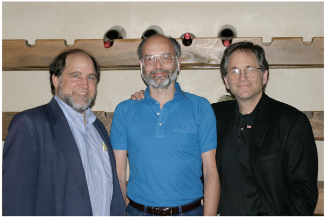
RSA Algorithm



Well-Known One-Way Functions

- Discrete Logarithm:
  Given p, g, and x, computing y in $y = g^x \pmod{p}$ is EASY
  Given p, g, y, computing x in $y = g^x \pmod{p}$ is HARD
- **Factoring**:
  Given p and q, computing n in $n = p \cdot q$ is EASY
  Given n, computing p or q in $n = p \cdot q$ is HARD
- Discrete Square Root:
  Given x and y, computing y in $y = x^2 \pmod{n}$ is EASY
  Given y and n, computing x in $y = x^2 \pmod{n}$ is HARD
- **Discrete eth Root**:
  Given x, n and e, computing y in $y = x^e \pmod{n}$ is EASY
  Given y, n and e, computing x in $y = x^e \pmod{n}$ is HARD

Rivest-Shamir-Adleman Algorithm

- Invented by three young faculty members at MIT: Ronald Rivest (CS), Adi Shamir (Math), and Leonard Adleman (Math) in the Summer and Fall of 1976:
  "Ron and Adi would come up with ideas, and Len would try to shoot them down. Len was consistently successful; late one night, though, Ron came up with an algorithm that Len couldnÕt crack."
- Following the ideas of building a public-key encryption method using trapdoor one-way functions of Merkle and Hellman
- It is based on the one-way functions factoring and discrete eth root
- The paper was published in 1977 (Comm. of ACM)
- The method was patented by MIT in 1983, which ended in 2000

Rivest-Shamir-Adleman Algorithm

- The User generates two large, approximately same size random primes: p and q
- The modulus n is the product of these two primes: $n = pq$
- Euler's totient function of n is given by $\phi(n) = (p-1)(q-1)$
- The User selects a number $1 < e < \phi(n)$ such that

  $$\gcd(e, \phi(n)) = 1$$

  and computes d with

  $$d = e^{-1} \pmod{\phi(n)}$$

  using the extended Euclidean algorithm

Rivest-Shamir-Adleman Algorithm

- e is the **public exponent** and d is the **private exponent**
- The **public key**: The modulus n and the public exponent e
- The **private key**: The private exponent d, the primes p and q, and $\phi(n) = (p-1)(q-1)$
- Encryption and decryption are performed by computing

$$\begin{aligned} C &= M^e \pmod{n} \\ M &= C^d \pmod{n} \end{aligned}$$

where M, C are the plaintext and ciphertext such that

$$0 \leq M, C < n$$

Correctness of RSA

- The correctness of the RSA algorithm follows from Euler's theorem: For n and a be positive, relatively prime integers, we have $a^{\phi(n)} = 1 \pmod{n}$
- Since we have $e \cdot d = 1 \bmod \phi(n)$, we can write $ed = 1 + K\phi(n)$ for some integer K, and thus

$$\begin{aligned} C^d &= (M^e)^d \pmod{n} \\ &= M^{ed} \pmod{n} \\ &= M^{1+K\phi(n)} \pmod{n} \\ &= M \cdot (M^{\phi(n)})^K \pmod{n} \\ &= M \cdot 1^K = M \pmod{n} \end{aligned}$$

provided that $\gcd(M, n) = 1$

Correctness of RSA

- On the other hand, if $\gcd(M, n) \neq 1$ and $M < n$, we have either $\gcd(M, n) = p$ or $\gcd(M, n) = q$
- Therefore, $M = u \cdot p$ or $M = v \cdot q$, for some integers $u < p$ and $v < q$, such that $\gcd(u, n) = 1$ and $\gcd(v, n) = 1$
- Without loss of generally, assume $M = u \cdot p$ with $\gcd(u, n) = 1$, we can write

$$C = M^e = (u \cdot p)^e = u^e \cdot p^e \pmod{n}$$

Since $\gcd(u, n) = 1$, we have $u^{ed} = u \pmod{n}$ and thus

$$\begin{aligned} C^d &= u^{ed} \cdot p^{ed} \pmod{n} \\ &= u \cdot p^{1+K\phi(n)} \pmod{n} \end{aligned}$$

- We will now show that $x = p^{1+K\phi(n)} \pmod{n}$ is equal to p, and thus $C^d = u \cdot p = M \pmod{n}$

Correctness of RSA

- Since $n = p \cdot q$, we write

$$x = p^{1+K\phi(n)} \pmod{p \cdot q}$$

- Due to the CRT, this implies

$$\begin{aligned} x_p &= p^{1+K\phi(n)} = 0 \pmod{p} \\ x_q &= p^{1+K\phi(n)} = p \pmod{q} \end{aligned}$$

