Math 55
Textbook RSA
Kenneth A. Ribet

> I have taught Math 55 three times in this millennium, most recently in spring, 2019. Colleagues suggested to me that I make available the notes that I wrote for myself before lecturing on RSA in 2019. This file is an adaptation of those notes.

I will assume that readers have access to the 8th edition of "Discrete Mathematics and Its Applications" by Kenneth H. Rosen. In that book, RSA is presented in §4.6. I begin by recalling some material concerning primes and greatest common divisors in §4.3 of Rosen.

Specifically, recall that a *prime* or *prime number* is an integer > 1 with no positive divisors other than 1 and itself. A prime is thus a number that "can't be factored."

The *greatest common divisor* or gcd of two integers a and b is defined whenever a and b are integers that are not both 0. It is literally the largest integer that divides both a and b. Bézout's theorem states that the gcd of a and b may be written in the form $sa + tb$, where s and t are integers. A consequence of this equation ("Bézout's identity") is that every divisor of a and b is a divisor of $\gcd(a, b)$. Thus the gcd of a and b is not only the "greatest" common divisor in terms of size: it's a multiple of every number that divides both a and b.

Bézout's theorem has important consequences, of which the following lemma is a sample. (In the lemma, a, b and c are integers, with a positive to fix ideas.)

Lemma. *If a divides a product bc and $\gcd(a, b) = 1$, then a divides c.*

This lemma, and its proof, are on page 287 of Rosen's text.

Corollary. *If p is a prime number and p divides bc, then p divides at least one of b, c.*

The proof is as follows: If p doesn't divide b, then $\gcd(p, b) = 1$ because p has only two positive divisors, namely 1 and p. The lemma above shows that p divides c in this case.

Note: the corollary above is known as Euclid's lemma.

Lemma. *If a and b are relatively prime, an integer is a multiple of both a and b if and only if it is a multiple of ab.*

To see this, suppose that n is a multiple of both a and b. Let s and t be as in Bézout's theorem, and write

$$n = n \cdot 1 = n(as + bt) = nas + nbt.$$

Because n is divisible by b, nas is divisible by ab; because n is divisible by a, nbt is divisible by ab. Hence $n = nas + nbt$ is divisible by ab.

Here is another lemma with the same general feeling:

Lemma. *If a is prime to b and to c, then a is prime to bc.*

Here, the word "prime" is a shorthand for "relatively prime"; the hypothesis is $1 = \gcd(a, b) = \gcd(a, c)$, and the desired conclusion is $1 = \gcd(a, bc)$. To prove the desired conclusion, it is enough to show that 1 is a linear combination of a and bc (i.e., a sum of a multiple of a and a multiple of bc); indeed, if this is true, then any divisor of both a and bc will be a divisor of 1 and therefore equal to 1.

To write 1 as a linear combination of a and bc, we use Bézout to write

$$1 = ax + by, \qquad 1 = za + wc.$$

Multiplying these together gives

$$1 = (ax + by)(za + wc) = a(xza + wcx + byz) + yw \cdot bc,$$

a linear combination of a and bc.

We will make use of **Fermat's Little Theorem**, which appears in §4.4 of the Rosen book:

If p is a prime and a is an integer not divisible by p, then $a^{p-1} \equiv 1 \bmod p$.

Let's call this statement Formulation I. There is also Formulation II:

If p is a prime and a is an integer, then $a^p \equiv a \bmod p$.

These two statements are easily seen to be equivalent:

In Formulation II, one can see two cases: the case where $a \equiv 0 \bmod p$ and the case where a is non-zero (and thus invertible) mod p. In the first case, the statement $a^p \equiv a \bmod p$ reads $0^p \equiv 0$, which is completely obvious. Thus we care only about the non-zero a mod p both in Formulation II and in Formulation I. The two congruences

$$a^p \equiv p, \qquad a^{p-1} \equiv 1 \pmod{p}$$

are then equivalent because we can pass between them in one direction by multiplying by a and in the other direction by multiplying by the multiplicative inverse of a.

