PWr, Inf. algo. CRYPTOGRAPHY, 2022 assignments list # 7, public key encryption

1. As you know, ElGamal public key encryption enables re-encryption: from a ciphertext of m one can get easily a random ciphertext of m. However, it requires knowledge of the public key used to create this ciphertext.

   - What happens if a wrong public key is used during re-encryption?
   - Show that it is possible to re-encrypt without knowledge of the public key, if a ciphertext of M has the form
     $$(\mathrm{PK}^k \cdot M, g^k, \mathrm{PK}^l, g^l)$$
     for random k, l.
   - Show that it is possible to re-encrypt without the public key if ElGamal is used to encrypt only one bit where 0 is encrypted as (PK^k, g^k), while 1 is encrypted as $(\mathrm{PK}^k \cdot z, g^k)$ for an arbitrary $z \neq 1$.

2. Recall Chinese Reminder Theorem.

   - Show that for an RSA number $n = p \cdot q$ there a 4 numbers $x < n$ such that $x^2 = 1$. Two of them are obvious: 1 and $n - 1$. What are the other ones? They are called non-trivial roots of 1.
   - Show that knowledge of a nontrivial root of 1 modulo n enables breaking RSA based on n.

3. One of the problems of RSA is its computational complexity. In order to compute $m^d \bmod n$ the basic trick is as follows:

   - find a binary representation of $d = d_u d_{u-1} \dots d_0$ (we can assume that $d_u = 1$)
   - put $c_u = m$ and inductively compute $c_{j-1} = c_j^2 \bmod n$ if $d_{j-1} = 0$, else $c_{j-1} = c_j^2 \cdot m \bmod n$.

   Show that $c_0 = m^d \bmod n$.

   Estimate computational complexity of this exponentiation method.

4. Complexity of RSA encryption and decryption can be reduced by application of Chinese Remainder Theorem. Check how to do it and what we can gain regarding the computation cost.

/-/ Mirosław Kutyłowski