Elementary Number Theory - Exercise 7b

ETH Zürich - Dr. Markus Schwagenscheidt - Spring Term 2023

Problem 1. 1. Choose two 4-digit primes p and q and generate your own public and private RSA keys[1].

2. Exchange your public keys with another student and send each other a (very) short encrypted message[2]. Use the following encoding for the letters:

a	b	c	d	e	f	g	h	i	j	k	l	m
01	02	03	04	05	06	07	08	09	10	11	12	13

n	o	p	q	r	s	t	u	v	w	x	y	z
14	15	16	17	18	19	20	21	22	23	24	25	26

Keep in mind that long messages have to be split into blocks of size less than N.

3. Figure out the private key of your RSA partner.

Solution 1. We pick the primes $p = 1129$ and $q = 7681$ and form the RSA modulus

$$N = pq = 8671849.$$

We also compute Euler's totient function

$$\varphi(N) = (p-1)(q-1) = 8663040.$$

For the public key we randomly choose

$$e = 127$$

and check that we indeed have $\gcd(e, \varphi(N)) = 1$. Inverting e modulo $\varphi(N)$ gives the private key

$$d = 7094143.$$

Now let us use these keys to encrypt and decrypt the message "numbertheory". It is encoded as

$$m = 14\,21\,13\,02\,05\,18\,20\,08\,05\,15\,18\,25$$

Since m is larger than N, and N is a 7-digit number, it will be convenient to split m into blocks of length 6, so we need to encode

$$m_1 = 142113$$
$$m_2 = 020518$$
$$m_3 = 200805$$
$$m_4 = 151825$$

[1] You could ask Wolframalpha for random 4-digit primes.
[2] Use Wolframalpha for the necessary computations.

Note that we can omit the leading 0 in $m_2 = 020518$, since the receiver will see after decrypting m_2 that the number of digits is odd, and hence a leading 0 is missing. Computing $m_i^d \pmod N$, we obtain the encrypted messages

$$\begin{aligned} c_1 &= 1666533 \\ c_2 &= 7025487 \\ c_3 &= 8543101 \\ c_4 &= 1002246 \end{aligned}$$

One can check that we indeed have $c_i^d \equiv m_i \pmod N$, so the messages c_i can be decrypted again.

In order to figure out the private key d from the public key e and the modulus N, in this small example we can just factorize $N = pq$ (e.g. in Wolframalpha), then compute $\varphi(N) = (p-1)(q-1)$, and then invert e modulo $\varphi(N)$ to obtain d.

Problem 2. Let $N = pq$ be a product of two odd primes, and $\varphi(N) = (p-1)(q-1)$. Show that p and q can quickly be computed if N and $\varphi(N)$ are known.

For example, given $N = 7261$ and $\varphi(N) = 7072$, compute p and q.

Solution 2. Suppose that we know $N = pq$ and $\varphi(N) = (p-1)(q-1)$. Then we can compute

$$N - \varphi(N) + 1 = pq - (p-1)(q-1) + 1 = p + q.$$

If we know the product pq and the sum $p+q$, then p and q can be recovered as the solutions of a quadratic equation (this is known as Vieta's rule). Indeed, we have

$$(x-p)(x-q) = x^2 - (p+q)x + pq = x^2 - (N - \varphi(N) + 1)x + N,$$

so p and q are given by the formula

$$p, q = \frac{(N - \varphi(N) + 1) \pm \sqrt{(N - \varphi(N) + 1)^2 - 4N}}{2}$$

For example, for $N = 7261$ and $\varphi(N) = 7072$ we have

$$N - \varphi(N) + 1 = 190,$$

so p and q are given by

$$\frac{190 \pm \sqrt{190^2 - 4 \cdot 7261}}{2} = \frac{190 \pm 84}{2} = 53 \quad \text{and} \quad 137.$$

Problem 3. Let $N = pq$ be a product of two odd primes. If p and q are too close, then N can quickly be factored, using *Fermat's factorization method*: the idea is to find a, b with $N = a^2 - b^2$, since then $N = (a-b)(a+b) = pq$ is a factorization of N. If p, q are close, then b will be relatively small, so a will roughly be equal to $\sqrt{N}$. Here's the algorithm:

Compute $a = \lceil\sqrt{N}\rceil, \lceil\sqrt{N}\rceil + 1, \lceil\sqrt{N}\rceil + 2, \ldots$ until $a^2 - N = b^2$ is a square. Then $N = (a-b)(a+b)$ is a factorization of N.

Show that Fermat's method will always find a factorization of $N = pq$, and use it to factor $N = 5959$.

