Powers in Modular Arithmetic, and RSA Public Key Cryptography

Lecture notes for Access 2008

It was a long time from Mary Queen of Scots and substitution ciphers until the end of the 1900's. Cryptography underwent the evolutionary and revolutionary changes which Simon Singh chronicals in *The Code Book*. By the mid 1970's there were amazingly complicated encryption algorithms which could be made essentially unbreakable. For example, in Chapter 6, Singh mentions the Lucifer cipher, a special version of which is known as the Data Encryption Standard, or DES.

However, no matter how convoluted the the encryption methods were, and how frequently the keys were changed for security reasons, all methods required that both parties to the message possessed *the* key for encryption and decryption.... and it was just assumed, because this had always been the case, that if you possessed the method to encrypt a message, then this was equivalent after perhaps a little work, to also knowing how to decrypt it. By the mid 1970's there were thousands of couriers flying all over the world, whose only job was to transfer cipher keys.

As the the precursor to the internet, namely the ARPAnet, was beginning to grow, Whitfield Diffie and Martin Hellman, as well as others, realized the huge potential for electronic transactions, together with the need for assured security. Diffie-Hellman were perhaps the first to realize that there was an entirely new way to think of cryptography; that perhaps there were encryption keys which you could let everyone in the world know, but for which you could never the less keep secret the decryption key. This would solve the problem of key distribution, since if you wanted to receive secure messages you could tell the world how to encrypt anthing they wanted to send you, but only you would know the decryption key which could stay safely at home. Diffie-Hellman called such encryption keys, "trapdoor", or "one way" functions, because knowing the encryption function did not automatically allow clever people to work out the decryption function. In 1977, Ronald Rivest, Adi Shamir and Leonard Adleman described one of the easiest one-way functions, and the resulting method of public key cryptography is called RSA, in their honor. As we shall see, this method relies on number theory and modular arithmetic, and will use everything we've been talking about up to this point.

Remark 1. In our examples so far we've been assigning numbers to each letter of a plaintext and then using modular arithmetic to construct a cipher, number by number (or letter by letter). In practice this just amounts to a (letter) substitution cipher, and so can be broken easily with frequency analysis. What we will do for RSA cryptography (and what has been done in cryptography for a long time before RSA) is to make packets consisting of lots of letters, and encrypt those. In RSA we will use HUGE moduli $N = pq$, which are products of two different prime numbers, and break messages into packets whose lengths fall into the residue range of N. Then we'll encrypt each packet back into the residue range by using a power function mod N. We'll hope to decrypt it with another power function.

As an example of what we mean by "packets", if N has at least 13 digits (i.e. $N > 10^{12}$), then you can use the table on page 9 of Tom Davis' notes, *Cryptography*, to convert any 6-character expression into a number in the residue range of N, since each character is represented with two digits.

Example 1. Use the Davis table on page 9 of his notes to convert the two word sentence *Hello there!* into two number packets, each of which is less than 10^{12}.

Remark 2. If you use the encryption function $f(x) \equiv x + a \mod N$, which corresponds to Caesar shifts, anyone can deduce the decryption function $g(x) \equiv x - a \mod N$ after at most N guesses to find the value of a. (Think of the example above, with $N > 10^{12}$.) Similarly, if you specify the encryption function $f(x) \equiv ax \bmod N$, for $gcd(a, N) = 1$, then ACCESS students and other hard-working smart people could use the Euclidean algorithm to quickly find a multiplicative inverse b of a mod N, in order to find the decryption function $g(x) \equiv bx$ mod N. So neither of these examples is a one-way function, even with big message packets. It will turns out that for long enough packets (e.g longer than 200 digits), suitable modular power encryption IS believed to be a one-way function.

We'll understand the power problem for $N = pq$ by first understanding it for $N = p$, a prime. Let's experiment:

Example 2. Let $N = 11$. Let our candidate encryption function be $f(x) \equiv x^2 \mod 11$, where we take the domain and range to be the residue numbers $\{0, 1, 2, \ldots, 10\}$. Complete the table below and explain why this function won't work to encrypt the numbers in our residue range.

x	0	1	2	3	4	5	6	7	8	9	10
x^2	0	1	4	9	5						

Example 3. Keeping $N = 11$, show that the function $f(x) \equiv x^3 \mod 11$ does encrypt (permute) the residue numbers:

x	0	1	2	3	4	5	6	7	8	9	10
x^3	0	1	8								

Example 4. We might hope that if our encryption function is $f(x) \equiv x^e \mod N$, then our decryption function is $g(x) \equiv x^d \mod N$, for some power d. For the encryption function in the previous example $e = 3$. We shall now deduce a possible value for the decryption power d: Since $f(2) = 8 \mod 11$, we want $g(8) = 2$, i.e. $8^d \equiv 2 \mod 11$. Compute successive powers of 8 until you are able to solve this equation for d. (Hint: $d = 7$.)