Euler's theorem mod pq

Suppose that p and q are distinct primes. It follows from the second lemma above that the numbers mod pq that are relatively prime to pq are exactly the numbers that are invertible both mod p and mod q. More symbolically, $\gcd(a, pq) = 1$ if and only if $\gcd(a, p) = 1$ and $\gcd(a, q) = 1$.

Theorem. *If* $\gcd(a, pq) = 1$, *then* $a^{(bp-1)(q-1)} \equiv 1$ *mod* pq.

Because p and q are distinct prime numbers, they have no common factor $>$ 1. Therefore, by the first of the two lemmas, the congruence to be proved is equivalent to the pair of congruences

$$a^{(p-1)(q-1)} \overset{?}{\equiv} 1 \pmod{p}, \qquad a^{(p-1)(q-1)} \overset{?}{\equiv} 1 \pmod{q}.$$

The first congruence follows from the Fermat Little Theorem statement $a^{p-1} \equiv 1$ mod p by raising both sides of the congruence to the $(q-1)$st power. The second is identical to the first, except that the roles of p and q have been reversed.

Note: the exponent $(p-1)(q-1)$ is easily seen to be the number of a mod pq with $\gcd(a, pq) = 1$. Thus $(p-1)(q-1)$ is the "Euler phi function" ϕ of pq. There's a more general congruence

$$a^{\phi(m)} \equiv 1 \pmod{m}$$

for $m \geq 1$ and a relatively prime to m. This is called Euler's theorem (from the 18th century). When m is prime, Euler's theorem is the same statement as Fermat's Little Theorem.

RSA

§4.6 of Rosen's book begins with a historical discussion of cryptography, which you might find enjoyable. If you want to read more on the history of crypto, I might recommend "The Code Book" by Simon Singh. Another helpful source is the first chapter or two of the book used for our Math 116 (cryptography) course. The book is "An Introduction to Mathematical Cryptography"; you can download it for free on `link.springer.com` (using a campus IP).

For Math 55, the relevant content is in §§4.6.5–4.6.6, RSA. The idea, briefly, is that you can send encrypted messages to Alice (a fictional random user) if you know her *public key*. The encrypted message, even if intercepted, cannot be read by anyone who does not have Alice's private key. Alice can compute her private key easily, but it is believed that there is no practical way for anyone else to compute the private key if the public key has been set up competently.

It is understood in this situation that the message to be encrypted is a relatively small integer, say a number between 0 and 2^{512}. Computers routinely encode text, photos, videos (and so on) as long binary strings. The strings can be broken up into blocs of 512 bits; each bloc is then encoded individually.

Back to Alice. We stipulate in this discussion that Alice has access to a "random prime number generator" that will generate primes whose size is around 2^{256}. This is not a stretch. When I first asked my software for a random prime less than 2^{100}, I immediately got back `274741263583299337729451334107`. I asked again and obtained `366180878209372365295189650653`. When I replaced "100" with "256," my program output

```
108405305729297391065869170180854257614939521977752675243870462546571100126701.
```

Alice obtains two primes by a procedure like this—call them p and q. Alice notes down these primes and computes the two numbers

$$pq, \qquad (p-1)(q-1).$$

By the way, it is easy to verify that knowledge of the product $(p-1)(q-1)$ is *equivalent* to knowledge of the two primes p and q if one has the product pq. For example, if you know that

$$pq = 9946890792848916619$$

and

$$(p-1)(q-1) = 9946890783557367120,$$

you have two equations for the two unknowns p and q and can solve them to get the pair $\{\, p, q \,\}$. After making a substitution, you have to find the roots of a quadratic equation and will probably want to use the quadratic formula.

We admit that the number pq is so large that factoring programs will fail to factor this number into its two constituents p and q. Thus we consider that p and q are secrets even when pq is public knowledge.

Alice chooses a random number e mod $(p-1)(q-1)$ and checks whether

$$\gcd(e, (p-1)(q-1)) = 1.$$

If the gcd is bigger than 1, she repeats the process. She continues until she gets to a number e for which the gcd is 1.

