Von EUKLID zum RSA-Coding

von Jörg Waldvogel

Emeritenstamm, Winterthur 30.4.2007

1. Was ist RSA-Codierung?

Eine neue Art der Verschlüsselung geheimer Botschaften

Notation: Klartext, geheim, GROSS
Code, offen, klein

Beispiel:

HUETET␣EUCH␣AM␣MORGARTEN

ivfufuafvdiabnanpshbsufo

(i) Aus der Rätselecke: Ersetzungscode

Geheimer Schlüssel:

Codierung ↓	␣ A B C D .... X Y Z	Decode ↑
	a b c d e .... y z ␣	

Leider sofort & einfach knackbar

Comm. ACM 21 (1978), 120-126 Waldvogel 55

LABORATORY FOR COMPUTER SCIENCE (formerly Project MAC)

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

MIT/LCS/TM-82

A METHOD FOR OBTAINING DIGITAL SIGNATURES AND PUBLIC-KEY CRYPTOSYSTEMS

Ronald Rivest
Adi Shamir
Len Adleman

April 1977

545 TECHNOLOGY SQUARE, CAMBRIDGE, MASSACHUSETTS 02139

56

July 16, 1978

Dear Mr. Waldvogel,

Thank you for your letter of 6/27/78. Your remark about Bob sending his name with his message is correct and what other readers have mentioned.

I don't have any preprints or references to Schroeppel's factoring algorithm - it has not been published yet. I suspect it may appear soon, however. Although I don't have Schroeppel's address with me right now, you can obtain it from my secretary (Wendy Glasser) at MIT. (I'm in California at the moment).

Thanks again for your letter.

Sincerely,

Ronald L. Rivest

Xerox Corp.
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, Calif 94304

(ii) Neue Idee : RSA

Rivest, Shamir, Adleman :

Comm. ACM 21, p. 120-126, 1978

Vorbereitungsschritt :

- Segmentierung der Botschaft (z.B. 2 char)
- Umsetzung in Zahlen (kann offensichtlich sein)

z.B	␣	A	B	C	...	Y	Z
	00	01	02	03		25	26

HU ET ET ␣E UC H␣ AM ␣M OR GA RT EN

0821 0520 0520 0005 2103 0800 0113 0013 1518 0721 1820 0514

Codierungsschritt :

- Rein numerisch : Modulo-Rechnung
- $\boxed{\text{Codier-Algorithmus öffentlich}}$

Public-Key Cryptosystems

- Der Decodier-Algorithmus ist praktisch nicht zu finden

Übersicht :
2. Der Codier-Algorithmus
3. Decodierung. Euklid
4. Warum unknackbar?
5. Ergänzungen

2. Der Codier-Algorithmus

Publizierte Parameter: m, Modul
e, Codierexponent

$B \longrightarrow \boxed{b := B^e \bmod m} \longrightarrow b$

Segment des Klartexts $0 < B < m$

chiffriertes Textsegment $0 < b < m$

Bemerkungen:

(i) $r := z \bmod m$: Rest bei Division $\frac{z}{m}$, $0 \le r < m$

Also $z = m \cdot \mathrm{floor}\left(\frac{z}{m}\right) + r$

(ii) Man kann mit Restklassen rechnen:

z. B. $m = 10$; berechne $(327 \cdot 659) \bmod 10$

$\cdots = (327 \bmod 10) \cdot (659 \bmod 10) = 7 \cdot 9 \bmod 10 = 3$

(iii) Obiges Beispiel: Sei $m = 2747$, $e = 17$

$B = B_1 = 821 \Rightarrow b_1 = 821^{17} \bmod 2747$

Man muss das Monstrum 821^{17} nicht berechnen

$B^2 \bmod m = 821^2 \bmod m = 674041 \bmod m = 1026$

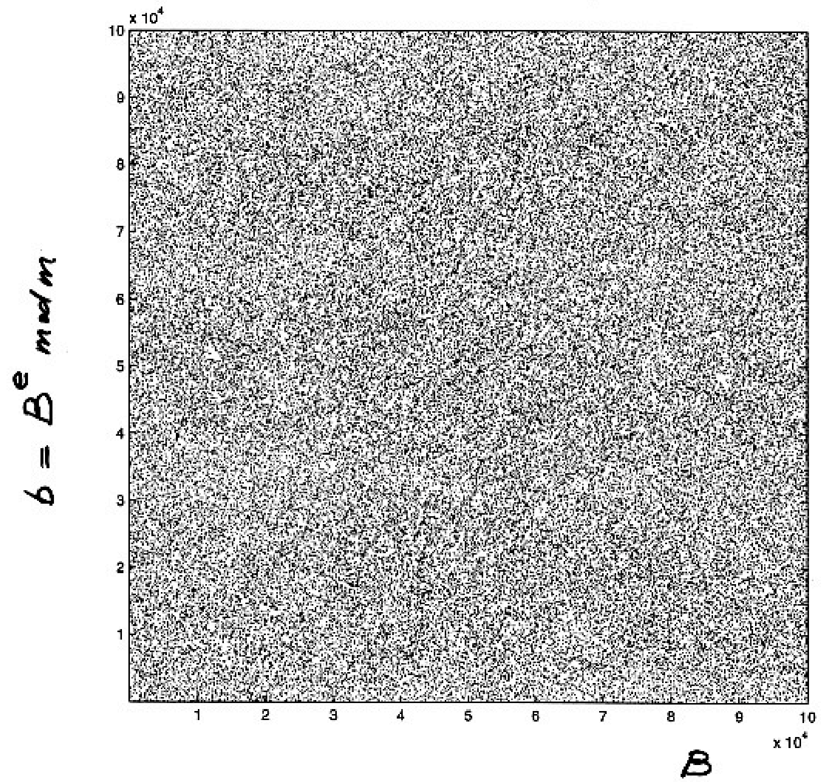
$B^4 \bmod m = 1026^2 \bmod m = 575$

$\vdots \qquad\qquad \vdots \qquad\qquad \vdots$

$B^{16} \bmod m = 985^2 \bmod m = 534$

$B^e \bmod m = (821 \cdot 534) \bmod m = 1641$

Die Codes b aller Botschaften B



$m = 100\,007$, $e = 53$

B_j: 0821 0520 0520 0005 2103 0800 0113 0013 1518 0721 1820 0514

b_j: 1641 1939 1939 1661 2699 1074 0720 0480 0165 1870 0420 2320

codierte Botschaft

3. Decodierung

Grundlage: Satz von Fermat ("kleiner")

Bemerkung: (a,p) bedeutet "ggT" von a und p

z.B. $(78,54) = ?$

zwei Methoden:

(i) Zürcher Sekundarschule, 1950 A.D.

$$78 = \boxed{2\cdot 3}\cdot 13$$
$$54 = \boxed{2\cdot 3}\cdot 3\cdot 3$$
$$\Rightarrow (78,54) = 6$$

Faktorisierungen, schwierig für grosse Zahlen!

(ii) Euklid, 300 v. Chr.

"Kettendivision", viel raffinierter schnell auch für 300 D-Zahlen

$\frac{78}{54}$: $78 = 1\cdot 54 + 24$

$\frac{54}{24}$: $54 = 2\cdot 24 + 6$

$\frac{24}{6}$: $24 = 4\cdot \boxed{6} + 0$

$$(78,54) = 6$$

3. Der Satz von Fermat (1601-1665)

Einführung: Nochmals Schuldivision

$$\frac{1}{7} = 1.00000000\cdots : 7 = 0.14285714\cdots$$

30
20
60
40
50
10
30
2

Wiederholung

Es folgt: $999999:7$ "geht auf"

d.h. $7 \mid 10^6 - 1$ oder $10^{7-1} \equiv 1 \bmod 7$

3.1. Satz und Beweis

SATZ (Fermat). Sei p prim und sei a kein Vielfaches von p, d.h. $(a,p) = 1$. Dann gilt

$$\boxed{a^{p-1} \equiv 1 \bmod p.}$$

Beweis: Vollständiges Restsystem (ohne Null), $1, 2, 3, \cdots, p-1 \bmod p$ bekommt man auch als

$1\cdot a, 2\cdot a, 3\cdot a, \cdots, (p-1)a \bmod p$ mit $a \neq kp$

(in permutierter Reihenfolge)

Betrachte das Produkt aller Repräs.:

(35

$$\underbrace{(p-1)!}_{\not\equiv 0 \bmod p} \equiv (p-1)!\; a^{p-1} \bmod p$$

$$\Rightarrow \quad a^{p-1} \equiv 1 \bmod p \qquad \text{QED}$$

Bemerkung: Die Umkehrung gilt nicht:

Aus $a^{m-1} \equiv 1 \bmod m$ folgt nicht $m = \text{prim}$.

