Institut for Matematik og Datalogi
Syddansk Universitet, Odense

4. december 2018
JFB/RF

Eksaminatorier DM534 Uge 49

[I denne uge er opgave 1 og 4 de mest eksamensrelevante.]

1. Why is a cryptographically secure hash function used in connection with RSA digital signatures?

2. With RSA, why would you never use the value 2 as one of of the two primes p and q?

3. In RSA, why must the message being encrypted be a non-negative integer strictly less than the modulus?

4. Consider an RSA system with Alice's public key $N = 1517$ and $e = 17$. Note that $1517 = 37 \cdot 41$.

   (a) Find Alice's secret key d. Use the Extended Euclidean Algorithm from slides 51–52 of the RSA slides used in lectures (there, e and d are called a and s (and d also x at one point)).

   (b) Try encrypting 423. Use the algorithm for fast modular exponentiation (page 33 on the slides). How many times during the recursive execution is the "**if** k is odd" case encountered, and how many times is the "**if** k is even" case encountered? [Do not include the base cases $k = 0$ and $k = 1$ in the counts.]

   (c) Decrypt the number obtained above, using fast modular exponentiation. Is the result correct? How many times during the recursive execution is the "**if** k is odd" case encountered, and how many times is the "**if** k is even" case encountered? [Do not include the base cases $k = 0$ and $k = 1$ in the counts.]

5. Why in RSA is it necessary that $gcd(e_A, (p_A - 1)(q_A - 1)) = 1$? Find an example (that is, values e_A, p_A and q_A) where this greatest common divisor is not equal to 1.

6. Try executing the Miller-Rabin primality test on 11, 15, and 561. First, what type of numbers are they (note: 561 is known from the slides)? For each, run the Miller-Rabin test for at least one value a of your own choice, preferably for more. Use e.g. Maple or a simple Java program importing the class `BigInteger` from the Java library for executing the exponentiations (simply raising to a power and then using modulus should work in reasonable time for numbers this size, hence there is no need to use the fast modular exponentiation algorithm in this exercise). Which calculations showed that the composite numbers were not prime—the first line (the Fermat test) or later lines?

   With 561, be sure to try an a relatively prime to 561 (most are, 2 is a simple example). What happens differently if you try $a = 3$? Can you explain the latter?

7. Find four different square roots of 1 modulo 143, i.e., numbers which multiplied by themselves modulo 143 give 1 (and which are at least 0 and less than 143). You may consider writing a simple program for finding them.

8. [Optional] Add two of these different square roots which are not negatives of each other modulo 143 (two where adding them together does not give 143). Find the greatest common divisor of this result and 143. Subtract these same two different square roots and find the greatest common divisor of this result and 143. Think about why you get these results. (These effects are the starting point for the math behind fast primality testing algorithms.)

9. If you have time, for fun try breaking these two encrypted messages:

   - This was entitled "Cold Country". It was encrypted using a monoalphabetic substitution cipher. A monoalphabetic substitution cipher works similarly to a Caesar cipher. However, instead of just shifting the alphabet a fixed amount to get the mapping defined for each letter, the key is a permutation of the alphabet, so that you decide according to this key what letter "A" maps to, what letter "B" maps to, etc. If the alphabet has 29 letters, the number of keys is now 29! Why? The original message here was in English, so there are only 26 letters. How many possible keys are there?

     TOWWJPHJC ZY RXW PHOTWYR ZYPHJC ZJ RXW

SFOPC. UFYR FB ZR ZY QFIWOWC SZRX ZQW
RXFMYHJCY FB BWWR CWWD.

Discuss which techniques you used.

- This English message was encrypted using a Caesar cipher. Decrypt it.

  YMNX HWDUYTLWFR NX JFXD YT IJHNUMJW.

  Discuss which techniques you used.