Homework Assignment # 6

Due Date: Wednesday, October 25

1. Compute the following exponentiations $x^e \bmod m$ applying the square-and-multiply algorithm:

   (a) $x = 2$, $e = 79$, $m = 101$

   (b) $x = 3$, $e = 197$, $m = 101$

   After every iteration step, show the exponent of the intermediate result in binary notation.

2. Encrypt and decrypt by means of the RSA algorithm with the following system parameters:

   (a) $p = 3,\ q = 11,\ a = 7,\ x = 5$

   (b) $p = 5,\ q = 11,\ b = 3,\ x = 9$

   Only use a pocket calculator at this stage.

3. One of the most attractive applications of public-key algorithms is the establishment of a secure session key for a private-key algorithm such as DES over an unsecure channel.

   Assume Bob has a pair of public/private keys for the RSA cryptosystem. Develop a simple protocol using RSA which allows the two parties Alice and Bob to agree on a shared secret key. Who determines the key in this protocol, Alice, Bob, or both?

4. In practice it is sometimes desirable that both communication parties influence the selection of the session key. For instance, this prevents the other party from choosing a key which is a *weak key* for the private-key algorithm used. Many popular block ciphers such as DES and IDEA have weak keys. Messages encrypted with weak keys can be recovered relatively easily from the ciphertext.

   Develop a protocol similar to the one above in which both parties influence the key. Assume that both Alice and Bob have a pair of public/private keys for the RSA cryptosystem. Please note that there are several valid approaches to this problem. Just show one.

5. This exercise[1] exhibits what is called a protocol failure. It provides an example where ciphertext can be decrypted by an opponent, without determining the key, if a cryptosystem is used in a careless way. The moral is that is insufficient to use a "secure" algorithm in order to guarantee "secure" communication.

   Suppose Bob has an RSA Cryptosystem with a large modulus n for which the factorization can not be found, e.g., n is 1024 bits long. Suppose Alice sends a message to Bob by representing each alphabetic character as an integer between 0 and 25 (i.e., $A \leftrightarrow 0$, $B \leftrightarrow 1$, ...) and then encrypting each letter as a separate plaintext character.

   (a) Describe how Oscar can easily decrypt a message which is encrypted in this way.

   (b) Illustrate this attack by decrypting the following ciphertext which was encrypted using RSA with $n = 18721$ and $b = 25$. Do not factor the modulus.

   $$y = 365,\ 0,\ 4845,\ 14930,\ 2608,\ 2608,\ 0$$

   (c) One way of fixing this problem is to encrypt several letters at once. But even with this approach, RSA is still a *deterministic* cryptosystem, that is, the same sequence of plaintext letters maps to the same ciphertext. This allows some form of traffic analysis (i.e., to draw some conclusion about the cleartext by merely observing the ciphertext). For instance, Oscar can see when an earlier message is being sent again, even though he can not decrypt the message.

   Suggest what has to be done to change this situation. The goal is that the encryption yields different ciphertexts even if the plaintext is the same.

[1] This problem was taken from D. Stinson: Cryptography, Theory and Practice. CRC Press

6. As we saw in the lecture, the modulus of RSA has been enlarged over the years in order to thwart the dramatically improved attacks. As one would assume, public-key algorithms are becoming slower as the word length increases. We will study the relation between word length and performance in this problem. The performance of RSA, and of almost any other public-key algorithm, is dependent on how fast modulo exponentiation with large number can be performed.

   (a) Assume that one modulo multiplication or squaring with k bit numbers takes $c \cdot k^2$ clock cycles, where c is a constant. How much slower is RSA encryption/decryption with 1024 bits compared to RSA with 512 on average. Only consider the encryption/decryption itself with an exponent of full length and the square-and-multiply algorithm.

   (b) In practice the Karatsuba algorithm, which has an asymptotical complexity of $\mathcal{O}(k^{\log_2 3})$, is often used for long number multiplication in cryptography. Assume that this more advanced technique requires $c' \cdot k^{\log_2 3} = c' \cdot k^{1.585}$, c' being a constant, clock cycles for multiplication or squaring. What is the ratio between RSA encryption with 1024 bits and RSA with 512 bits if the Karatsuba algorithm is used in both cases? Again, assume that full length exponents are being used.

7. **(Optional Problem, 10 extra points)**

   There are ways to improve the square-and-multiply algorithm, that is, to reduce the number of operations required. Although the number of squaring is fixed, the number of multiplications can be reduced. Your task is to come up with a modified version of the square-and-multiply algorithm which requires fewer multiplications. Give a detailed description of how the new algorithm works and what the complexity is (number of operations).

   Hint: Try to develop a generalization of the square-and-multiply algorithm which processes more than one bit at a time. The basic idea is to handle k (e.g., $k = 3$) exponent bits per iteration rather than one bit in the original square-and-multiply algorithm.