Math 114 Discrete Mathematics

Cryptography and the number theory behind it

D Joyce, Spring 2018

Private-key codes. There are two kinds of secret codes, private-key and public-key codes. The private-key codes are what you would expect them to be, person X takes a message M and encrypts it using an encryption function ϕ into a another message $\phi(M)$ and sends that to person Y. Person Y receives the encrypted message $\phi(M)$ and knows the decryption function ϕ^{-1} and uses it to reconstruct the original message $\phi^{-1}(\phi(M)) = M$. The encryption functions ϕ and ϕ^{-1}, also called *keys*, are kept private.

The keys ϕ and ϕ^{-1} often work on sequences of numbers of some sort or other, usually on integers in some range $0, 1, \ldots, n-1$, and can often be described in terms of arithmetic functions. The integers in that range can be identified with the ring of integers modulo n, denoted $\mathbf{Z}/n\mathbf{Z}$, or more simply $\mathbf{Z}_n$, and the arithmetic is addition, subtraction, and multiplication modulo n. The message M is a string of integers modulo n,

$$M = a_1 a_2 \ldots a_k$$

and its encrypted message $\phi(M)$ is the string of encrypted integers

$$\phi(M) = f(a_1) f(a_2) \ldots f(a_k)$$

modulo n, and the encryption key is a function $f : \mathbf{Z}_n \to \mathbf{Z}_n$, while the decryption key is its inverse $f^{-1} : \mathbf{Z}_n \to \mathbf{Z}_n$.

An example of this is the shift cipher $f : \mathbf{Z}_n \to \mathbf{Z}_n$ given by $f(a) \equiv a + c \pmod{n}$ where c is a constant, and $f^{-1}(a) \equiv a - c \pmod{n}$.

Private-key codes can be impossible to break. That is, if you receive the encrypted message $\phi(M)$, it can be impossible to recover M. But there are requirements to set up the encryption scheme that make it impracticable in many cases. Before the message is sent, persons X and Y have to meet or otherwise agree on the encryption and decryption keys ϕ and ϕ^{-1}. In many situations that's fine. But if X and Y decide at the moment (where they're chatting over public communications) that they want to have a secure remote conversation, they can't set up such keys. That's the situation over the internet when you want to transfer money, for instance, or have any other secure conversation.

Public-key codes. In the 1970s the concept of public-key codes was developed. For a public-key code, the encryption key ϕ is public (assumed to be known to everyone), but the decryption key ϕ^{-1} remains private. It would work because the private key could not be determined from the public key. You would think that if you knew one function, it wouldn't be hard to find the inverse function. But there are a number of these *trap-door* functions that don't seem to be easily inverted.

The most used cryptography system is the RSA algorithm proposed by Rivest, Shamir, and Adleman in their article "On Digital Signatures and Public Key Cryptosystems," *Communications of the ACM* 21 (1978): 120–126. The RSA algorithm is based on exponentiation modulo n. The encoding and decoding algorithms are actually functions $\mathbf{Z}_n \to \mathbf{Z}_n$. That means that the message M has to start out as a string of elements $a_1 a_2 \ldots a_k$, each in $\mathbf{Z}_n$. A real message actually needs a preliminary coding to convert it into a string of numbers modulo n.

The theorem that ensures that RSA will work is Euler's theorem, although it can be verified by the Chinese remainder theorem. What is needed is Euler's theorem in the case that n is the product of two distinct primes, $n = pq$. In that case Euler's theorem says that whenever a is relatively prime to n, then

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

Here are the steps in creating the two keys for the encoding and decoding functions for the RSA system.

1. Select two large prime numbers p and q, and let n be their product pq.

2. Select a number e relatively prime to
$$(p-1)(q-1).$$
(This number e can be pretty small, typically $e=3$ or $e=5$.)

3. Compute
$$d \equiv e^{-1} \pmod{(p-1)(q-1)},$$
that is, solve the congruence
$$ex \equiv 1 \pmod{(p-1)(q-1)},$$
and call the solution d. Solving that linear congruence will involve the Euclidean algorithm.

4. The encoding algorithm is the function $\mathbf{Z}_n \to \mathbf{Z}_n$ which converts the original message a into the coded message $a^e \pmod{n}$. Make public this encoding algorithm, that is, make public n and e.

5. The decoding algorithm is the function $\mathbf{Z}_n \to \mathbf{Z}_n$ which converts an coded message b back into the original message $b^d \pmod{n}$. Keep private this decoding algorithm, that is, don't tell anyone d, and don't tell anyone p or q either, because then d could be determined.

Why does this work as a public-key code? What's publicized is n and e. What's needed to break the code is to find d. It appears that there's no other way to find d other than to factor n into its two prime factors. There is, however, no known efficient way to factor large numbers like n.

Why is the decoding algorithm actually inverse to the encoding algorithm? Let's check that starting with a message letter a, then encoding it, then decoding it, returns the original message a. That is, we need to verify the congruence
$$(a^e)^d \equiv a \pmod{n}.$$

We'll do that in two cases. First, when a is relatively prime to n (which is the case for nearly all a). In that case, we can use Euler's theorem modulo n. We'll also use the fact that
$$ed \equiv 1 \pmod{(p-1)(q-1)},$$
that is, that $ed = 1 + c\,\phi(n)$ for some number c. Then, modulo n we have
$$\begin{aligned}(a^e)^d &\equiv a^{ed}\\ &\equiv a^{1+c\,\phi(n)}\\ &\equiv a\,(a^{\phi(n)})^c\\ &\equiv a\,1^c \equiv a \pmod{n}\end{aligned}$$
That takes care of the case that $(a,n)=1$.

Suppose now that a is not relatively prime to n. If $a \equiv 0 \pmod{n}$, then clearly, $(a^e)^d \equiv a \pmod{n}$. Otherwise a is divisible by exactly one of the two primes p and q. Let's say p divides it. Of course, modulo p we have $(a^e)^d \equiv 0 \equiv a \pmod{p}$. Now, since a is relatively prime to q, we can apply Fermat's little theorem modulo q (which is which what Euler's theorem generalizes). Then we have
$$\begin{aligned}(a^e)^d &\equiv a^{ed}\\ &\equiv a^{1+c(p-1)(q-1)}\\ &\equiv a\,(a^{q-1})^{c(p-1)}\\ &\equiv a\,1^{c(p-1)} \equiv a \pmod{q}\end{aligned}$$
Now, since $(a^e)^d \equiv a$ both modulo p and modulo q, therefore the congruence holds modulo their product $n = pq$.

That finishes the proof that the decoding function actually is inverse to the encoding function.

Math 114 Home Page at `http://math.clarku.edu/~djoyce/ma114/`