

Problem Set 5

Problem 1: Fermat's little theorem and Euclidean algorithm

- (a) **Modular roots** Show that the 7th root of 47 modulo 143 is

$$[47^{103} \bmod 143].$$

Note that $143 = 11 \cdot 13$.

- (b) **Computing seemingly huge numbers by hand** Compute the final two (decimal) digits of 3^{1000} (by hand).

Hint: The answer is $[3^{1000} \bmod 100]$. If $\gcd(a, b) = 1$, then $\phi(a \cdot b) = \phi(a) \cdot \phi(b)$.

- (c) Use the extended Euclidean algorithm to compute (by hand) the modular inverse of 20 modulo 457. Verify your answer with WolframAlpha.

Problem 2: Additive cyclic groups

In this problem, we study the additive group $(\mathbb{Z}_N, +)$ for some integer $N > 1$, for example $N = 9$.

- (a) Explain how to view $(\mathbb{Z}_N, +)$ as cyclic group generated by $\langle 1 \rangle$. Draw the circle for $N = 9$.
- (b) Which are the other $x \in \mathbb{Z}_N$ that generate the cyclic group $\langle x \rangle = (\mathbb{Z}_N, +)$? How many generators are there?
- (c) **Easy discrete-logarithm problem.** Explain why the discrete-logarithm problem (as described above Def 8.62 in [KL]) in the additive group $(\mathbb{Z}_N, +)$ generated by $\langle x \rangle$ is easy to solve, while the discrete-logarithm problem in multiplicative groups (\mathbb{Z}_p^*, \cdot) for some (suitably chosen) large prime p is believed to be hard.

Problem 3: Diffie-Hellman

- (a) **Computational Diffie-Hellman** Define the hardness of the Computational Diffie-Hellman problem with respect to the group-generation algorithm \mathcal{G} .
- (b) **DDH is hard \Rightarrow CDH is hard \Rightarrow DLog is hard** Prove that hardness of the CDH problem implies hardness of the discrete-logarithm problem, and that hardness of the Decisional Diffie-Hellman problem implies hardness of the CDH problem.

Problem 4: RSA

- (a) Let $N = pq$ be a RSA-modulus and let $(N, e, d) \leftarrow \text{GenRSA}$. Show that the ability of efficiently factoring N allows to compute d efficiently. Given

$$N = 2140310672120493293362298457402658192652108411554313695782475927669427$$

try to compute $\phi(N)$. If that takes too long compute $\phi(p \cdot q)$, where

$$p = 58240080352490526776497122885950201,$$

$$q = 36749789134330529121473391864214027.$$

Knowing that compute d for given N and $e = 11$. Is it

$$1945736974654993903056634961275143725147489931575688907101782888641091?$$

- (b) **Factoring RSA Moduli** Let $N = pq$ be a RSA-modulus and let $(N, e, d) \leftarrow \text{GenRSA}$. In this exercise, you show that for the special case of $e = 3$, computing d is equivalent to factoring N . Show the following:
1. The ability of efficiently factoring N allows to compute d efficiently. This shows one implication.
 2. Given $\phi(N)$ and N , show how to compute p and q . **Hint:** Derive a quadratic equation (over the integers) in the unknown p .
 3. Assume we know $e = 3$ and $d \in \{1, 2, \dots, \phi(N) - 1\}$ such that $ed \equiv 1 \bmod \phi(N)$. Show how to efficiently compute p and q . **Hint:** Obtain a small list of possibilities for $\phi(N)$ and use 2.
 4. Given $e = 3$, $d = 29531$ and $N = 44719$, factor N using the method above.

Problem 5: Diffie-Hellman Key Exchange

- (a) **Man-In-The-Middle Attacks** Describe in detail a man-in-the-middle attack on the Diffie-Hellman key-exchange protocol whereby the adversary ends up sharing a key k_A with Alice and a (different) key k_B with Bob, and Alice and Bob cannot detect that anything has gone wrong. What happens if Alice and Bob try to detect the presence of a man-in-the-middle adversary by sending each other (encrypted) questions that only the other party would know how to answer?
- (b) **Diffie-Hellman problem** Let us consider the cyclic group \mathbb{Z}_{13}^* .

1. Show that 2 is a generator of \mathbb{Z}_{13}^* . Draw the circle of the cyclic group.
2. In \mathbb{Z}_{13}^* , it holds that

$$\text{DH}_2(6, 9) = \text{DH}_2(2^5, 2^8) = 2^{40} = 2^{40 \bmod 12} = 2^4 = 3$$

Show in the same way that $\text{DH}_2(12, 10) = 1$.

Problem 6: Key Exchange with Bit Strings

Consider the following key-exchange protocol:

1. Alice chooses $k, r \leftarrow \{0, 1\}^n$ at random, and sends $s := k \oplus r$ to Bob.
2. Bob chooses $t \leftarrow \{0, 1\}^n$ at random and sends $u := s \oplus t$ to Alice.
3. Alice computes $w := u \oplus r$ and sends w to Bob.
4. Alice outputs k and Bob computes $w \oplus t$.

Show that Alice and Bob output the same key. Analyze the security of the scheme (i.e., either prove its security or show a concrete attack).