Cryptanalysis Course
Part V – Factorization

Tanja Lange
Technische Universiteit Eindhoven

01 December 2016

with some slides by
Daniel J. Bernstein

How to find square product of congruences $(i - jm)(i - jr)$?

Start with congruences for, e.g., y^2 pairs (i, j).

Look for y-smooth congruences: y-smooth $i - jm$ and y-smooth $f_d$norm$(i - jr) =$ $f_d i^d + \cdots + f_0 j^d = j^d f(i/j)$. Norm covers all d roots r.
Here "y-smooth" means "has no prime divisor $> y$."

Find enough smooth congruences. Perform linear algebra on exponent vectors mod 2.

Polynomial selection

Many f's possible for n.
How to find f that
minimizes NFS time?

General strategy:
Enumerate many f's.
For each f, estimate time using
information about f arithmetic,
distribution of $j^{\deg f} f(i/j)$,
distribution of smooth numbers.

Let's restrict attention to $f(x) = (x - m)(f_5 x^5 + f_4 x^4 + \cdots + f_0)$.

Take m near $n^{1/6}$.
Expand n in base m:
$n = f_5 m^5 + f_4 m^4 + \cdots + f_0$.
Can use negative coefficients.

Have $f_5 \approx n^{1/6}$.
Typically all the f_i's
are on scale of $n^{1/6}$.

(1993 Buhler Lenstra Pomerance)

To reduce f values by factor B:

Enumerate many possibilities
for m near $B^{0.25}n^{1/6}$.

Have $f_5 \approx B^{-1.25}n^{1/6}$.
f_4, f_3, f_2, f_1, f_0 could be
as large as $B^{0.25}n^{1/6}$.
Hope that they are smaller,
on scale of $B^{-1.25}n^{1/6}$.

Conjecturally this happens
within roughly $B^{7.5}$ trials.
Then $(i - jm)(f_5 i^5 + \cdots + f_0 j^5)$
is on scale of $B^{-1}R^6 n^{2/6}$
for i, j on scale of R.
Several more ways; depends on n.

Asymptotic cost exponents

Number of bit operations
in number-field sieve,
with theorists' parameters,
is $L^{1.90\ldots+o(1)}$ where $L =$
$\exp((\log n)^{1/3}(\log\log n)^{2/3})$.

What are theorists' parameters?

Choose degree d with
$d/(\log n)^{1/3}(\log\log n)^{-1/3}$
$\in 1.40\ldots + o(1)$.

Choose integer $m \approx n^{1/d}$.
Write n as
$m^d + f_{d-1}m^{d-1} + \cdots + f_1 m + f_0$
with each f_k below $n^{(1+o(1))/d}$.
Choose f with some randomness
in case there are bad f's.

Test smoothness of $i - jm$
for all coprime pairs (i, j)
with $1 \le i, j \le L^{0.95\ldots+o(1)}$,
using primes $\le L^{0.95\ldots+o(1)}$.

$L^{1.90\ldots+o(1)}$ pairs.
Conjecturally $L^{1.65\ldots+o(1)}$
smooth values of $i - jm$.

Use $L^{0.12\ldots+o(1)}$ number fields.

For each (i, j)
with smooth $i - jm$,
test smoothness of $i - jr$
and $i - j\beta$ and so on,
using primes $\le L^{0.82\ldots+o(1)}$.

$L^{1.77\ldots+o(1)}$ tests.
Each $|j^d f(i/j)| \le m^{2.86\ldots+o(1)}$.
Conjecturally $L^{0.95\ldots+o(1)}$
smooth congruences.

$L^{0.95\ldots+o(1)}$ components
in the exponent vectors.

Three sizes of numbers here:

$(\log n)^{1/3}(\log\log n)^{2/3}$ bits:
y, i, j.

$(\log n)^{2/3}(\log\log n)^{1/3}$ bits:
m, $i - jm$, $j^d f(i/j)$.

$\log n$ bits: n.

Unavoidably 1/3 in exponent:
usual smoothness optimization
forces $(\log y)^2 \approx \log m$;
balancing norms with m
forces $d \log y \approx \log m$;
and $d \log m \approx \log n$.