Es gibt also zusammengesetzte Zahlen, die sich im Fermat-Satz wie Primz. verhalten.

Beispiel: $m = 15 = 3 \cdot 5$, $a = 4$

Dennoch: $a^{m-1} = 4^{14} = (2^7)^4 \equiv 8^4 \equiv 4^2 \equiv 1 \bmod 15$

3.2. Der Fermat-Test

Kontraposition (des Satzes v. Fermat)

(*)
$$\boxed{\text{Es existiere ein } a > 0 \text{ mit } a^{m-1} \not\equiv 1 \bmod m;\ \text{dann ist } m \text{ zusammengesetzt}}$$

- Die Überprüfung von (*) heisst Fermat-Test
- a heisst Zeuge (witness) für die Zusammengesetztheit von m
- Zahlen m, die den Fermat-Test nicht bestehen, heissen Fermat-Pseudoprimzahlen (bezüglich Basis a).

Ein gescheiterter Fermat-Test, $a^{m-1} \equiv 1 \bmod m$, gibt keine Information über "$m = \text{prim}$?"

(36

Beispiele

k	$12^k \bmod 65$	$2^k \bmod 65$
2	14	4
4	$196 \equiv 1$	16
8	1	$256 \equiv -4$
16	1	16
32	1	$256 \equiv -4$
64	1	$16 \not\equiv 1$

(i) $m = 65$

- $a = 12$ (z.B. würfeln): $a^{m-1} \equiv 1$ (m)

  Pech! Man weiss nichts. Neues a!
- $a = 2$: $a^{m-1} \not\equiv 1 \pmod m$

  $\Rightarrow$ m ist zusammengesetzt

  $a = 2$ ist Zeuge dafür

  - Man hat keine Information über die Faktoren von m!
  - Dafür ist der Test "billig", polynomial in Zahlenlänge

(ii) $m = 341$

- $a = 2$, $a^{m-1} \equiv 1 \bmod m$ (Seite 27)

  $\Rightarrow$ m ist Fermat-Pseudoprimzahl bez. 2
- $a = 3$: $3^5 \equiv -98$, $3^{10} \equiv 56$, $3^{15} \equiv -32$

  $3^{30} \equiv 1024 \equiv 1$

  $3^{340} = 3^{10} \cdot (3^{30})^{11} \equiv 56$

  $\Rightarrow$ m ist "entlarvt" als zusammengesetzt

Zusammenfassung

Der Satz von Fermat liefert einen schnellen Test auf Zusammengesetztheit.

Verallgemeinerung: Satz von EULER

Nochmals Schuldivision $\quad m = 21$

$$1.000\ldots : 21 = 0{,}047619'047619'047619'\ldots$$

160
130
40
190
100
160
...

Nur zu $m = 21$ teilerfremde Reste, nie $3, 6, 7, \ldots$!

Fermat-Test mit $a = 10$:

$$10^{20} \bmod 21 \equiv (-5)^{10} \equiv 4^5 \equiv 11^2 \equiv 16 \neq 1$$

$\Rightarrow$ 21 ist zusammengesetzt

Definition:

Eulersche φ-Funktion:

$\varphi(m) :=$ Anzahl der zu m teilerfremden Reste (aus $0, 1, 2, \ldots, m-1$)

Beispiele:
- $m = p =$ prim
  $\varphi(p) = p - 1$ $\quad$ einfach!
- $m = p \cdot q$, p, q prim
  $\varphi(p \cdot q) = (p-1)(q-1)$
  schwierig, braucht $m = p \cdot q$
- $\varphi(21) = 2 \cdot 6 = 12$ $\quad \frac{10^{12}-1}{21}$ "geht auf"

Verallgemeinerung des Satzes v. Fermat:

Betrachte die teilerfremden Restkl. mod m,

$$r_1 = 1,\ r_2,\ r_3,\ \ldots,\ r_{\varphi(m)}$$

sowie einen Faktor a mit $(a, m) = 1$.

Dann durchlaufen

$$a \cdot r_1,\ a \cdot r_2,\ \ldots,\ a \cdot r_{\varphi(m)}$$

wiederum alle teilerfremden Restklassen mod m, in permutierter Reihenfolge.