Solution 3. If $N = pq$, then we can write

$$N = pq = \left(\frac{p+q}{2}\right)^2 - \left(\frac{p-q}{2}\right)^2 =: a^2 - b^2.$$

Note that a must be at least $\lceil\sqrt{N}\rceil$, since otherwise $N = a^2 - b^2$ would be impossible. Hence, Fermat's factoring algorithm will find a and b after finitely many steps.

We apply the algorithm to $N = 5959$.

1. $a = \lceil\sqrt{N}\rceil = 78$. Then $a^2 - N = 78^2 - 5959 = 125$ is not a square.
2. $a = \lceil\sqrt{N}\rceil + 1 = 79$. Then $a^2 - N = 79^2 - 5959 = 282$ is not a square.
3. $a = \lceil\sqrt{N}\rceil + 2 = 80$. Then $a^2 - N = 80^2 - 5959 = 441 = 21^2$ *is* a square. We find

$$N = 5959 = 80^2 - 21^2 = (80-21)(80+21) = 59 \cdot 101.$$

Problem 4. Let $N = pq$, where p is an odd prime, but q is a *Carmichael number* with $\gcd(p, q) = 1$. Show that the RSA encryption and decryption still works on messages m with $\gcd(m, N) = 1$.

Solution 4. We have to be careful to distinguish between $\varphi(N)$ and $(p-1)(q-1)$, since these numbers will in general not be the same if q is a Carmichael number. The key generation uses $(p-1)(q-1)$. Let $1 < e, d < (p-1)(q-1)$ be coprime to $(p-1)(q-1)$ and such that $ed \equiv 1 \pmod{(p-1)(q-1)}$. A message m with $1 \leq m \leq N$ with $\gcd(m, N) = 1$ will be encrypted as

$$c = m^e \pmod{N},$$

and encrypted as

$$c^d \equiv (m^e)^d \equiv m^{ed} \equiv m^{1+k(p-1)(q-1)} \pmod{N}$$

so we need to show that

$$m^{(p-1)(q-1)} \equiv 1 \pmod{N}.$$

Since $\gcd(p, q) = 1$, by the Chinese Remainder Theorem it suffices to show that

$$m^{(p-1)(q-1)} \equiv 1 \pmod{p}, \quad \text{and} \quad m^{(p-1)(q-1)} \equiv 1 \pmod{q}.$$

The first identity follows from Fermat's little theorem, and the second identity follows since q is a Carmichael number (here we used that $\gcd(m, N) = 1$). Summarizing, we find

$$c^d \equiv m \pmod{N},$$

so the RSA decryption still works.

Problem 5. In cryptographic applications, it is often important to keep computation costs low. Hence, it is common to use rather small public keys e to speed up the RSA encryption. A typical choice is $e = 3$, since the encryption then takes only 2 multiplications. Here we discuss two attacks on RSA with $e = 3$.

1. Bob uses the public key $e = 3$ and the modulus $N = 126589$. Alice sends the encrypted message $c = 3375$ to Bob. Can you decrypt the message (without factoring N)?

2. Bob, Charles, and Dora all use the same public key $e = 3$, but with different moduli N_B, N_C, N_D. Let us assume that N_B, N_C, N_D are pairwise coprime[3]. Alice sends the same message m to Bob, Charles, and Dora, encrypted as c_B, c_C, c_D with their respective public keys and moduli. Use the Chinese Remainder Theorem to explain how m can be decrypted, without factoring any of the moduli.

Solution 5. 1. We know that $c = m^3 \pmod{N}$. Since $c = 3375$ is a cube in the integers, $3375 = 15^3$, the original message was $m = 15$. To avoid this problem, one can use *padding*: make the message m longer by adding random extra stuff at the end of the message, such that m^3 is larger than N.

2. By the Chinese Remainder Theorem, there is a unique x with $1 \leq x \leq N_B N_C N_D$ such that

$$\begin{aligned} x &\equiv c_B \pmod{N_B}, \\ x &\equiv c_C \pmod{N_C}, \\ x &\equiv c_D \pmod{N_D}. \end{aligned}$$

Since m^3 is another solution of this system, and $1 \leq m^3 \leq N_B N_C N_D$, we must have $x = m^3$. Hence, we can recover m by taking the third root of x.

Problem 6 (sage)**.** 1. Implement the RSA key generation and encryption/decryption. You could ask the user for primes p and q, or offer random primes.

2. Implement Fermat's factorization method, and factor $N = 105327569$.

[3] Bonus question: how can we break RSA if N_B, N_C, N_D are not pairwise coprime?