Coppersmith’s factorization factory

Tanja Lange

Coding Theory and Cryptology

EIPSI

Department of Mathematics and Computer Science

tanja@hyperelliptic.org

22.10.2009

Joint work with Daniel J. Bernstein (UIC), Chen-Mou Cheng (NTU), and Bo-Yin Yang (Academia Sinica)

Encryption

Important Distinction

- Partners share a secret (key)
  ⇒ use of *symmetric encryption* methods
  - classical methods (Caesar, . . .) belong to this class
  - One-time pad
  - AES
- Partners have no initial secret
  ⇒ use of public key cryptography
  - RSA
  - Diffie-Hellman key-exchange
  - ElGamal signatures

Most applications are hybrid, i.e., they use a combination of both (e.g. use public key cryptography to get session key).

Public Key cryptography

Each user has two keys

- a private key chosen by him which *never* leaves the secure environment,
- a public key which is made public in a kind of phone book.
- These keys are linked by a (computational) one-way function; the general principles are public.
- Sender uses public key of recipient to encrypt
  $\Rightarrow$ only recipient knows secret key and can decrypt.
- Sender uses his private key to sign message
  $\Rightarrow$ signature must be due to him as only he knows the secret; every recipient can verify the signature using the sender's public key.

What do mathematicians do here?

- Find good candidates for computational one-way functions.
- Look for variants having better performance.
- Analysis of attacks.

Proposed candidates

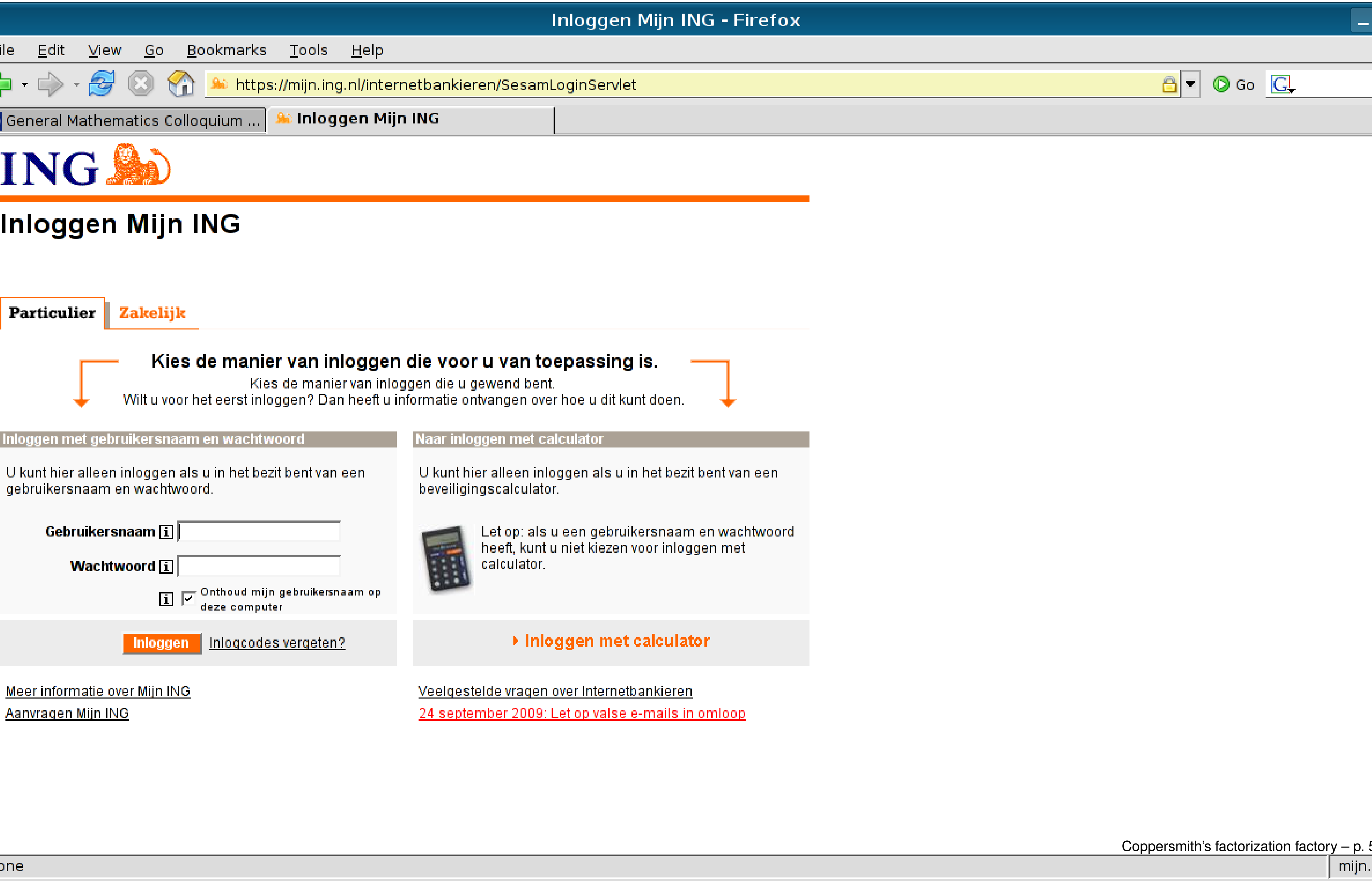
- Multiplying two integers is easy while factoring is hard (RSA).

