Lecture Notes 11

45 Generating RSA Encryption and Decryption Exponents

We showed in section 44 (lecture notes 10) that RSA decryption works for $m \in \mathbf{Z}_n^*$ if e and d are chosen so that

$$ed \equiv 1 \pmod{\phi(n)}, \tag{1}$$

that is, d is e^{-1} (the inverse of e) in $\mathbf{Z}^*_{\phi(n)}$.

We now turn to the question of how Alice chooses e and d to satisfy (1). One way she can do this is to choose a random integer $e \in \mathbf{Z}^*_{\phi(n)}$ and then solve (1) for d. We will show how to solve for d in Sections 46 and 47 below.

However, there is another issue, namely, how does Alice find random $e \in \mathbf{Z}^*_{\phi(n)}$? If $\mathbf{Z}^*_{\phi(n)}$ is large enough, then she can just choose random elements from $\mathbf{Z}_{\phi(n)}$ until she encounters one that also lies in $\mathbf{Z}^*_{\phi(n)}$. A candidate element e lies in $\mathbf{Z}^*_{\phi(n)}$ if $\gcd(e, \phi(n)) = 1$, which can be computed efficiently using Algorithm 42.2 (Euclidean algorithm).[1]

But how large is large enough? If $\phi(\phi(n))$ (the size of $\mathbf{Z}^*_{\phi(n)}$) is much smaller than $\phi(n)$ (the size of $\mathbf{Z}_{\phi(n)}$), Alice might have to search for a long time before finding a suitable candidate for e.

In general, $\mathbf{Z}^*_m$ can be considerably smaller than m. For example, if $m = |\mathbf{Z}_m| = 210$, then $|\mathbf{Z}^*_m| = 48$. In this case, the probability that a randomly-chosen element of $\mathbf{Z}_m$ falls in $\mathbf{Z}^*_m$ is only $48/210 = 8/35 = 0.228\ldots$.

The following theorem provides a crude lower bound on how small $\mathbf{Z}^*_m$ can be relative to the size of $\mathbf{Z}_m$ that is nevertheless sufficient for our purposes.

Theorem 1 *For all $m \geq 2$,*

$$\frac{|\mathbf{Z}^*_m|}{|\mathbf{Z}_m|} \geq \frac{1}{1 + \lfloor \log_2 m \rfloor}.$$

Proof: Write m in factored form as $m = \prod_{i=1}^t p_i^{e_i}$, where p_i is the i^{th} prime that divides m and $e_i \geq 1$. Then $\phi(m) = \prod_{i=1}^t (p_i - 1)p_i^{e_i - 1}$, so

$$\frac{|\mathbf{Z}^*_m|}{|\mathbf{Z}_m|} = \frac{\phi(m)}{m} = \frac{\prod_{i=1}^t (p_i - 1)p_i^{e_i - 1}}{\prod_{i=1}^t p_i^{e_i}} = \prod_{i=1}^t \left(\frac{p_i - 1}{p_i}\right). \tag{2}$$

To estimate the size of $\prod_{i=1}^t (p_i - 1)/p_i$, note that $(p_i - 1)/p_i \geq i/(i + 1)$. This follows since $(x - 1)/x$ is monotonic increasing in x, and $p_i \geq i + 1$. Then

$$\prod_{i=1}^t \left(\frac{p_i - 1}{p_i}\right) \geq \prod_{i=1}^t \left(\frac{i}{i+1}\right) = \frac{1}{2} \cdot \frac{2}{3} \cdot \frac{3}{4} \cdots \frac{t}{t+1} = \frac{1}{t+1}. \tag{3}$$

Clearly $t \leq \lfloor \log_2 m \rfloor$ since $2^t \leq \prod_{i=1}^t p_i \leq m$ and t is an integer. Combining this fact with equations (2) and (3) gives the desired result. ∎

[1] $\phi(n)$ itself is easily computed for an RSA modulus $n = pq$ since $\phi(n) = (p - 1)(q - 1)$.

For n a 1024-bit integer, $\phi(n) < n < 2^{1024}$. Hence, $\log_2(\phi(n)) < 1024$, so $\lfloor \log_2(\phi(n)) \rfloor \le 1023$. By Theorem 1, the fraction of elements in $\mathbf{Z}_{\phi(n)}$ that also lie in $\mathbf{Z}^*_{\phi(n)}$ is at least 1/1024. Therefore, the expected number of random trials before Alice finds a number in $\mathbf{Z}^*_{\phi(n)}$ is provably at most 1024 and is most likely much smaller.