Exercise 1. But we need to check that the decryption power $d = 7$ works for every x in our residue range! Let group number x check that this is so, for the residue number $x + 1$, except that group 1 gets to double check $x = 3$ along with group 2, since we just did $x = 2$. Be clever to minimize your computing!

Exercise 2. Since RSA cryptography uses moduli $N = pq$, where p and q are (HUGE) prime numbers, we'll experiment with small prime numbers $p = 3$, $q = 5$, $N = 15$, and use the mod 15 table of powers below to figure out good and bad encryption powers e. (A good encryption function permutes the residue numbers, so that it has an inverse decryption function.) First, you will have to fill in rows 6 and 7 of the table!

Power table, mod 15

power →	1	2	3	4	5	6	7	8	9	10
residue										
0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1
2	2	4	8	1	2	4	8	1	2	4
3	3	9	12	6	3	9	12	6	3	9
4	4	1	4	1	4	1	4	1	4	1
5	5	10	5	10	5	10	5	10	5	10
6										
7										
8	8	4	2	1	8	4	2	1	8	4
9	9	6	9	6	9	6	9	6	9	6
10	10	10	10	10	10	10	10	10	10	10
11	11	1	11	1	11	1	11	1	11	1
12	12	9	3	6	12	9	3	6	12	9
13	13	4	7	1	13	4	7	1	13	4
14	14	1	14	1	14	1	14	1	14	1

Exercise 3. $f(x) \equiv x^3 \mod 15$ is a good encryption function. What part of the power table confirms this fact? Find a power d so that $g(x) \equiv x^d \mod 15$ is the decryption function for $f(x)$. Use the power table to check your work.

When decryption powers exist, and how to find them

We've been doing a lot of experimentation with modular arithmetic, which is a really good way to get ideas about what might be true. Number theory has been a favorite for many great mathematicians, and so some of their names are attached to the following important theorems, which maybe they were also led to by experimentation. These results from two centuries ago turn out to be the underpinning of RSA cryptography.

Theorem 1 (Fermat's Little Theorem). If p is a prime and if $0 < a < p$ is a residue number, then $a^{p-1} \equiv 1 \mod p$. (And, for any integer b, $b^p \equiv b \mod p$.)

Proof. Pick any non-zero residue a as above, and consider the corresponding row of the mod p multiplication table. (You can make this less abstract by using the mod 7 table as an example, see below.) Since a has a multiplicative inverse mod p, $ax \equiv ay$ only when $x \equiv y$. (Why?) Therefore the residues across the row, namely the numbers

$$1a, 2a, 3a, ..., (p-1)a$$

must all be different, i.e. a permuation of the non-zero residues $1, 2, ..., (p-1)$. Thus the product of all these terms satisfies

$$(1a)(2a)...(p-1)a \equiv (1)(2)...(p-1) \mod p,$$

$$a^{p-1}(1)(2)...(p-1) \equiv (1)(2)...(p-1) \mod p.$$

Multiply both sides of this equation by the multiplicative inverses of $2, 3, ...(p-1)$, i.e. cancel the term $(2)(3)...(p-1)$ from both sides of the equation. Deduce

$$a^{p-1} \equiv 1 \mod p.$$

Finally, if b is an integer and p/b then $b^p \equiv b \equiv 0 \mod p$. If p is not a factor of b then write a for its non-zero residue mod p. In this case $b^p \equiv a^p \equiv aa^{p-1} \equiv a \equiv b \mod p$. □

Example 5. Here's how to illustrate Little Fermat concretely, using $p = 7$. Start with the mod 7 multiplication table, without the zero row and column:

×	1	2	3	4	5	6
1	1	2	3	4	5	6
2	2	4	6	1	3	5
3	3	6	2	5	1	4
4	4	1	5	2	6	3
5	5	3	1	6	4	2
6	6	5	4	3	2	1

Take any row, say the row for $a = 3$. The entries going across are the residues for

$$(3)(1), (3)(2), (3)(3), (3)(4), (3)(5), (3)(6)$$

and they are just a permuation of the original non-zero residues. Thus, taking the product of the entries in this row, mod 7, we have

$$3^6 6! \equiv 6! \mod 7.$$

6! has a multiplicative inverse mod 7, since it's a product of numbers with multiplicative inverses. Multiplying both sides of the equation by this number, we deduce a special case of Little Fermat, for $a = 3, p = 7$:

$$3^6 \equiv 1 \mod 7.$$

Corollary 1. if $f(x) \equiv x^e \mod p$, and d is a multiplicative inverse of e, mod $p-1$, then $g(x) \equiv x^d \mod p$ is the inverse function of f (on the set of residue numbers).

