Name: Harris Christiansen (christih@purdue.edu)

Homework 4

1. **Factorizing the RSA modulus.** Let N be the product of two random n-bit prime numbers p and q. Recall that $\varphi(N)$ is the size of $\mathbb{Z}_N^*$, and we have $\varphi(N) = (p-1)(q-1)$. Construct an efficient algorithm that takes as input N and $\varphi(N)$, and outputs the prime factors of N.

   Solution.

   (a) We are given $N = p \cdot q$ and $\varphi(N) = (p-1)(q-1)$. Given N and $\varphi(N)$ this is effectively a problem of two equations, two unknowns.

$$\varphi(N) = (p-1)(q-1) = (p \cdot q) - p - q + 1$$
$$p = N/q$$
$$\varphi(N) = ((N/q) \cdot q) - (N/q) - q + 1 = N - (N/q) - q + 1$$
$$\varphi(N) \cdot q = (N \cdot q) - (N/q \cdot q) - (q \cdot q) + (1 \cdot q)$$
$$\varphi(N) \cdot q = (N \cdot q) - N - q^2 + q$$
$$q^2 + (\varphi(N) - N - 1)q + N = 0$$
$$\text{Let } k = -\varphi(N) + N + 1.$$
$$q^2 - kq + N = 0$$
$$\text{Using the Quadratic formula...}$$
$$q = \frac{k \pm \sqrt{k^2 - 4N}}{2}$$
$$p = N/q$$

2. **Sophie-Germain Primes.** Recall that the Prime Number Theorem states that there are roughly $\frac{N}{\log N}$ prime numbers $< N$. To generate a random n-bit prime number, recall that, we followed the following two steps

   - First, we counted the number of n-bit primes, and
   - Finally, we generated T random numbers and one of them turned out to be a prime number.

   We chose T such that the probability of finding an n-bit prime number in these T attempts is $\geqslant (1 - 2^{-t})$, for a parameter t.

   Now, we want to do this for the Sophie-Germain primes. We shall rely on the conjecture that there are $\frac{N}{\log^2 N}$ Sophie-Germain primes $< N$.

   (a) How many Sophie-Germain primes need n-bits in their binary representation?

   (b) Construct an algorithm that that as input (n, t) and outputs a random n-bit Sophie-Germain prime with probability $\geqslant (1 - 2^{-t})$.

   Solution.

   (a) Given that there are $\frac{N}{\log^2 N}$ Sophie-Germain primes < N, we can determine the number of Sophie-Germain primes with exactly N-bits by using this equation and subtracting the number of Sophie-Germain primes with exactly (N-1)-bits.

   $$\frac{N}{\log^2 N} - \frac{N/2}{\log^2 N/2}$$

   Where N is the largest n-bit prime number.

   (b) Since the probability of finding an n-bit prime number is $\geqslant (1-2^{-t})$, by selecting a random (n+1)-bit number, subtracting 1, and dividing by 2, such that the result is an n-bit number, will result in selecting a Sophie-Germain prime with probability $\geqslant (1 - 2^{-t})$.

3. **Encryption along with Signature.** Recall that in RSA-based public-key encryption, if Bob announces his public-key $\mathsf{pk}_B = (N_B, e_B)$ then other parties can encrypt and send messages to Bob that he can decrypt (using the trapdoor d_B that he keeps with himself).

   Recall that in RSA-based signatures, if Alice announces her public-key $\mathsf{pk}_A = (N_A, e_A)$ then she can sign messages that other people can verify that Alice has generated the signature (because Alice holds the trapdoor d_A).

   How can Alice encrypt a message m of her choice and send it to Bob so that only Bob can recover the message, and Bob is guaranteed that it is indeed Alice who sent the ciphertext?

   Solution.

   (a) Begin with Alice encrypting message m for Bob using his public-key $\mathsf{pk}_B = (N_B, e_B)$, via RSA-based public-key encryption. Call this m_e

   (b) Now have Alice sign the encrypted message m_e using RSA-based signatures. Call this m_{es}

   (c) Alice can now send Bob the pair (m_e, m_{es}), which Bob can use m_{es} to guarantee Alice sent the ciphertext, and only Bob can decrypt the message m_e using his private-key d_B.