Factoring Algorithms and Other Attacks on the RSA

T-79.5501 Cryptology

Lecture 8
April 8, 2008

Kaisa Nyberg

The Pollard $p-1$ Algorithm

- Let B be a positive integer and p a factor of n. The Pollard $p-1$ algorithm works if all prime power divisors of $p-1$ are less than B.
- Set $a = 2$.
- For $j = 2, \ldots, B$ compute $a = a^j \bmod n$, that is, compute $a = 2^{B!} \bmod n$.
- Compute $d = \gcd(a-1, n)$.
- If $1 < d < n$, then return d; else return “failure”.
- The complexity of the algorithm is $\mathcal{O}(B \log B (\log n)^2 + (\log n)^3$.

Why It Works

- If $q < B$ for every prime power q that divides $p-1$, then $p-1$ divides $B!$.
- Since p divides n, it must be that $a \equiv 2^{B!} \pmod{p}$.
- Since $2^{p-1} \equiv 1 \pmod{p}$, it follows that $a \equiv 1 \pmod{p}$.
- Then p divides $a-1$ and therefore p divides d = gcd$(a-1, n)$.
- d is a non-trivial divisor of n unless $a = 1$.
- If $a = 1$ the algorithm can be repeated using some other value than 2 to initialize a.

Dixon's Random Squares

- Suppose that we can find integers x and y such that $x \not\equiv \pm y \pmod{n}$ and $x^2 \equiv y^2 \pmod{n}$.
- Then n divides neither $x - y$ nor $x + y$.
- Then gcd$(x - y, n)$ is a non-trivial divisor of n.
- For example, $10^2 \equiv 32^2 \pmod{77}$. It follows that gcd$(32 - 10, 77)$ is a non-trivial divisor of 77, which indeed holds.
- The algorithm uses a factor base $\mathcal{B}$ which is a set of small primes.
- Then generate several integers z such that the prime factors of $z^2 \bmod n$ are in the set $\mathcal{B}$.
- Find a subset $z_1, \ldots, z_s$ of these integers such that the total number of occurences of each prime factor in the squares of these numbers is even.
- Then $z_1^2 \times \ldots \times z_s^2$ is equivalent modulo n to a square of a product of numbers from $\mathcal{B}$.

Random Squares Example

- $n = 15770708441$ and $\mathcal{B} = \{2,3,5,7,11,13\}$.
- Select

$$\begin{aligned} z_1 &= 8340934156 \text{ , then } z_1^2 \equiv 3 \times 7 \pmod{n} \\ z_2 &= 12044942944 \text{ , then } z_2^2 \equiv 2 \times 7 \times 13 \pmod{n} \\ z_3 &= 2773700011 \text{ , then } z_3^2 \equiv 2 \times 3 \times 13 \pmod{n}. \end{aligned}$$

- $(z_1 z_2 z_3)^2 \equiv 9503435785^2 \equiv (2 \times 3 \times 7 \times 13)^2 = 546^2 \pmod{n}$.
- $\gcd(9503435785 - 546, 15770708441) = 115759$.
- Current state of factoring algorithms, see: LENSTRA Arjen, Update on Factoring “A kilobit special number field sieve factorization” at `http://wiki.uni.lu/esc/docs/A+kilobit+special+number+field+sieve+factorization.ppt`

Computing $\phi(n)$

- If we can compute $\phi(n)$, then one can factor n.
- Given $\phi(n)$ one can solve p from the system of equations

$$\begin{aligned} n &= pq \\ \phi(n) &= (p-1(q-1) \end{aligned}$$

- By substituting $q = n/p$ to the second equation, one gets

$$p^2 - (n - \phi(n) + 1)p + n = 0.$$

- The two solutions p of this quadratic equation are the factors of n.

The Private Exponent

- If we can compute the private exponent then we can factor n with at least probability 1/2. Repeating m times gives success probability $1-(1/2)^m$.
- *Las Vegas algorithm* is a randomized algorithm which may fail to give an answer, but if it gives an answer, the answer is correct.
- Given a, b and n, with $ab \equiv 1 \pmod{\phi(n)}$.
- The idea is to find a non-trivial square root of 1 modulo n.
- write $ab - 1 = 2^s r$, where r is odd.
- Choose w at random such that $1 \le w \le n-1$. Check that $\gcd(w,n) = 1$. (If not, a non-trivial factor of n has been found!)
- Compute $v = w^r \bmod n$. If $v = 1$, then return "failure".
- Else find $k \le s$ such that $v_0 = v^{2^{k-1}} \neq 1 (\bmod n)$ and $v_0^2 = v^{2^k} \equiv 1 (\bmod n)$.
- If $v_0 \equiv -1 \pmod{n}$, then return "failure".
- Else compute $d = \gcd(v_0 + 1, n)$. Return d, which is a non-trivial factor of n.

Wiener's Small Private Exponent Attack

- If $3a < \sqrt[4]{n}$, where $n = pq$ and $q < p < 2q$, then there is an efficient deterministic algorithm for computing a and the factorization of n.
- See separate power point slides.

The Rabin Cryptosystem

- Let $n = pq$, where p and q are distinct primes and p, $q \equiv 3 \pmod 4$.
- Let $\mathcal{P} = \mathcal{C} = \mathbb{Z}_n$ [a], and define $\mathcal{K} = \{n, p, q\}$.
- For $K = (n, p, q)$, define

$$\begin{aligned} e_K(x) &= x^2 \bmod n \text{ , and} \\ d_K(y) &= \sqrt{y} \bmod n. \end{aligned}$$

- The value n is the public key, while p and q comprise the private key.

[a] Testbook restricts plaintexts and ciphertexts to $\mathbb{Z}_n^*$

Security of the Rabin Cryptosystem

- **Theorem:** Decrypting in the Rabin Cryptosystem is as hard as factoring the modulus.
- Trivially, if factoring is easy then decrypting is easy. It remains to prove the converse.
- Assume we have an efficient algorithm $\mathcal{A}$ for computing decryptions in the Rabin Cryptosystem. Then $\mathcal{A}$ can be used as a basis of a Las Vegas algorithm for factoring the modulus. The failure probability of this algorithm is 1/2.
- Select $x \in \mathcal{P}$ and compute $y = x^2 \bmod n$.
- Give y to $\mathcal{A}$, which returns u which is one of the four possible square roots of y modulo n.
- If $u \neq \pm x (\bmod n)$ (the probability that this happens is equal to 1/2) then we can compute a nontrivial divisor of n as $\gcd(x+u, n)$ (or as $\gcd(x-u, n)$).

The Insecurity of the Rabin Cryptosystem

- The same proof shows that the Rabin Cryptosystem is completely insecure against Chosen Plaintext Attack.
- In the Chosen Plaintext Attack the attacker is assumed to have access to a Decryption Oracle.

Bleichenbacher’s Attack and OAEP

- Bleichenbacher’s attack against RSA with PKC#1 padding shows the importance of resistance against Chosen Ciphertext Attack (CCA).
- In the CCA the attacker has access to an oracle which gives some partial information about the plaintext.
- The Optimal Asymmetric Encryption Padding (OAEP) has been designed to provide “plaintext awareness”.