RSA

The Algorithm

This trapdoor encryption system was introduced by Rivest, Shamir and Adleman (1978). It has, so far, withstood all known attacks.

The system is simplicity itself. Each user of the system makes two numbers, e_U and n_U public and keeps a number d_U secret. In order for A to send a message to B, A looks up B's public values and, if the message is m (written as a number), then A blocks the message into pieces of size $< n_B$ and sends $c = m^{e_B} \bmod n_B$. Then B decodes by $m = c^{d_B} \bmod n_B$. The security of the system lies in the choices of the public and private keys.

To understand these choices we need to consider some number theory.

Euler's Function

For any integer n, Euler's Totient Function, $\phi(n)$ is the number of integers greater than or equal to 1 which are relatively prime to n. It can be shown that:

$$\phi(n) = n \prod_{p|n} (1 - \frac{1}{p}), \quad \text{p a prime.}$$

Example: $\phi(12) = |\{1,5,7,11\}| = 4$.

$$12(1\text{-}1/2)(1\text{-}1/3) = 12(1/2)(2/3) = 2(2) = 4.$$

Euler's Theorem: If gcd (a,n) = 1 then $a^{\phi(n)} \equiv 1 \bmod n$

A Corollary

Corollary: If n is a product of distinct primes then $a^{1+t\phi(n)} \equiv a \bmod n$ for any integer t.

Pf: Let p be any prime that divides n. If gcd(a,p) = 1, then

$$a^{1+t\phi(p)} = a\left(a^{\phi(p)}\right)^t \equiv a(1)^t \equiv a \bmod p$$

is valid by Euler's Theorem. On the other hand, if a ≡ 0 mod p, then the statement is trivially true. Since the congruence holds for each prime dividing n, it also holds for n.

Application to RSA

For the RSA choices, each user selects two prime numbers (**about 100 digits long**) p and q and sets $n_U = pq$. Note that $\phi(n_U) = (p-1)(q-1)$. [p and q are no longer used, but must be kept secret]. Next, e_U is selected subject to $1 < e_U < \phi(n_U)$ and gcd $(e_U, \phi(n_U)) = 1$. Finally, d_U is calculated (using the extended Euclidean Algorithm) so that $e_U d_U \equiv 1 \bmod \phi(n_U)$. We now see that

$$c^{d_B} = \left(m^{e_B}\right)^{d_B} \equiv m^{1+r\phi(n_B)} \equiv m \bmod n_B$$

by the corollary.

Practical RSA

To set up RSA two large primes must be obtained. Finding p and q can be done with a fast *primality tester*.

The practical user of RSA must be on guard against some common pitfalls, known as protocol failures. In these cases, how a message gets encoded to a numerical equivalent may defeat the cryptosystem.

The RSA scheme can be used for signatures in the usual way.

Breaking RSA

The only known way to break the system is to find $\phi(n_U)$ which is *almost* equivalent to factoring n_U. The *Rabin variation* is a version of RSA in which it can be shown that the security is equivalent to the difficulty of factoring. So the security rests (perhaps) on the difficulty of factoring large numbers. To avoid those situations where fast factoring algorithms exist one should select p and q so that:

a) p and q are not too close (one should be a few decimal digits longer)

b) p-1 and q-1 have a small gcd and both have at least one large prime factor.

One must also be aware of the current state of the art in factoring large numbers.

PGP

RSA in practice is very slow (1500 times slower than DES). As a result it is rarely used for very long messages or high traffic situations. Frequently it can be found as a part of a faster crytposystem.

Phil Zimmerman's public domain program ***PGP*** (Pretty Good Privacy) is a combination of RSA and a fast private key system, IDEA (International Data Encryption Algorithm). The RSA algorithm is used to encrypt the private key generated for the IDEA. Once this is transmitted, the private key is used to decrypt the message which is sent, encrypted by IDEA. IDEA is considered to be much stronger than DES and uses a 128 bit key.