердің негізгі ерекшелігі – олардың деңгейі жоғары қауіпсіздігі болып табылады: құпия кілттің мәнін біреуге берудің, не хабарлаудың қажеттілігі жоқ, кілттің дұрыс екендігіне сендірудің де керегі жоқ. Симметриялы криптожүйелерде құпия кілтті жіберу кезінде оның ашылып қалуының қаупі бар.

Бірақ ашық кілтті криптожүйелер негізіндегі алгоритмдер келесі кемшіліктерге ие:

- жаңа құпия және ашық кілттердің генерациясы жаңа үлкен жай сандар генерациясына негізделген, ал сандардың жай екендігін тексеру көп процессорлық уақытты алады.