$$\text{primes } p, q \Rightarrow N = p \cdot q \longleftrightarrow \text{ integer} N \Rightarrow p, q.$$

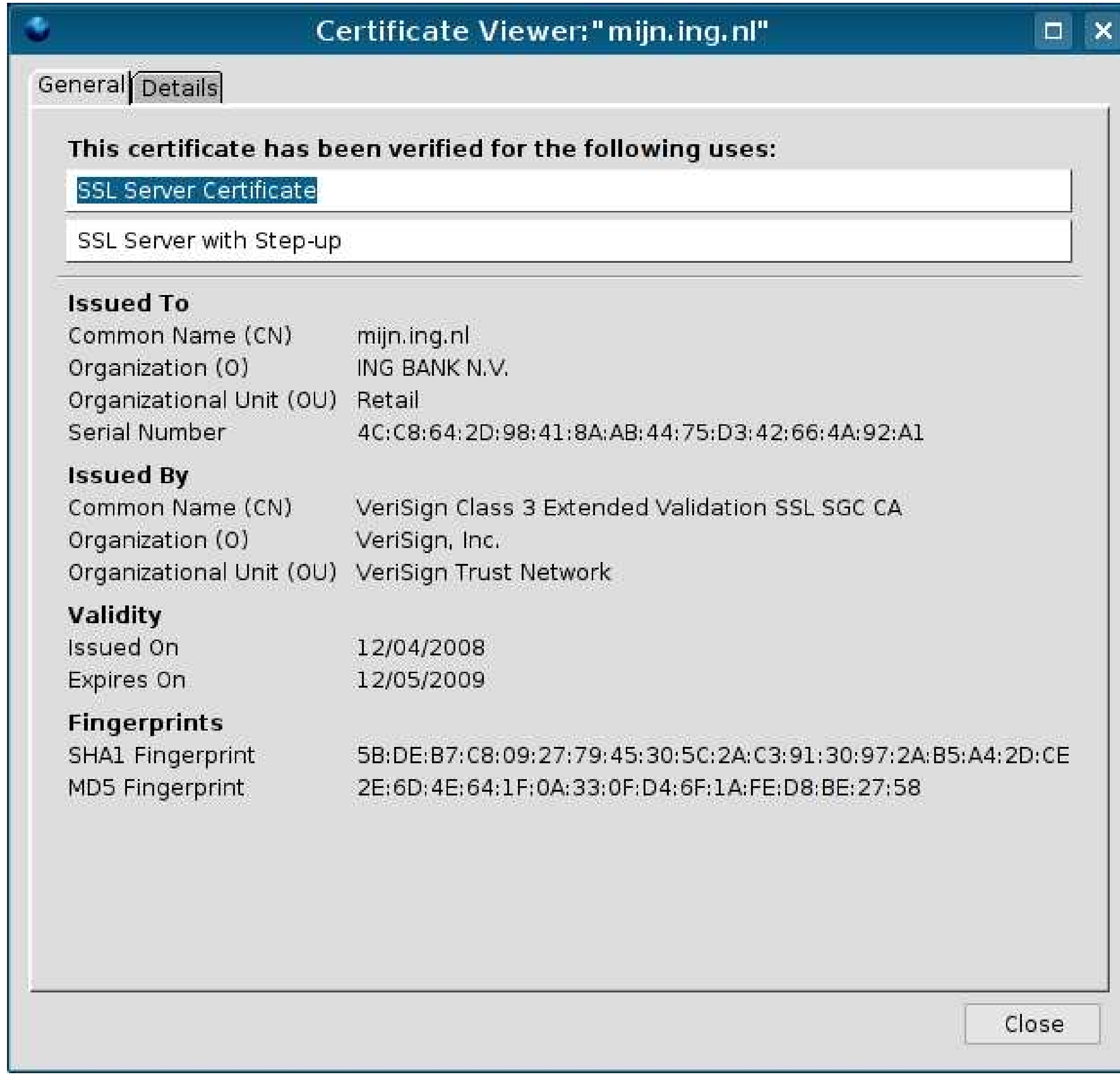
- In some groups computing scalar multiples is easy while finding the corresponding scalar given two group elements is hard $\Rightarrow$ discrete logarithm problem (DLP).