Damit folgt analog zu S. 34:

(b) SATZ von Euler

$$\boxed{\text{Falls } (a, m) = 1, \text{ gilt } a^{\varphi(m)} \equiv 1 \bmod m}$$

Satz von Fermat ist Spezialfall für m prim.

(c) Das inverse Element $a^{-1} \bmod m$

Sei $(a, m) = 1$

Gesucht: x mit $ax \equiv 1 \bmod m$

Wir schreiben $x = a^{-1} \bmod m$

Behauptung $\quad \boxed{x \equiv a^{\varphi(m)-1} \bmod m}$

Beweis: $a \cdot x \equiv a^{\varphi(m)} \equiv 1 \bmod m$ (nach Euler)

Decodierung. Mit den selben Waffen !

$b = B^e \bmod m$ Voraussetzg: $(e,m)=1$

Suche Decodierexponent D, so dass

$$\boxed{B = b^D \bmod m} \quad \forall\, b$$

Also: $B = (B^e)^D \bmod m = B^{e \cdot D} \bmod m$

Euler: $1 = B^{\varphi(m)} \bmod m$

Resultat: Wähle D so, dass $e \cdot D - k\,\varphi(m) = 1$ für ein gewisses k

$$\Rightarrow \boxed{e \cdot D = 1 \bmod \varphi(m)} \qquad D = e^{-1} \bmod \varphi(m)$$

Im Beispiel: $m = 2747 = 41 \cdot 67$ schwer zu finden!

$\varphi(m) = 40 \cdot 66 = 2640$

Bem: Für $m = 2749$ (prim) : $\varphi(m) = 2748$ dies kann jeder!

Eine kleine Rechnung:

$D = \dfrac{1 + k \cdot 2640}{17}$, k so, dass Division 'aufgeht'

modulo 17 rechnen: $1 + k \cdot 5 = 0 \bmod 17$

im Kopf: $k = 10$

$D = \dfrac{26401}{17} = 1553$ $\qquad \boxed{B = b^{1553} \bmod 2747}$

Bemerkung: D systematisch mit Euklid

$e = 17$, $\varphi(m) = 2640$; $D = e^{-1} \bmod m$?

1. Entwickle $\frac{\varphi(m)}{e}$ in Kettenbruch ($\equiv$ Euklid):

$$\frac{2640}{17} = 155 + \cfrac{1}{\frac{17}{5}} = 155 + \cfrac{1}{3 + \cfrac{1}{2 + \frac{1}{2}}}$$

Approximanten: $\left.\dfrac{p_k}{q_k}\right|_{k=0}^{n} = \left\{\dfrac{155}{1}, \dfrac{466}{3}, \dfrac{1087}{7}, \dfrac{2640}{17}\right\}$, $n=3$

2. $D = (-1)^n p_{n-1} = (-1)^3 \cdot 1087 = 1553 \bmod 2640$

4. Warum unknackbar ?

Wähle m genügend gross,

früher	512 bit	=	155 Dez.
ab ca 2000	1024 bit	=	309 Dez.

als Produkt von 2 ca. gleich langen Primzahle

Komplexität von Operationen mit Langzahlen:

Zahlen:	x, y
Logarithmus	$L := \log x$ (nat. Log.)
# Dezimalen	$N \approx 0.4343 \cdot L$

Operation	Aufwand
Schulmultiplikation	$c \cdot N^2$, $c > 0$
FFT-Multipl.	$c \cdot N \cdot \log N$, schneller f. $N \gtrsim 200$
Modulare Potenz	$c \cdot N^3$
Euklid	$c \cdot N^3$
Erkenn. von Composites	$c \cdot N^3$
Primzahlerkennung	$c \cdot N^\alpha$, $3 < \alpha < 4$
Primalitätsbeweis	$c \cdot N^\alpha$, $\alpha = 10.5$ oder $\alpha = 4$?

Erzeugung grosser Primzahlen.

Mittlerer Abstand von 2 Pz. bei x ist $\log x$ (Primzahlsatz)

z.B. bei $x = 10^{155}$: Mit $\log x \approx 357$ Versuchen (Verbesserter Fermat-Test = Miller-Rabin) hat man im Mittel eine prp (probable prime). Ev. durch Primalitätsbeweis erhärten!