46 Diophantine equations and modular inverses

Now that Alice knows how to choose $e \in \mathbf{Z}^*_{\phi(n)}$, how does she find d? That is, how does she solve (1)? Note that d, if it exists, is a multiplicative inverse of $e \pmod n$, that is, a number that, when multiplied by e, gives $1 \pmod n$.

Equation (1) is an instance of the general Diophantine equation

$$ax + by = c \tag{4}$$

Here, a, b, c are given integers. A solution consists of integer values for the unknowns x and y. To put (1) into this form, we note that $ed \equiv 1 \pmod{\phi(n)}$ iff $ed + u\phi(n) = 1$ for some integer u. This is seen to be an equation in the form of (4) where the unknowns x and y are d and u, respectively, and the coefficients a, b, c are $e, \phi(n), 1$, respectively.

47 Extended Euclidean algorithm

It turns out that (4) is closely related to the greatest common divisor, for it has a solution iff $\gcd(a, b) \mid c$. It can be solved by a process akin to the Euclidean algorithm, which we call the *Extended Euclidean algorithm*. Here's how it works.

The algorithm generates a sequence of triples of numbers $T_i = (r_i, u_i, v_i)$, each satisfying the invariant

$$r_i = au_i + bv_i \ge 0. \tag{5}$$

The first triple T_1 is $(a, 1, 0)$ if $a \ge 0$ and $(-a, -1, 0)$ if $a < 0$. The second trip T_2 is $(b, 0, 1)$ if $b \ge 0$ and $(-b, 0, -1)$ if $b < 0$.

The algorithm generates T_{i+2} from T_i and T_{i+1} much the same as the Euclidean algorithm generates $(a \bmod b)$ from a and b. More precisely, let $q_{i+1} = \lfloor r_i / r_{i+1} \rfloor$. Then $T_{i+2} = T_i - q_{i+1}T_{i+1}$, that is,

$$\begin{aligned} r_{i+2} &= r_i - q_{i+1}r_{i+1} \\ u_{i+2} &= u_i - q_{i+1}u_{i+1} \\ v_{i+2} &= v_i - q_{i+1}v_{i+1} \end{aligned}$$

Note that $r_{i+2} = (r_i \bmod r_{i+1})$, [2] so one sees that the sequence of generated pairs (r_1, r_2), (r_2, r_3), $(r_3, r_4), \ldots$, is exactly the same as the sequence of pairs generated by the Euclidean algorithm. Like the Euclidean algorithm, we stop when $r_t = 0$. Then $r_{t-1} = \gcd(a, b)$, and from (5) it follows that

$$\gcd(a, b) = au_{t-1} + bv_{t-1} \tag{6}$$

Returning to equation (4), if $c = \gcd(a, b)$, then $x = u_{t-1}$ and $y = v_{t-1}$ is a solution. If c is a multiple of $\gcd(a, b)$, then $c = k \gcd(a, b)$ for some k and $x = ku_{t-1}$ and $y = kv_{t-1}$ is a solution. Otherwise, $\gcd(a, b)$ does not divide c, and one can show that (4) has no solution. See Handout 6

[2] This follows from the division theorem, which can be written in the form $a = b \cdot \lfloor a/b \rfloor + (a \bmod b)$.

for further details, as well as for a discussion of how many solutions (4) has and how to find all solutions.

Here's an example. Suppose one wants to solve the equation

$$31x - 45y = 3 \tag{7}$$

In this example, $a = 31$ and $b = -45$. We begin with the triples

$$\begin{aligned} T_1 &= (31, 1, 0) \\ T_2 &= (45, 0, -1) \end{aligned}$$

The computation is shown in the following table:

i	r_i	u_i	v_i	q_i
1	31	1	0	
2	45	0	-1	0
3	31	1	0	1
4	14	-1	-1	2
5	3	3	2	4
6	2	-13	-9	1
7	1	16	11	2
8	0	-45	-31	

From $T_7 = (1, 16, 11)$ and (5), we obtain

$$1 = a \times 16 + b \times 11$$

Plugging in values $a = 31$ and $b = -45$, we compute

$$31 \times 16 + (-45) \times 11 = 496 - 495 = 1$$

as desired. The solution to (7) is then $x = 3 \times 16 = 48$ and $y = 3 \times 11 = 33$.