$$P \in G, \text{ integer } a \Rightarrow Q = [a]P \longleftrightarrow Q, P \in G \Rightarrow a.$$

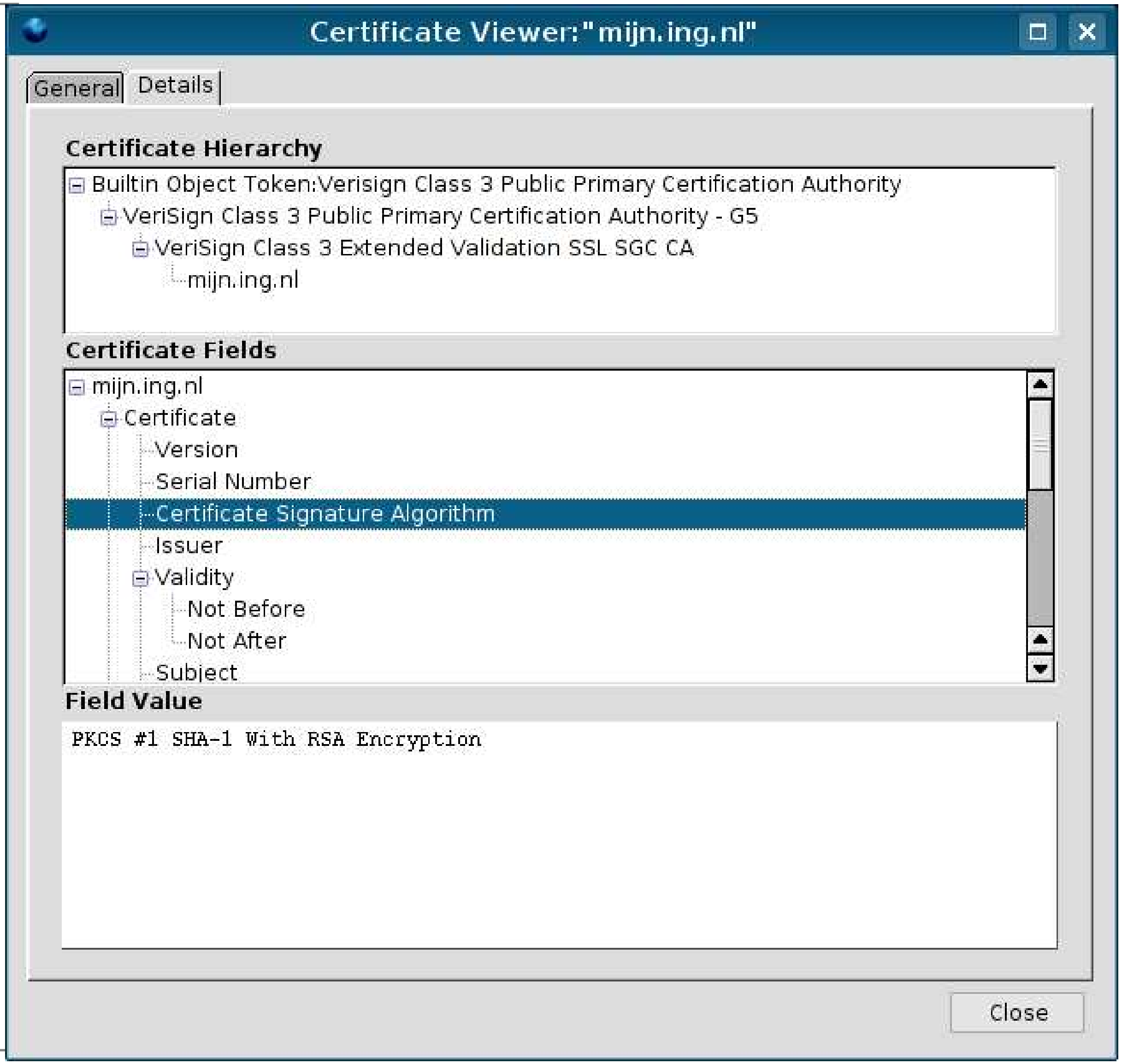
Real-world use I



Real-world use II



Real-world use III



Subject's Public Key

```
30 81 89 02 81 81 00 cc d1 35 c6 8f f6 d9 e9 1f
0b a6 a8 b7 53 4d 8d 21 69 11 16 8e 50 2c c2 34
f4 9e 9e 41 d7 74 47 20 a8 71 0e a1 58 31 67 87
5e d5 31 9b 44 dd 12 74 53 8b 34 c1 d9 a7 b0 5e
27 b5 e8 d1 9c c3 7c 73 bd 6a 61 e4 75 7a e2 fd
6f 7c ca e4 48 05 97 e0 1f 4a 99 9e 29 53 78 fd
65 6d 77 78 9f 9e 63 fc 05 71 18 90 a4 c2 19 97
73 e3 17 a8 ff 15 cd 0a 68 d8 43 aa 6c c5 b3 75
58 31 42 d9 e5 8b 5f 02 03 01 00 01
```

Contains 1024 bit modulus.

RSA – schoolbook version!

RSA Cryptosystem I

Secret computation – key generation:

- pick two primes p, q,
- compute $N = p \cdot q$,
- compute $\varphi(N) = (p-1)(q-1)$, where φ is the Euler-φ function,
- pick integer e co-prime to $\varphi(N)$,
- compute d with $de \equiv 1 \bmod \varphi(N)$.

Note that for any m with $\gcd(m, N) = 1$ the order of m modulo N is $\varphi(N)$.
So we have

$$(m^e)^d \equiv m^{ed} \equiv m^{k\varphi(N)+1} \equiv m \bmod N,$$

for some integer k.

RSA Cryptosystem II

- secret key d,
- public key (e, N).
- All intermediate values of the key generation should be deleted.
- Encryption of message m (school-book version!):
  - compute $c \equiv m^e \bmod N$,
  - send ciphertext c.
- Decryption of ciphertext c:
  - compute $c^d \equiv (m^e)^d \bmod N$,
  - obtain cleartext m.

Signature on message m is $m^d \bmod N$. Signing requires knowledge of secret key d.

