Problems for week 3, Cryptography Course - TDA 352/DIT 250

General remarks on problems for the weekly problem session: Exercises will be classified in four different levels:

1. **Easy**: the exercise will require low numerical computations or it can be just a way to look back at the content of the lecture. Exercises of this level should easily be done with just *pen and paper* and are **important to pass** the exam.
2. **Medium**: the exercises will require some time to do (from 5 to 15 minutes each). Maybe a separate paper for some computation is needed! You need to study a bit to answer the questions. These exercises also **may appear** in the exam.
3. **Hard**: the exercises will require you to spend a lot of time doing numerical computations (and we highly recommend using a PC) **or** the questions are real challenge to see if you understood the course in depth. Some of these exercises **may appear** in the exam.
4. **Think**: problems that aim to using your imagination. You are invited to discussion with your colleagues/friends/family and find your best solutions. Generally, the exercises of this level do not take a lot of time in writing the solutions but they will let you think/discuss for (maybe) 30/40 minutes.

The problems this week are partly simple exercises in using primitives using small numbers, partly old exam problems, intended to test these aspects (for the second item, mostly improper use of primitives).

Easy

---

1. (a) Use the extended Euclidean algorithm to show that 42 and 25 are relatively prime and to find x and y such that $42x + 25y = 1$.
   (b) Are there many solutions x, y to (a)?
   (c) Compute $1/25$ in $\mathbb{Z}_{42}^*$.
2. Use Fermat's theorem to simplify the following modular expressions
   (a) $3^{102} \pmod{11}$
   (b) $3^{502} \pmod{11}$
3. Given two prime numbers $p = 11$ and $q = 17$, an RSA modulus $N = p \cdot q$, and an encryption exponent $e = 3$.
   (a) Compute the RSA decryption exponent d.
   (b) Encrypt the message $M = 107$. Then apply the decryption algorithm to recover the plaintext.
   (c) Do the same for the message $M = 110$. Any comments?
4. (Not solved completely in class; requires too much computations.) Find all generators in $\mathbb{Z}_{17}^*$. Find also all subgroups and their generators.
5. Describe $\mathbb{Z}_{15}^*$ and its subgroups.
6. Show that 3 is not a generator of $\mathbb{Z}_{83}^*$.

Medium

7. Use Euler's theorem to find the last (least significant) decimal digit of the number 3^{100}.

8. Use the Chinese Remainder Theorem to find the number which when divided by 3 gives the remainder 1 and when divided by 7 gives the remainder 4.

9. Consider an RSA system with modulus $N = pq$, public key e and private key d. Show that if the Adversary finds out $\Phi(N) = (p-1)(q-1)$, (s)he can easily factorize N.

10. We consider RSA encryption.

    (a) It is often recommended to choose a small public key exponent to increase efficiency. A common choice is $e = 3$. Why not $e = 2$?

    (b) We consider double encryption using a common modulus N and two public keys e_1 and e_2 with corresponding private keys d_1 and d_2. So a message m is encrypted first using RSA encryption with the key e_1; the result is encrypted again using key e_2. There are two arguments why this does not increase security. One is an argument against double encryption in general and the other against this particular proposal. Give both arguments.

    (c) Public-key algorithms are usually used for encrypting short messages. But if we need to encrypt a longer message we can split it into blocks, use RSA for each block and use a block cipher mode. Which of the two modes CBC and CTR would you recommend in such a situation?

11. Let us consider the group $\mathbb{Z}_{25}$.

    (a) Write down all the elements in $\mathbb{Z}_{25}^*$ (the group of invertible elements of $\mathbb{Z}_{25}$).

    (b) Is 2 a generator of $\mathbb{Z}_{25}^*$? Prove it.

    (c) Using $g = 2$, find a generator of the subgroups G_i where i is the order of the group and $i \in \{1, 2, 4, 5, 10, 20\}$.

Hard

12. **Consider** the exercise (11), can you find a way to generalize that solution to any cyclic group $G = \langle g \rangle$ of order $\text{ord}(G) = p_1^2 \cdot p_2$ where p_1, p_2 are two distinct primes.

13. Show that if a and b are both at most 2^n, then Euclid's algorithm computes $\gcd(a, b)$ in at most $2n$ iterations/recursive calls.
    Hint: Prove first that if $a \geq b > 0$, then $a \bmod b < a/2$.

14. Alice and Bob use RSA with the same modulus but different encryption exponents e_A and e_B. An eavesdropping adversary gets hold of two ciphertexts, the message m encrypted for Alice and Bob, respectively. How would the adversary proceed?

15. We consider the use of RSA encryption with a 1024 bits modulus to transmit a 56 bit DES key to be used as session key. One can develop a meet-in-the-middle attack on this practice, based on the fact that a random 56 bit number m can with significant probability (> 10 %) be factored as $m = m_1 \cdot m_2$, where both m_1 and m_2 are 28 bit numbers.

    So, assume that the DES key m has such a factorisation and that the ciphertext $c = m^e \bmod N$ has been intercepted by an adversary. Describe the attack in detail and give estimates of how much computation and storage that is needed for the attack.

16. Let p be a large prime and g a generator for $\mathbb{Z}_p^*$. We consider the discrete logarithm problem, i.e., the problem of finding x when $y = g^x$ is known (in $\mathbb{Z}_p^*$). The purpose of this exercise is to demonstrate how the least significant bit of x can be found using a single modular exponentiation.

(a) Show that $g^{(p-1)/2} = p - 1$.

(b) Show how one can find the least significant bit of x by computing $y^{(p-1)/2}$.

Think

17. (This problem has more to do with modular arithmetic than with cryptography. The moral is that a computer representing negative numbers in two-complement form is more easily understood as working in $\mathbb{Z}_{2^w}$.) A certain computer has word length w, so its words can represent $\mathbb{Z}_{2^w}$ in binary in the natural way. Its instruction set implements the four arithmetical operations in $\mathbb{Z}_{2^w}$. Now a certain programming language wants to think of $\mathbb{Z}_{2^w}$ as containing the elements

$$-2^n, -2^n + 1, \ldots - 1, 0, 1, \ldots, 2^n - 1$$

where $n = w - 1$ (rather than $0, 1 \ldots, 2^w - 1$) and call this set `int`. Explain how negative numbers are represented as bit patterns and show that the arithmetic instructions work correctly.

18. Consider the RSA encryption system. Why do we require that p and q are of the same size? Would RSA be secure if p and q were **coprime** numbers but **not** prime numbers?