Appendix B

RSA encryption

B.1 The RSA algorithm

Choose two distinct primes p, q, and let

$$n = pq.$$

We know that if x is coprime to n (ie $\gcd(x, n) = 1$) then

$$x^{\phi(n)} = x^{(p-1)(q-1)} \equiv 1 \bmod n.$$

Choose an exponent e coprime to $\phi(n)$, and let $\alpha : \mathbb{Z}/(n) \to \mathbb{Z}/(n)$ be the map

$$\alpha : x \mapsto x^e.$$

Then we can determine f such that

$$ef \equiv 1 \bmod \phi(n),$$

eg by using the Euclidean algorithm. Let $\beta : \mathbb{Z}/(n) \to \mathbb{Z}/(n)$ be the map

$$\alpha : x \mapsto x^f.$$

Then if x is coprime to n

$$x^{ef} \equiv x \bmod n,$$

ie

$$\beta(\alpha(x)) = x;$$

β is the *inverse* of α, at least for x not divisible by p or q.

B.2 Encryption

Let us choose very large primes p, q, say with about 150 digits, or about 500 bits, each.

This will not take long, using either the Miller-Rabin or the AKS test. If we take an odd integer u with about 150 digits at random, and then test $u, u+1, u+2, \ldots$ for primality we can be be reasonably sure that we will meet a prime in about $\ln u \approx 15 \ln 10$ steps, by the Prime Number Theorem. (Of course we can reduce the number of tests by omitting even numbers, and perhaps numbers divisible by small primes, so the number might be reduced to a dozen or so.)

Next we choose $e \in (1, \phi(n))$ at random. We *publish* the numbers n and e — RSA is a *public key encryption* system, and these are our public keys.

Now if someone wants to send us a secret message they encode it using our public keys. We have computed the secret key f, and thus can decode the message.

We are betting that nobody can determine the factors p and q by factorising n, or determine f in some other way. In effect, we are relying on the belief that *factorisation cannot be computed in polynomial time*. More precisely, there is no algorithm that can factorise any number n in less that $P(\ln n)$ steps, where $P(x)$ is some fixed polynomial.

For example, dividing by all numbers up to $\sqrt{n}$ is an *exponential time algorithm* since

$$\sqrt{x} = e^{\ln x/2}.$$

Remarks:

1. If we want 1000-bit security, we would probably choose n to have 1024 bits, to simplify computation.

2. Note that $x^e \bmod n$ can be computed in polynomial time (probably in quadratic time) by repeatedly squaring x, always working modulo n.

3. There is an extremely small probability that some block x of the message will be divisible by p or q, and will therefore be "corrupted". However, we can ignore this possibility on the grounds that is far more likely to be corrupted in other ways.

B.3 Elliptic curve encryption

After the RSA encryption algorithm was published, it was realised that other groups arising in number theory could perhaps be used in place of $(\mathbb{Z}/n)^\times$.

The most popular candidate was *elliptic curves*. A general curve of degree 3 in x and y is an elliptic curve. An elliptic curve can be brought to the standard form

$$y^2 = x^3 + ax^2 + bx + c.$$

There ia a natural group structure on an elliptic curve, arising as follows. Suppose P, Q are 2 points on the curve. Suppose the line PQ

has the equation $L(x, y) = 0$, where $L(x, y)$ is linear. The points where PQ meets the elliptic curve will be given by a cubic polynomial in x, say. Two of the three roots of this polynomial will correspond to P and Q. Thus the cubic polynomail will factorise completely into linear factors, and the third factor will correspond to a point $R = P \circ Q$. If the coefficients a, b, c of the elliptic curve are rational, and the points P, Q have rational coordinates then the point R will also have rational coordinates. This defines a natural group structure on the set of rational points on the curve.

All this can be carried out with elliptic curves defined over a finite field, eg $\mathbb{F}_{2^n}$. The encryption generally corresponds to the map

$$P \mapsto eP,$$

where e plays a similar rôle to e in the RSA encryption, except that now the bet is that nobody will be able to compute the inverse map

$$eP \mapsto P.$$

(This is known as the "discrete log problem", and can be applied equally to the group $(\mathbb{Z}/n)^\times$.)

Elliptic curve cryptography (ECC) is gradually taking over from RSA encryption. It is generally believed to be more secure, and the computations involve smaller numbers, so can be carried out in less time.

'Arithmetic on elliptic curves' is probably the most active area of research in number theory today, and was the basic tool in Wiles' proof of Fermat's Last Theorem. Elliptic curves give rise to zeta functions like Riemann's, with Euler-like factorisation into terms corresponding to primes.