Batch NFS

The number-field sieve used $L^{1.90\ldots+o(1)}$ bit operations finding smooth $i - jm$; only $L^{1.77\ldots+o(1)}$ bit operations finding smooth $j^d f(i/j)$.

Many n's can share one m; $L^{1.90\ldots+o(1)}$ bit operations to find squares for *all* n's.

Oops, linear algebra hurts; fix by reducing y.
But still end up factoring batch in much less time than factoring each n separately.

Asymptotic batch-NFS parameters:

$d/(\log n)^{1/3}(\log\log n)^{-1/3} \in 1.10\ldots + o(1)$.

Primes $\le L^{0.82\ldots+o(1)}$.

$1 \le i, j \le L^{1.00\ldots+o(1)}$.

Computation independent of n finds $L^{1.64\ldots+o(1)}$ smooth values $i - jm$.

$L^{1.64\ldots+o(1)}$ operations for each target n.

Batch NFS for RSA-3072

Expand n in base $m = 2^{384}$:
$n = n_7 m^7 + n_6 m^6 + \cdots + n_0$
with $0 \le n_0, n_1, \ldots, n_7 < m$.

Assume irreducibility of
$n_7 x^7 + n_6 x^6 + \cdots + n_0$.

Choose height $H = 2^{62}+2^{61}+2^{57}$:
consider pairs $(a, b) \in \mathbf{Z} \times \mathbf{Z}$ such
that $-H \le a \le H$, $0 < b \le H$,
and $\gcd\{a, b\} = 1$.

Choose smoothness bound
$y = 2^{66} + 2^{55}$.

There are about
$12H^2/\pi^2 \approx 2^{125.51}$
pairs (a, b).

Find all pairs (a, b) with
y-smooth $(a - bm)c$ where
$c = n_7a^7 + n_6a^6b + \cdots + n_0b^7$.

Combine these congruences
into a factorization of n,
if there are enough congruences.

Number of congruences needed
$\approx 2y/\log y \approx 2^{62.06}$.

Heuristic approximation:
$a - bm$ has same y-smoothness
chance as a uniform random
integer in $[1, Hm]$,
and this chance is u^{-u}
where $u = (\log(Hm))/\log y$.

Have $u \approx 6.707$
and $u^{-u} \approx 2^{-18.42}$,
so there are about
$2^{107.09}$ pairs (a, b)
such that $a - bm$ is smooth.

Heuristic approximation:
c has same y-smoothness chance
as a uniform random integer in
$[1, 8H^7m]$,
and this chance is v^{-v}
where $v = (\log(8H^7m))/\log y$.

Have $v \approx 12.395$
and $v^{-v} \approx 2^{-45.01}$,
so there are about
$2^{62.08}$ pairs (a, b) such that
$a - bm$ and c are both smooth.

Safely above $2^{62.06}$.

Biggest step in computation:
Check $2^{125.51}$ pairs (a, b)
to find the $2^{107.09}$ pairs
where $a - bm$ is smooth.

This step is independent of N,
reused by many integers N.

Biggest step in computation:
Check $2^{125.51}$ pairs (a, b)
to find the $2^{107.09}$ pairs
where $a - bm$ is smooth.

This step is independent of N,
reused by many integers N.

Biggest step depending on N:
Check $2^{107.09}$ pairs (a, b)
to see whether c is smooth.

This is much less
computation! . . . or is it?

The $2^{107.09}$ pairs (a, b)
are not consecutive,
so no easy way to sieve
for prime divisors of c.

The $2^{107.09}$ pairs (a, b)
are not consecutive,
so no easy way to sieve
for prime divisors of c.

Fix: factor each number
separately:
start with trial division,
then Pollard rho,
then Pollard $p - 1$,
then ECM.

The $2^{107.09}$ pairs (a, b)
are not consecutive,
so no easy way to sieve
for prime divisors of c.

Fix: factor each number
separately:
start with trial division,
then Pollard rho,
then Pollard $p - 1$,
then ECM.

Most of them covered in
http://facthacks.cr.yp.to/

The rho method

Define $\rho_0 = 0$, $\rho_{k+1} = \rho_k^2 + 11$.

Every prime $\le 2^{20}$ divides $S =$
$(\rho_1 - \rho_2)(\rho_2 - \rho_4)(\rho_3 - \rho_6)$
$\cdots(\rho_{3575} - \rho_{7150})$.
Also many larger primes.