- The first is true, because $p = 0 \pmod{p}$
- The second equality is true because

$$\phi(n) = 0 \pmod{q-1}$$

since $\phi(n) = (p-1)(q-1)$

Correctness of RSA

- Therefore, we find

$$x = \begin{cases} 0 & \pmod{p} \\ p & \pmod{q} \end{cases}$$

Applying the CRT to the residues $(0, p)$ and moduli (p, q), we obtain

$$x = 0 \cdot q \cdot c_1 + p \cdot p \cdot c_2 \pmod{pq}$$

such that $c_1 = q^{-1} \pmod{p}$ and $c_2 = p^{-1} \pmod{q}$

- We can write $p \cdot c_2 = 1 + L \cdot q$, and obtain x as

$$x = p \cdot p \cdot c_2 = p \cdot (1 + L \cdot q) = p + L \cdot pq \pmod{pq}$$

Thus, we find $x = p \pmod{n}$

Example RSA

- The primes $p = 11$ and $q = 13$, the modulus $n = 11 \cdot 13 = 143$
- $\phi(n) = \phi(143) = (p-1)(q-1) = 10 \cdot 12 = 120$
- Find e such that $\gcd(e, \phi(n)) = \phi(e, 120) = 1$
- Since $120 = 2^3 \cdot 3 \cdot 5$, we select $e = 7$
- $d = e^{-1} \pmod{\phi(n)}$ which gives $d = 7^{-1} \pmod{120}$ as $d = 103$
- Encryption $C = M^e \pmod{n}$ gives $C = 8^7 \pmod{143}$ as $C = 57$
  Decryption $D = C^d \pmod{n}$ gives $M = 57^{103} \pmod{143}$ as $M = 8$
- Encryption $C = M^e \pmod{n}$ gives $C = 11^7 \pmod{143}$ as $C = 132$
  Decryption $D = C^d \pmod{n}$ gives $M = 132^{103} \pmod{143}$ as $M = 11$

Security of RSA

- The public key (e, n) is published
  The private key parameters $p, q, \phi(n), d$ are kept by the user
- Breaking RSA:
  one or several or all instances $\rightarrow$ Computing M, given C and (e, n)
  all instances $\rightarrow$ Computing d, given (e, n)
- Taking discrete eth Root $\rightarrow$ Computing M
  Compute eth Root of $M^e \pmod{n}$ and obtain M
- Factoring $n \Rightarrow$ Computing d
  Factor $n = pq$, compute $d = e^{-1} \bmod (p-1)(q-1)$
- Computing $\phi(n) \Rightarrow$ Computing d
  Compute $d = e^{-1} \bmod \phi(n)$

Knowing $\phi(n) \Rightarrow$ Factoring n

- We know $n = pq$, we can also write:

$$n - \phi(n) + 1 = pq - (p-1)(q-1) + 1 = p + q$$

Thus we have pq and $p+q$, and we can write a quadratic equation whose roots are p and q

$$x^2 - (p+q)x + pq = x^2 - (n - \phi(n) + 1) + n = 0$$

The roots of the equations are

$$\frac{1}{2}(n - \phi(n) + 1 \pm \sqrt{(n - \phi(n) + 1)^2 - 4n}\ )$$

Example: For $n = 143$ and $\phi(n) = 120$, we write
$p,\ q = (1/2)(143 - 120 + 1 \pm \sqrt{(143 - 120 + 1)^2 - 4 \cdot 143}) = 11,\ 13$

Knowing $d \Rightarrow$ (Probabilistically) Factoring n

- Given d, we can write

$$d \cdot e = 1 + K\phi(n)$$

since they are inverses of one another mod $\phi(n)$

- Since $d \cdot e - 1$ is a multiple of $\phi(n)$, for $\gcd(a, n) = 1$ we can write

$$a^{de-1} = (a^{\phi(n)})^K = 1 \pmod{n}$$

- If there is a universal exponent b such that $a^b = 1 \pmod{n}$ for all a with $\gcd(a, n)$, then there is exists a probabilistic method for factoring n
- This probabilistic factorization algorithm is based on the Miller-Rabin primality test

Breaking RSA $\stackrel{?}{\Rightarrow}$ Factoring

- Factoring n indeed breaks the RSA encryption algorithm
- However, does "Breaking RSA" mean that we can factor n ?
- There is no general proof for such a claim — a lack of progress
- A related result: Breaking the low-exponent RSA is not as hard as factoring integers
- Strong evidence: An RSA breaker cannot be used for factoring integers