To keep down the size of the numbers, say $p = 1234567891$ and $q = 8056981609$. Then $(p-1)(q-1) = 9946890783557367120$. A random integer between 0 and $m = (p-1)(q-1)$ is $e = 8469471273725486737$; this was the third random integer that I generated—the first two were even numbers. You can check that $\gcd(e, m) = 1$. Bézout gives

$$1 = -2351874692700353087e + 2002548895193940751m.$$

An inverse for e mod m is then -2351874692700353087, which is the same as 7595016090857014033 mod m.

Alice's public key is the pair of numbers (pq, e). In my example, her public key consists of 9946890792848916619 and 8469471273725486737. In the general scenario, pq is so large that it cannot be factored. We can factor the pq in this example very quickly, but we pretend for the discussion that this is not the case. Because of this "play acting," the public key in question might be described as a "toy example."

Alice's private key is the inverse of e mod m. In my example, her private key is 7595016090857014033. Rosen uses the letter "d" for the private key; let's do that too. Then

$$ed \equiv 1 \pmod{m}, \qquad m = (p-1)(q-1).$$

This means

$$ed = 1 + k(p-1)(q-1)$$

for some integer $k \geq 0$.

The point now is that if M is an integer prime to pq, then

$$M^{ed} = M \cdot (M^{(p-1)(q-1)})^k \equiv M \cdot 1^k = M \pmod{pq}.$$

Said differently,

$$(M^e)^d \equiv M \pmod{pq}.$$

If I want to send an encrypted message to Alice, I take my plain text message M and raise it to the eth power mod pq. Alice can decrypt the message by raising the encrypted message M^e to the dth power mod pq.

Digression: why is M prime to pq? If M is a random integer mod pq, it's prime to pq with probability $\frac{(p-1)(q-1)}{pq}$. If p and q are both large, this probability is extremely close to 1.

For example, suppose my message is 1291999611 and we use the e, p and q as in the toy example. Then the encrypted version of the message is

$$1291999611^{8469471273725486737} \pmod{9946890792848916619},$$

or 4526414970456754822. It was validating for me to check that

$$4526414970456754822^{7595016090857014033} \equiv 1291999611$$

modulo 9946890792848916619.

Although the choices of p and q in our example are too small for hiding secrets, they are too large for hand computations. Here's a completely silly example to is suitable for hand calculation: We'll imagine that Alice chooses $p = 13$, $q = 17$. Then $pq = 221$ and $(p-1)(q-1) = 192$ (which we pretend to be a deep secret).

A random number that Alice might choose for her "e" is 79. Then Alice's public key consists of 221 and 79.

To get Alice's private key, we invert 79 mod 192. The Euclidean algorithm gives

$$1 = -17 \cdot 79 + 7 \cdot 192.$$

Hence -17 is the inverse of 79 mod 192, though it might be nicer to use a positive number and say that the inverse is 175.

Our "message" to be transmitted is some number mod 221. If the message is a multiple of 13 or of 17, we can and definitely should crack the RSA modulus 221 by computing the gcd of our message with 221. The probability that a random message will break the modulus is $\left(1 - \frac{192}{221}\right)$, or about 13%. As mentioned above, this probabity becomes minuscule if p and q are large (one says "of cryptographic size").

Let's say the message is 123. Then the encrypted message that we send to Alice is 123^{79} mod 221, or 98. The numbers involved are small enough that weird things happen. For example, $123^6 \equiv -1$ mod 221. Because $79 = 13 \cdot 6 + 1$,

$$123^{79} \equiv 123 \cdot (123^6)^{13} \equiv 123 \cdot (-1)^{13} \equiv -123 \equiv 98 \pmod{221}.$$

We send Alice the encrypted message "98."

Alice computes the plain text "123" by calculating 98^{175} mod 221. Again, $98^6 \equiv -1$ mod 221; this shouldn't be a surprise because

$$98^6 \equiv (123^{79})^6 = (123^6)^{79} \equiv (-1)^{79} = -1 \pmod{221}.$$

Writing $175 = 29 \cdot 6 + 1$, we see that

$$98^{175} \equiv (-1)^{29} \cdot 98 \equiv 221 - 98 = 123 \pmod{221}.$$

This example might be more friendly than the large "toy example" above. I hope so, anyway!