Can compute $\gcd\{c, S\}$ using
$\approx 2^{14}$ multiplications mod c,
very little memory.

Compare to $\approx 2^{16}$ divisions
for trial division up to 2^{20}.

More generally: Choose z.
Compute $\gcd\{c, S\}$ where $S = (\rho_1 - \rho_2)(\rho_2 - \rho_4)\cdots(\rho_z - \rho_{2z})$.

How big does z have to be for all primes $\le y$ to divide S?

Plausible conjecture: $y^{1/2+o(1)}$; so $y^{1/2+o(1)}$ mults mod c.

Reason: Consider first collision in $\rho_1 \bmod p, \rho_2 \bmod p, \ldots$.
If $\rho_i \bmod p = \rho_j \bmod p$
then $\rho_k \bmod p = \rho_{2k} \bmod p$
for $k \in (j-i)\mathbf{Z} \cap [i, \infty] \cap [j, \infty]$.

The $p-1$ method

$S_1 = 2^{232792560} - 1$ has prime divisors
3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 53, 61, 67, 71, 73, 79, 89, 97, 103, 109, 113, 127, 131, 137, 151, 157, 181, 191, 199 etc.

These divisors include
70 of the 168 primes $\leq 10^3$;
156 of the 1229 primes $\leq 10^4$;
296 of the 9592 primes $\leq 10^5$;
470 of the 78498 primes $\leq 10^6$;
etc.

An odd prime p
divides $2^{232792560} - 1$
iff order of 2 in the
multiplicative group $\mathbf{F}_p^*$
divides $s = 232792560$.

Many ways for this to happen:
232792560 has 960 divisors.

Why so many?
Answer: $s = 232792560$
$= \mathrm{lcm}\{1, 2, 3, 4, 5, \ldots, 20\}$
$= 2^4 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 19$.

Can compute $2^{232792560} - 1$ using 41 ring operations. (Side note: 41 is not minimal.)

Ring operation: 0, 1, $+$, $-$, $\cdot$.

This computation: 1; $2 = 1 + 1$; $2^2 = 2 \cdot 2$; $2^3 = 2^2 \cdot 2$; $2^6 = 2^3 \cdot 2^3$; $2^{12} = 2^6 \cdot 2^6$; $2^{13} = 2^{12} \cdot 2$; 2^{26}; 2^{27}; 2^{54}; 2^{55}; 2^{110}; 2^{111}; 2^{222}; 2^{444}; 2^{888}; 2^{1776}; 2^{3552}; 2^{7104}; 2^{14208}; 2^{28416}; 2^{28417}; 2^{56834}; 2^{113668}; 2^{227336}; 2^{454672}; 2^{909344}; 2^{909345}; $2^{1818690}$; $2^{1818691}$; $2^{3637382}$; $2^{3637383}$; $2^{7274766}$; $2^{7274767}$; $2^{14549534}$; $2^{14549535}$; $2^{29099070}$; $2^{58198140}$; $2^{116396280}$; $2^{232792560}$; $2^{232792560} - 1$.

Given positive integer n,
can compute $2^{232792560} - 1 \bmod n$
using 41 operations in $\mathbf{Z}/n$.
Notation: $a \bmod b = a - b\lfloor a/b \rfloor$.

e.g. $n = 8597231219$: ...

$$\begin{aligned}
2^{27} \bmod n &= 134217728; \\
2^{54} \bmod n &= 134217728^2 \bmod n \\
&= 935663516; \\
2^{55} \bmod n &= 1871327032; \\
2^{110} \bmod n &= 1871327032^2 \bmod n \\
&= 1458876811; \ldots; \\
2^{232792560} - 1 \bmod n &= 5626089344.
\end{aligned}$$

Given positive integer n,
can compute $2^{232792560} - 1 \bmod n$
using 41 operations in $\mathbf{Z}/n$.
Notation: $a \bmod b = a - b \lfloor a/b \rfloor$.

e.g. $n = 8597231219$: ...

$$\begin{aligned}
2^{27} \bmod n &= 134217728; \\
2^{54} \bmod n &= 134217728^2 \bmod n \\
&= 935663516; \\
2^{55} \bmod n &= 1871327032; \\
2^{110} \bmod n &= 1871327032^2 \bmod n \\
&= 1458876811; \ldots; \\
2^{232792560}-1 \bmod n &= 5626089344.
\end{aligned}$$

Easy extra computation (Euclid):
$\gcd\{5626089344, n\} = 991$.