Faktorisierung

Bis heute **kein** schneller Algorithm. bekannt
Woran liegt dies ??

- Obige Operationen betreffen Manipulation der N Ziffern von $x, y, \ldots$. Aufwand: $c N^\alpha$ **SCHNELL!**
- Faktorisierung scheint die Manipulation von Mengen mit fast x Elementen zu erfordern! **AUFWAND!**

z.B. Schulfaktorisierung (trial division)
Versuche Division $\frac{x}{2}, \frac{x}{3}, \frac{x}{5}, \frac{x}{7}, \ldots, \frac{x}{p}, \ldots$
für alle Primzahlen $p < \sqrt{x}$

Bester heute bekannter Algorithmus:
GNFS (General Number Field Sieve)

$$\boxed{\text{Aufwand} = c \cdot e^{\alpha (\log x)^{1/3} \cdot (\log \log x)^{2/3}}, \alpha \approx 1.4}$$

Extrapolation (gewagt)

155 Dez, $x = 10^{155}$ in 4 Monaten

$\Rightarrow$ 309 Dez in 26000 Jahren

Leider:
Bis heute kein Beweis bekannt, dass es <u>keinen</u> schnellen Faktorisierungs-Algorithmus geben kann.

5. Ergänzungen

Das quadratische Sieb, QS

Beispiel einer Methode, die grosse ganze Zahlen praktikabel faktorisiert

$\boxed{\text{Sei } m = 1261}$

Erster Schritt: Daten sammeln

(parallel, Monate mit 1000 Prozessoren)

Mit $x \approx \sqrt{m}$: viele Zahlen $x^2 - m$ (möglichst) vollständig faktorisieren

x	x^2-m	faktorisiert	take	
31	-300	$-2^2 \cdot 3 \cdot 5^2$	0	1
~~32~~	~~-237~~	~~$-3 \cdot 79$~~		
~~33~~	~~-172~~	~~$-2^2 \cdot 43$~~		
34	-105	$-3 \cdot 5 \cdot 7$	1	0
35	-36	$-2^2 \cdot 3^2$	1	1
36	35	$5 \cdot 7$	1	0
37	108	$2^2 \cdot 3^3$	1	1

Bemerkungen

(i) Wähle (genügend grosse) Faktorbasis F (= Teilmenge der vorkommenden Faktoren). Hier $F = \{-1, 2, 3, 5, 7\}$

(ii) Faktoren $p \in F$ kommen in je 2 "p-Rechen". Billig voraussagbar!

(iii) Eintrag der Faktoren mit Siebtechnik. Sehr schnell!

(iv) Nur die in F vollständigen Faktorisierungen sammeln

Zweiter Schritt

Suche Teilmenge der Sammlung, so dass

Produkt aller Faktoren = vollständ. Quadrat mod m

$$\boxed{\left(\prod_k x_k\right)^2 - y^2 = 0 \mod m}$$

$\Rightarrow$ take = 0 oder 1

Die Auswahl:

Lineares Gleichungssystem modulo 2 (nur 0, 1).
10^6 Gleichungen, 10^8 Einsen in Matrix. GAUSS-Elimination.
1 Woche auf "Number Cruncher".

Erstes Beispiel, $\quad take = \{0,1,1,1,1\}$

$$(\underbrace{34\cdot 35\cdot 36\cdot 37}_{3})^2 - (\underbrace{2^2\cdot 3^3\cdot 5\cdot 7}_{1258})^2 = 0 \text{ mod } 1261$$

Reduktion mod m

$$\Rightarrow (\underbrace{3+1258}_{0 \text{ mod } m})(3-1258) = 0 \quad \text{mod } m$$

Ausser Spesen
nichts gewesen

Zweites Beispiel, $\quad take = \{1,0,1,0,1\}$

$$(\underbrace{31\cdot 35\cdot 37}_{1054})^2 - (\underbrace{2^3\cdot 3^3\cdot 5}_{1080})^2 = 0 \quad \text{mod } 1261$$

$$(1054+1080)(1054-1080) = 2134\,(-26) = 0 \text{ mod } 1261$$

Aha!!