Parameters & common pitfalls

- Factorization hardest if p and q have same size.
- Current choice: N has 1024 bits and p and q have 512 bits.
- 1024 bits still feel secure, agencies recommend it for documents needing no more than 5 years of security
- Better security 2048 bits and not too painful performance (if one really wants to stick to RSA ...).
- Problem if one takes 'same size' too literally:

$$
\begin{aligned}
N = {} & 1000000000000000000000000000000000000000000000000000000\\
& 0000000000000000000000000000000000000000000000000000000000\\
& 0000000000000000000000000000000000000000000000000000000029\\
& 9999999999999999999999999999999999999999999999999999999999\\
& 9999999999999999999999999999999999999999999999999999999999\\
& 99999999999999999999999999999999999999999999999999997921.
\end{aligned}
$$

Pitfalls ctd.

- Yes, this looks like very close to a power of 10, actually close to 10^{340}. Squareroot $\sqrt{N}$ is almost an integer, almost 10^{170}.

Pitfalls ctd.

- Yes, this looks like very close to a power of 10, actually close to 10^{340}. Squareroot $\sqrt{N}$ is almost an integer, almost 10^{170}.
- Brute-force search $\gcd(N, 10^{170} - i)$ finds factor $q = 10^{170} - 33$ and then $p = N/q = 10^{170} + 63$.

Pitfalls ctd.

- Yes, this looks like very close to a power of 10, actually close to 10^{340}. Squareroot $\sqrt{N}$ is almost an integer, almost 10^{170}.
- Brute-force search $\gcd(N, 10^{170} - i)$ finds factor $q = 10^{170} - 33$ and then $p = N/q = 10^{170} + 63$.
- This attack is well known and mentioned e.g. in the Handbook of Applied Cryptography. Yet it is found in practice by amateurs, usually with powers of 2 rather than 10.
- Famous example: letter bomber Franz Fuchs in Austria (6 attack series between 1993 and capture in 1997) send RSA-encrypted letter to Austrian magazine 'Profil'. Modulus N had 1024 bits, i.e. full size, but both primes were extremely close to 2^{512}. Found by Hans Dobbertin.

Attacks against generic RSA

Best known attacks against generic RSA try to find a, b with

$$a^2 \equiv b^2 \bmod N, \text{ i.e., } N | (a^2 - b^2). \tag{1}$$

If $N \nmid (a \pm b)$ then

$$1 < \gcd(N, a - b) < N$$

finds nontrivial factor.
The quadratic sieve finds such a relation by computing for many integers a_i

$$(a_i^2 \bmod N) = p_1^{e_1} p_2^{e_2} \cdots p_r^{e_r} a_i', \tag{2}$$

where p_j in some factor base F and $\gcd(a_i', p_j) = 1$ for all $p_j \in F$. Collect relations of form (2) with $a_i' = 1$; construct relation (1) using linear algebra.

Number field sieve

- Better method to find relations, but relations live in a number field.
- Replace a_1^2 in the quadratic sieve by

$$(a_i - b_i m)(a_i - b_i \alpha), \text{ where } m \equiv \alpha \bmod N$$

  and α is an algebraic number (root of an irreducible polynomial over $\mathbb{Z}$).
- Relations have smaller height, requiring less time for the cofactorization.
- Final step is again solving a big linear system of equations over $\mathbb{Z}/2$; factorbasis consists of prime ideals of the ring of integers of $\mathbb{Q}(\alpha)$.

Number field sieve for RSA-2048

- Expand N in base $m = 2^{256}$:

$$N = n_7 m^7 + n_6 m^6 + \cdots + n_0$$

  with $0 \le n_0, n_1, \ldots, n_7 < m$.
- Choose height H of a and b as $H = 2^{55}$; i.e. consider all pairs $(a, b) \in \mathbb{Z} \times \mathbb{Z}$ such that $-H \le a \le H$, $0 < b \le H$, and $\gcd(a, b) = 1$.
- Choose smoothness bound $y = 2^{50}$.
- Write $c = n_7 a^7 + n_6 a^6 b + \cdots + n_0 b^7$
- Find all pairs (a, b) such that $a - bm$ and c are y-smooth.
- Smoothnes of c yields factorization of the ideal $a - b\alpha$.
- Collect smooth relations, combine them to factor N.