This $p - 1$ method (1974 Pollard) quickly factored $n = 8597231219$. Main work: 27 squarings mod n.

Could instead have checked n's divisibility by $2, 3, 5, \ldots$. The 167th trial division would have found divisor 991.

Not clear which method is better. Dividing by small p is faster than squaring mod n. The $p - 1$ method finds only 70 of the primes ≤ 1000; trial division finds all 168 primes.

Scale up to larger exponent

$s = \text{lcm}\{1, 2, 3, 4, 5, \ldots, 100\}$:

using 136 squarings mod n

find 2317 of the primes $\le 10^5$.

Is a squaring mod n

faster than 17 trial divisions?

Or

$s = \text{lcm}\{1, 2, 3, 4, 5, \ldots, 1000\}$:

using 1438 squarings mod n

find 180121 of the primes $\le 10^7$.

Is a squaring mod n

faster than 125 trial divisions?

Extra benefit:

no need to store the primes.

Plausible conjecture: if K is $\exp\sqrt{\left(\frac{1}{2}+o(1)\right)\log H\log\log H}$ then $p-1$ divides $\operatorname{lcm}\{1,2,\ldots,K\}$ for $H/K^{1+o(1)}$ primes $p \le H$.
Same if $p-1$ is replaced by order of 2 in $\mathbf{F}_p^*$.

So uniform random prime $p \le H$ divides $2^{\operatorname{lcm}\{1,2,\ldots,K\}}-1$ with probability $1/K^{1+o(1)}$.

$(1.4\ldots+o(1))K$ squarings mod n produce $2^{\operatorname{lcm}\{1,2,\ldots,K\}}-1 \bmod n$.

Similar time spent on trial division finds far fewer primes for large H.

Safe primes

This means numbers are easy to factor if their factors p_i have smooth $p_i - 1$.

To construct hard instances avoid such factors – that's it?

ANSI does recommend using "safe primes", i.e., primes of the form $2p' + 1$ when generating RSA moduli.

This does not help against the NFS nor against the following algorithms.

The $p+1$ factorization method

(1982 Williams)

Define $(X, Y) \in \mathbf{Q} \times \mathbf{Q}$ as the 232792560th multiple of $(3/5, 4/5)$ in the group Clock($\mathbf{Q}$).

The integer $S_2 = 5^{232792560} X$
is divisible by
82 of the primes $\le 10^3$;
223 of the primes $\le 10^4$;
455 of the primes $\le 10^5$;
720 of the primes $\le 10^6$;
etc.

Given an integer n,
compute $5^{232792560}X \bmod n$
and compute gcd with n,
hoping to factor n.

Many p's not found by $\mathbf{F}_p^*$
are found by Clock($\mathbf{F}_p$).

If -1 is not a square mod p
and $p+1$ divides 232792560
then $5^{232792560}X \bmod p = 0$.

Proof: $p \equiv 3 \pmod 4$, so
$(4/5 + 3i/5)^p = 4/5 - 3i/5$ and
so $(p+1)(3/5, 4/5) = (0, 1)$
in the group Clock($\mathbf{F}_p$)
so $232792560(3/5, 4/5) = (0, 1)$.

The elliptic-curve method

Stage 1: Point P on E over $\mathbf{Z}/n$,
compute $R = sP$ for
$s = \text{lcm}\{2, 3, \ldots, B_1\}$.

Stage 2: Small primes
$B_1 < q_1, \ldots, q_k \le B_2$
compute $R_i = q_i R$.

If order of P on $E/\mathbf{F}_{p_i}$
(same curve, reduce mod p_i)
divides sq_i, then
$R_i = (0, 1)$ (using Edwards).

Compute $\gcd\{n, \prod y(R_i)\}$.

Good news (for the attacker):
All primes $\le H$ found after reasonable number of curves.

Order of elliptic-curve group $\in [p+1-2\sqrt{p}, p+1+2\sqrt{p}]$.
If a curve fails, try another.

