Attacks on Public Key

Public key algorithms are theoretically easier to attack than symmetric key algorithms because the attacker has a copy of the public key that was used to encrypt the message. The job of the attacker is further simplified because the message presumably identifies which public key encryption algorithm was used to encrypt the message. Public key algorithm attacks fall into two categories: *key search attacks* and *analytic attacks*.

Key Search Attacks

Key search attacks are the most popular kind of attacks to mount on public key encrypted messages because they are the most easily understood. These attacks attempt to derive a private key from its corresponding public key.

Let's consider the RSA public key system. Key search attacks are performed by attempting to factor a large number that is associated with the public key. The number is the product of two prime numbers. Once the large composite number is factored, the private key can be readily derived from the public key.

Analytic Attacks

The other way of attacking a public key encryption system is to find a fundamental flaw or weakness in the mathematical theory on which the encryption system is based. The first public key encryption system to be patented was based on a mathematical problem called the Superincreasing Knapsack Problem. A few years after this technique was suggested, a way was found to mathematically derive the secret key from the public key in a very short amount of time.

Known Versus Published Methods

It is worth noting that it is always possible that there is a difference between the best known methods and the best published methods. If a major mathematical break-through in factoring is discovered, it might not be published for all to see. For example, if a new method is developed by a government agency, it might be kept a secret so that the agency can decrypt messages sent by officials of other countries.

Implementation Strength

Strong algorithms and good choices for keys are not always sufficient to assure cryptographic strength. It is also vital that the implementation of the algorithm, along with any key generation and storage, be correct and carefully tested. A buggy implementation, poor random number generation, or sloppy handling of keys may all increase the exposure of your information.