Proof. Notice that if $x = 0$ the result holds. Thus we can assume $x = a$, a non-zero residue. Since e and d are multiplicative inverses mod $p-1$, we have

$$ed = 1 + m(p-1)$$

for some counting number m. Thus

$$g(f(a)) \equiv g(a^e) \equiv (a^e)^d \equiv a^{ed}$$

$$\equiv a^{1+m(p-1)} \equiv a^1(a^{p-1})^m \equiv a(1^m) \equiv a \mod p,$$

by Fermat's Little Theorem! This shows that g is the inverse function to f. $\square$

Example 6. For $p = 11$ and $e = 3$, find d using this Corollary. Does your answer agree with the earlier example, where we found acceptable d by brute force?

Theorem 2 (Euler-Fermat Theorem). If $N = pq$ is a product of two prime numbers, define $N_2 = (p-1)(q-1)$. If a is any residue number mod N, then

$$a^{N_2+1} \equiv a \mod N.$$

Proof. The real Euler-Fermat Theorem is more general than what is stated above, and part of a much bigger story. (You can read about it in a good number theory book, or Wikipedia). I found the following "magic" proof of the special case stated above by browsing the Wikipedia topic "RSA Cryptography"! Let a be any residue number mod N, with $N = pq$ as above. Use the usual laws of exponents to write

$$a^{N_2+1} = a^{(p-1)(q-1)+1} = a^{pq-p-q+2} = a^{(q-1)p}a^{-q+2} = (a^{(q-1)})^p a^{-q+2}.$$

But Fermat's Little Theorem says

$$(a^{(q-1)})^p \equiv a^{(q-1)} \mod p.$$

Substitute this into the the exponent equation and in interpret $\mod p$:

$$a^{N_2+1} \equiv (a^{(q-1)})^p a^{-q+2} \equiv a^{(q-1)}a^{-q+2} \equiv a^{q-1-q+2} \equiv a \mod p.$$

Actually, there's one comment we need to make about the computation above: If a is divisible by p then it doesn't have a multiplicative inverse $\mod p$ and the term a^{-q+2} which has a negative exponent doesn't make sense. But, luckily in this case $a^{N_2+1} \equiv a \mod p$ also holds, because both sides are congruent to $0 \mod p$. In summary, and then interchanging the roles of p and q above, we see that

$$\begin{aligned} a^{N_2+1} &\equiv a \mod p, \\ a^{N_2+1} &\equiv a \mod q. \end{aligned}$$

This means $a^{N_2+1} - a$ is a multiple of p and also a multiple of q. Since p and q are two different prime numbers this means that $a^{N_2+1} - a$ is actually a multiple of pq, i.e. $a^{N_2+1} \equiv a \mod N$. Magic! □

Example 7. What is the value of N_2 when $N = 15$? Does the mod 15 power table verify Euler-Fermat in this case?

Corollary 2. Let $N = pq$, $N_2 = (p-1)(q-1)$ as above. Let e be relatively prime to N_2. Then for residues $x \mod N$, $f(x) \equiv x^e \mod N$ (with range in the residue numbers) has inverse function $g(x) \equiv x^d \mod N$. Here d is the multiplicative inverse of e, mod N_2. (This result turns out to be the basis for RSA cryptography.)

Proof. Since e is chosen to be relatively prime to N_2 we know the $\mod N_2$ multiplicative inverse d exists, from all of our work with the Euclidean algorithm! Thus $ed = 1 + mN_2$ for some counting number m. If $m = 1$, then $ed = 1 + N_2$ and

$$g(f(x)) \equiv (x^e)^d \equiv x^{ed} \equiv x^{1+N_2} \equiv x \mod N$$

by Euler-Fermat. If $m = 2$, then $ed = 1 + 2N_2$ and

$$g(f(x)) \equiv (x^e)^d \equiv x^{ed} \equiv x^{1+2N_2} \equiv x^{1+N_2} x^{N_2} \equiv x^1 x^{N_2} \equiv x^{1+N_2} \equiv x \mod N$$

by two applications of Euler-Fermat. Similarly, if $ed = 1 + 3N_2$ then three applications of Euler-Fermat gives the result, and by induction we see that the same result holds for all counting numbers $m = 1, 2, 3, 4, 5, ...$ □

Example 8. For $p = 3$ and $q = 5$ we have $N = 15$, $N_2 = 8$. For the encryption power $e = 3$, compare we the decryption power d guaranteed by this corollary to the power(s) we found earlier from the mod 15 power table.

We're now ready to discuss the RSA algorithm, and why this method of encryption is a one-way function. We'll start with the section 9 Davis-notes example. It's also good to read chapters 5-6 of *The Code Book*.