Plausible conjecture: if B_1 is $\exp\sqrt{\left(\frac{1}{2}+o(1)\right)\log H\log\log H}$ then, for each prime $p \le H$, a uniform random curve mod p has chance $\ge 1/B_1^{1+o(1)}$ to find p.

Find p using, $\le B_1^{1+o(1)}$ curves; $\le B_1^{2+o(1)}$ squarings.
Time subexponential in H.

Bad RSA randomness

2004 Bauer–Laurie:
checked 18000 PGP RSA keys;
found 2 keys sharing a factor.

2012.02.14 Lenstra–Hughes–
Augier–Bos–Kleinjung–Wachter
"Ron was wrong, Whit is right"
(Crypto 2012): checked $7 \cdot 10^6$
SSL/PGP RSA keys; found $6 \cdot 10^6$
distinct keys; factored 12720 of
those,
thanks to shared prime factors.

2012.02.17 Heninger–Durumeric–Wustrow–Halderman announcement (USENIX Security 2012):
checked $>10^7$ SSL/SSH RSA keys; factored 24816 SSL keys, 2422 SSH host keys.

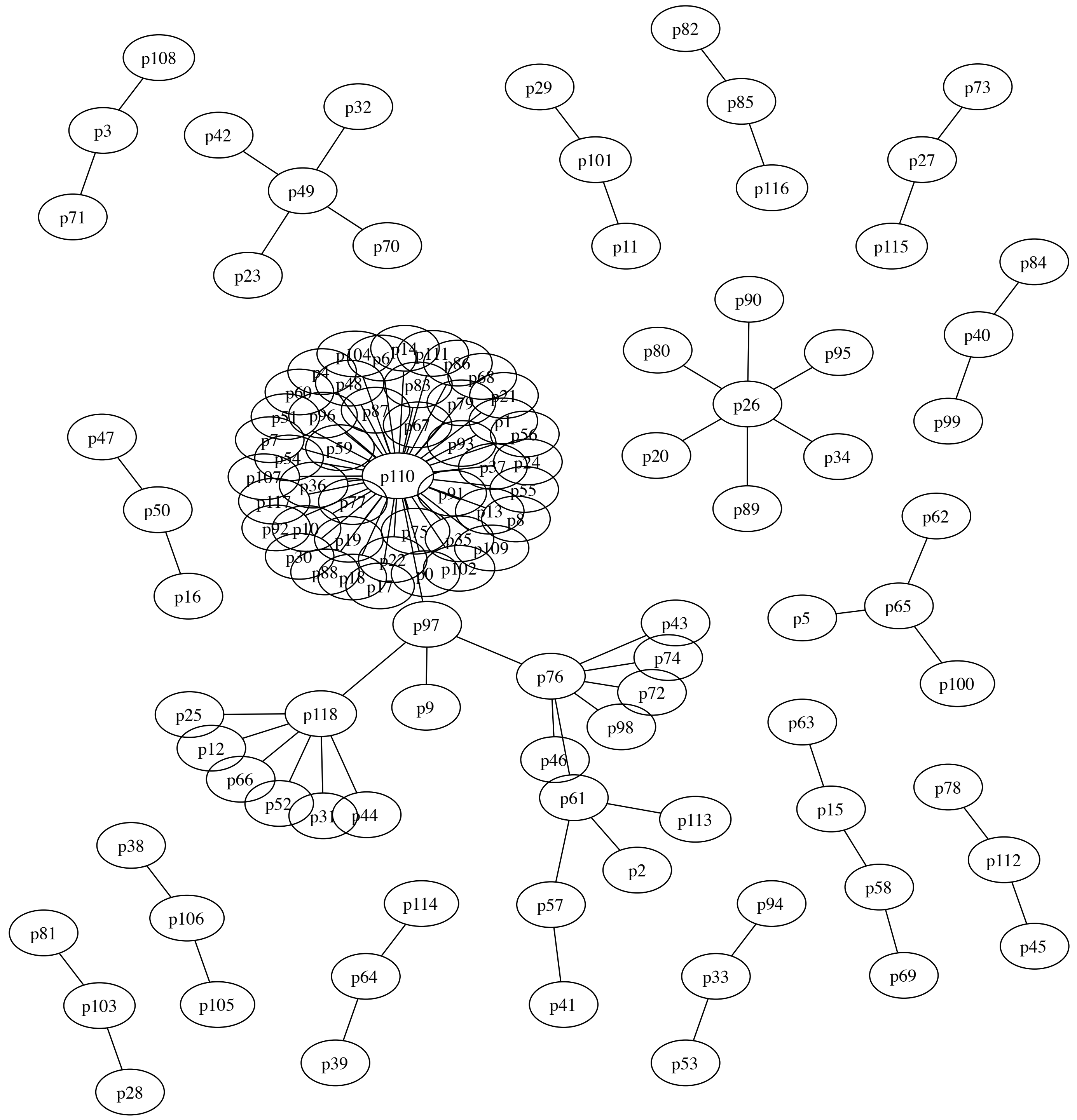
“Almost all of the vulnerable keys were generated by and are used to secure embedded hardware devices such as routers and firewalls, not to secure popular web sites such as your bank or email provider.”