Analysis I

- There are about $12H^2/\pi^2 \approx 2^{110.28}$ such pairs (a, b).
- Heuristic approximation: $a - bm$ has same y-smoothness chance as a uniform random integer in $[1, Hm]$, and this chance is $u_1^{-u_1}$ where $u_1 = \frac{\log(Hm)}{\log y}$.
- Have $u_1 = (256 + 55)/50 = 6.22$ and $u_1^{-u_1} \approx 2^{-16.40}$, so there are about $2^{93.88}$ 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 $u_2^{-u_2}$ where $u_2 = \frac{\log(8H^7m)}{\log y}$.
- Have $u_2 = (3 + 7 \cdot 55 + 256)/50 = 12.88$ and $u_2^{-u_2} \approx 2^{-47.49}$, so there are about $2^{46.39}$ pairs (a, b) such that $a - bm$ and c are both smooth.

Analysis II

- Have $u_2 = (3 + 7 \cdot 55 + 256)/50 = 12.88$ and ${u_2}^{-u_2} \approx 2^{-47.49}$, so there are about $2^{46.39}$ pairs (a, b) such that $a - bm$ and c are both smooth.
- Number of primes and prime ideals up to y is $2y/\log y \approx 2^{45.88}$. The number of pairs (a, b) exceeds this; can algebraically combine these pairs into a congruence of squares modulo N.
- If still no solution, can add a few more congruences

Coppersmith's observation

- Biggest step in computation: Check $2^{110.28}$ pairs (a, b) to find the $2^{93.88}$ pairs (a, b) where $a - bm$ is smooth.
- But this step is independent of N. What if we want to factor many integers N?
- Biggest step depending on N: Check $2^{93.88}$ pairs (a, b) to see whether $c = n_7 a^7 + n_6 a^6 b + \cdots + n_0 b^7$ is smooth.
- This is much less computation!...or is it?

Sieving

- Dividing $a - bm$ by all primes $\leq y$ means $y/\log y \approx 2^{44.88}$ divisions. Much less arithmetic to fix b and sieve $a - bm$: mark every 3rd $a - bm$ as being divisible by 3, every 5th $a - bm$ as being divisible by 5, etc.
- Can also sieve polynomial values $n_7 a^7 + n_6 a^6 b + \cdots + n_0 b^7$ for consecutive integers a. So we can quickly sieve all $2^{110.28}$ values $n_7 a^7 + n_6 a^6 b + \cdots + n_0 b^7$.
- But this expands the $2^{93.88}$ values of c to $2^{110.28}$ values. Is there a better way to check smoothness of the $2^{93.88}$ values of c?

Sieving

- Dividing $a - bm$ by all primes $\le y$ means $y/\log y \approx 2^{44.88}$ divisions. Much less arithmetic to fix b and sieve $a - bm$: mark every 3rd $a - bm$ as being divisible by 3, every 5th $a - bm$ as being divisible by 5, etc.
- Can also sieve polynomial values $n_7a^7 + n_6a^6b + \cdots + n_0b^7$ for consecutive integers a. So we can quickly sieve all $2^{110.28}$ values $n_7a^7 + n_6a^6b + \cdots + n_0b^7$.
- But this expands the $2^{93.88}$ values of c to $2^{110.28}$ values. Is there a better way to check smoothness of the $2^{93.88}$ values of c?
- Yes: ECM! (elliptic curve method of factorization)
- Even better: early-abort ECM!

Efficient ECM

- Best general purpose factoring algorithm.
- Recent speedups due to
  - better choice of elliptic curve representation (Edwards curves allow faster curve operations);
  - beter choice of curves (large $\mathbb{Q}$-rational torsion group gives higher chance of factoring);
  - better software implementation (`eecm.cr.yp.to`);
  - can efficiently use latest trend in high-performance computing – Graphics Processing Units (GPUs) can run many instances in parallel (many cofactorizations and many curves per number).
  - 1 billions modular multiplications/sec on one PC!
- Recent news: $\ll$ 1 million modular mults. to detect 2^{50} smoothness.

The END

The END

Thanks!