How not to choose random large primes

X.509, the Secure Sockets Layer and other programs choose random large primes for use in RSA and other public key ciphers.

A. Lenstra et al. collected several million 1024-bit RSA public keys from X.509 certificates and PGP on the Web. They performed simple security tests on these numbers, for example, checking whether they were composite, whether they had any very small prime factors, whether the key was the square of a prime, and whether the keys were all different. They found a small number (5.0%) that failed at least one of these tests.

In some cases an RSA key is repeated because one user posts it in multiple places on the Web. This is not a problem. But if two users have the same RSA public key, and one of them knows it, then he could read enciphered mail for the other. This is a problem.

Form the graph whose vertices are primes and whose edges are actual RSA public keys connecting their prime factors.

If c different RSA keys are used, then one would expect this graph to have c disjoint connected components, each consisting of two vertices joined by a single edge. There would be a total of $2c$ primes and c edges of multiplicity one.

The actual graph the researchers found contained many edges with multiplicity higher than one. These are RSA keys that two or more users share.

The actual graph contained about 2000 disjoint connected components with three or more vertices. Most of these were trees of depth one rooted at a common factor of two RSA keys, and most with 2 or 3 leaves. But one tree had depth one and 4627 vertices, that is, 4627 RSA keys shared a single common prime factor. There were also six connected components with four vertices and three edges, but not a depth one tree. Incredibly, the graph also contained the complete graph K_9 on nine vertices.

The authors discovered these results by computing the GCD of each pair of RSA keys. Two RSA keys can both be factored this way if they share a single large prime.

This finding suggests poor seeds for a random number generator, that is, two (or more) users used the same seed to generate a prime for their RSA key.

Generating an RSA modulus consists in finding two random prime numbers in such a way that these primes are not selected by anyone else. In order to do this securely, one must use a random seed about as long as the RSA key being generated, that is, twice as long as each prime.

Clearly, this rule is not followed. For example, if the random seeds are only 32 bits long, then one would begin to generate repeated primes after about $2^{16} \approx 65000$ primes were generated, by the birthday paradox.