These computations find q_2 in
$p_1q_1, p_2q_2, p_3q_3,$
$p_4q_2, p_5q_5, p_6q_6;$
and thus also p_2 and p_4.
Obvious:GCD computation.
Faster: scaled remainder trees.

Nice follow-up project:
Do this with Taiwan citizen cards.
Online data base of RSA keys.

These were generated on
certified smart cards;
should have good randomness.
But: student broke 103 keys.

Closer look at the 119 primes



Prime p110 appears 46 times

```
c0000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
000000000000000000000000000002f9
```

Prime p110 appears 46 times

```
c0000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
000000000000000000000000000002f9
```

which is the next prime after $2^{511} + 2^{510}$.

Prime p110 appears 46 times

```
c0000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
000000000000000000000000000002f9
```

which is the next prime after $2^{511} + 2^{510}$.

Next up

```
c9242492249292499249492449242492
24929249924949244924249224929249
92494924492424922492924992494924
49242492249292499249492449242 4e5
```

Several other factors exhibit such a pattern.

Prime generation

Choose a bit pattern of length 1, 3, 5, or 7 bits, repeat it to cover more than 512 bits, and truncate to exactly 512 bits.

Prime generation

Choose a bit pattern of length 1, 3, 5, or 7 bits, repeat it to cover more than 512 bits, and truncate to exactly 512 bits.
For every 32-bit word, swap the lower and upper 16 bits.

Prime generation

Choose a bit pattern of length 1, 3, 5, or 7 bits, repeat it to cover more than 512 bits, and truncate to exactly 512 bits.
For every 32-bit word, swap the lower and upper 16 bits.
Fix the most significant two bits to 11.

Prime generation

Choose a bit pattern of length 1, 3, 5, or 7 bits, repeat it to cover more than 512 bits, and truncate to exactly 512 bits.
For every 32-bit word, swap the lower and upper 16 bits.
Fix the most significant two bits to 11.
Find the next prime greater than or equal to this number.

Factoring by trial division

Choose a bit pattern of length 1, 3, 5, or 7 bits, repeat it to cover more than 512 bits, and truncate to exactly 512 bits.
For every 32-bit word, swap the lower and upper 16 bits.
Fix the most significant two bits to 11.
Find the next prime greater than or equal to this number.

Factoring by trial division

Choose a bit pattern of length 1, 3, 5, or 7 bits, repeat it to cover more than 512 bits, and truncate to exactly 512 bits.
For every 32-bit word, swap the lower and upper 16 bits.
Fix the most significant two bits to 11.
Find the next prime greater than or equal to this number.

Do this for any pattern:
0,1,001,010,011,100,101,110
00001,00010,00011,00100,00101,. . .

Computing GCDs factored 105 moduli, of which 18 were new.

Computing GCDs factored 105 moduli, of which 18 were new.

Breaking RSA-1024
by “trial division”.
Factored 4 more keys using patterns of length 9.

Computing GCDs factored 105 moduli, of which 18 were new.

Breaking RSA-1024
by "trial division".
Factored 4 more keys using patterns of length 9.

More factors by studying other keys and using lattices.
"Factoring RSA keys from certified smart cards: Coppersmith in the wild"
(with D.J. Bernstein, Y.-A. Chang, C.-M. Cheng, L.-P. Chou, N. Heninger, N. van Someren)
http://smartfacts.cr.yp.to/