Zum Dessert ein ggT:

$$(1261, 26) = \ ?$$

Euklid:

$$1261 = 48\cdot 26 + 13$$
$$26 = 2\cdot 13 + 0$$

13 ist ein Faktor von 1261

$$\boxed{1261 = 13\cdot 97}$$

A Famous Factorization

```
Date: 27 April 1994
From: Massachusetts Institute of Technology (MIT)
Distribution: world
Subject: RSA-129

We are happy to announce that the 129-digit modulus

RSA_129 = 114381625757888867669235779976146612010218296721242\
          362562561842935706935245733897830597123563958705058\
          98907514759929002687954354 1

has been factored as  RSA_129 = p64 * p65,  where

p64   =    3490529510847650949147849619903898133417764638493\
          387843990820577

p65   =   32769132993266709549961988190834461413177642967992\
          942539798288533

The encoded message published was

b     =    9686961375462206147714092225435588290575999112457\
          431987469512093081629822514570835693147662288398962\
          8013391990551829945157815154

This number came from an RSA encryption of a 'secret' message,
using RSA_129 as the modulus and the public exponent e = 9007.
When encrypted with the 'secret' decoding exponent

D     =   1066986143685780244428687713289201547807099066339\
          378628012262244966310631259117744708733401685974623\
          06553968544513277109053606095

this becomes

B     =   200805001301070903002315180419000118050019172105011\
          309190800151919090618010705

Using the decoding scheme 01=A, 02=B, ..., 26=Z, and 00 for a
space between words, the decoded message reads

          THE MAGIC WORDS ARE SQUEAMISH OSSIFRAGE
```

To find the factorization of RSA_129, we used the double large-prime variation of the multiple-polynomial quadratic sieve factoring method. The sieving step took approximately 5000 mips years, and was carried out in 8 months by about 600 volunteers from more than 20 countries, on all continents except Antarctica. Combining the partial relations produced a sparse matrix of 569466 rows and 524338 columns. This matrix was reduced to a dense matrix of 188614 rows and 188160 columns using structured Gaussian elimination. Ordinary Gaussian elimination on this matrix, consisting of 35489610240 bits (4.13 gigabyte), took 45 hours on a 16K MasPar MP-1 massively parallel computer. The first three dependencies all turned out to be 'unlucky' and produced the trivial factor RSA_129. The fourth dependency produced the above factorization.

We would like to thank everyone who contributed their time and effort to this project. Without your help this would not have been possible.

Derek Atkins
Michael Graff
Arjen Lenstra
Paul Leyland

Remark:
The 129-digit modulus RSA_129, together with the encoded message, was originally published by Ronald L. Rivest, Adi Shamir and Leonard Adleman: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, Communications of the ACM, 21 (1978), p. 120-126. An eloquent summary is given by Martin Gardner in Scientific American, August 1977, p. 120-124. A prize of $ 100 was set out for the first person who decodes the message. Was it worth it?
For more information see, e.g., Wikipedia under RSA.

Factorizations of Special Numbers

(This is easier than the factorization of 'general' numbers)

Reported by Sam Wagstaff
Purdue University, West Lafayette, IN 47907

Newsletter August 19, 2004, page 94, line 4989
Done by: NFSNET, Method: Special Number Field Sieve (SNFS)

$$m = 10^{223} + 1 = 11 * 208729 * 1697477 * p10 * p73 * p129$$

```
p10  = 5156432569

p73  = 1966032824334071843648738236716922141567459089390 1\
       54109618864643162638011

p129 = 25309322317097404277807835737102129793277608545288\
       59196338893269457315069679673863428510642087796964\
       17921593338374442943683661653
```

Newsletter May 5, 1999, page 81, line 4342
Done by: The Cabal, Method: SNFS

$$m = (10^{211} - 1) / 9 = p93 * p118$$

```
m   =  11111111111111111111111111111111111111111111111111\
       11111111111111111111111111111111111111111111111111\
       11111111111111111111111111111111111111111111111111\
       11111111111111111111111111111111111111111111111111\
       11111111111

p93 =  69262455732438962066278232267733671113810848258828\
       1739734375570506492391931849524636731866879

p118 = 16042040371818984928424521776342331208254948956044\
       45254059369227570068074354992595031636365651567169\
       241873842145514809

phi(m) = (p93-1) * (p118-1)
    =  11111111111111111111111111111111111111111111111111\
       11111111111111111111111111111111111111111109506907\
       07392921261826865824085232066589599555272434317971\
       50307454325604484743787817039042389671520100197126\
       32233729424
```