Exercises for Cryptology 1
Discrete Logarithms

Christian Majenz & Carsten Baum, DTU

April 17, 2023

Exercise 1. (**Computing Discrete Logarithms with smooth group order**)

Let p be a prime and $g, h \in Z_p^*$ where g has order $p-1$. Then given p, g, h the discrete logarithm problem is to find the unique $a \in Z_{p-1}$ such that $g^a = h \bmod p$.

For this exercise, let $p = 31$, $g = 11, h = 5$.

1. Try out all possible choices of a to find the discrete logarithm. For an arbitrary p, how many multiplications modulo p would you have to do (in the worst case) to find a this way?
2. We observe that $p - 1 = 2 \cdot 3 \cdot 5$ and want to use this to simplify the computation of the discrete logarithm. Let $x = (p-1)/2, y = (p-1)/3, z = (p-1)/5$ and consider the elements g^x, g^y, g^z. What do you know about the order of these elements modulo p?
3. We can find the value $a \bmod 2$ by computing the discrete logarithm of h^x for the base g^x. Similarly, we can obtain $a \bmod 3$ from g^y, h^y and $a \bmod 5$ from g^z, h^z. Can you use this to find $a \in Z_{30}$ more efficiently?
4. More generally, assume that $p - 1$ has ℓ prime factors that are all smaller than B. Can you (roughly) say how many multiplications modulo p you have to do, in comparison to the trivial method that tries out all choices of a, to recover the discrete logarithm?

Exercise 2. (**When the Decisional Diffie Hellman Problem is easy**)

Let p be a prime. In the lecture, we considered the DDH problem in the case when $g \in Z_p^*$ was of large prime order q such that $q|p-1$. Now instead, assume that $g \in Z_p^*$ is a generator of the whole group Z_p^*.

Show that, in this case, one can distinguish tuples of the form $(g, g^a, g^b, g^{a \cdot b})$ for $a, b \in Z_{p-1}$ from tuples of the form (g, g^a, g^b, g^c) for $a, b, c \in Z_{p-1}$ with a very good chance. For this, use the observations from the previous exercise and consider what happens if you raise each element in the tuple to $(p-1)/2$.

Exercise 3. (**From Diffie Hellman to Public-Key Encryption**)

Let p be a prime and $g \in Z_p^*$ be of large prime order $q|p-1$. In the Diffie Hellman Key Exchange Protocol, Alice and Bob exchange messages $A = g^a \bmod p, B = g^b \bmod p$ where $a, b \in Z_q$.

1. Assume that Bob publishes the message B as a public key, while he keeps b as his secret key. Alice now encrypts a message $m \in \{0, 1\}$ as follows:
   (a) She chooses $a \in Z_q, r \in Z_q^*$, and computes $c_1 = g^a \bmod p$.
   (b) If $m = 0$ then she sets $c_2 = B^a \bmod p$, otherwise she sets $c_2 = B^a \cdot g^r \bmod p$.
   (c) She lets c_1, c_2 be the ciphertext for Bob.

   Show how Bob can recover the message.
2. Show that this encryption scheme is IND-CPA secure assuming DDH is hard in the group Z_p^* with generator g. Namely, show that if there exists an attacker that wins the IND-CPA security game with probability $P > 1/2$, then we can use it to construct an algorithm that breaks DDH with the same probability.

❓ Exercise 4. (The Pedersen Commitment)

Commitments are an advanced cryptographic primitive. They allow a sender to "commit" to a message m towards the receiver by sending a value c. Having only c (i.e. before m is "opened" to the receiver), the receiver cannot say what message m is contained inside c. At the same time, once c is sent to the receiver then the sender cannot change his mind and open c to another message m' anymore towards the sender. More formally, a commitment scheme consists of two algorithms:

Commit A Com algorithm which, on input m outputs values c, d.

Open An $Open$ algorithm which, on input m, c, d outputs a bit.

It is required that the commitment scheme is binding and hiding:

Binding It should be computationally difficult for a sender to generate values m, m', d, d', c such that $Open(m, c, d) = Open(m', c, d') = 1$ while $m \neq m'$. Note that sender has a free choice of all these values, as long as both messages m, m' are different but open the same commitment c which the sender can also choose.

Hiding Given m_0, m_1 by an adversary, this adversary should not be able to decide if it is a commitment to m_0 or m_1 for an honestly generated commitment c (similar to the IND-CPA property for encryption schemes, where the adversary can pick two "potential" messages but cannot say which one is ultimately encrypted in the ciphertext).

Towards constructing a commitment scheme, let us, as before, assume that p is a prime and $g \in Z_p^*$ is of large prime order $q|p-1$. We assume that p, q, g are public knowledge for everyone. A first attempt for a commitment scheme is the following:

Commit On input $m \in Z_q$, output $c = g^m \bmod p$ and $d = \perp$.

Open On input $m \in Z_q, c \in Z_p^*$ output 1 if $c = g^m \bmod p$ and 0 otherwise.

Show that this construction is insecure because it is not hiding!

A version of this, which bears resemblance to the previous exercises, is actually secure! It is called the *Pedersen Commitment* and it works as follows, assuming an additional $h \in \langle g \rangle$ (i.e. $h = g^a$ for some value a). h is also of prime order q and also publicly known (and fixed for sender and receiver):

Commit On input $m \in Z_q$, sample a random $r \in Z_q$ and output $c = g^m h^r \bmod p$ and $d = r$. The receiver will obtain c while the sender keeps d to itself.

Open On input $m \in Z_q, c \in Z_p^*, d \in Z_q$ output 1 if $c = g^m h^r \bmod p$, otherwise output 0.

1. Assume that neither sender nor receiver know the discrete logarithm of h to the base g modulo p. Then the aforementioned commitment scheme is binding. To prove this, assume for contradiction that there exists a sender algorithm that can, on input p, q, g, h, generate values m, m', c, r, r' such that $g^m h^r \bmod p = g^{m'} h^{r'} \bmod p$ where $m \neq m'$. Then show that you can use this sender algorithm to compute the discrete logarithm of h to base g modulo p!

2. Show that the commitment scheme is also hiding! To do this, you can use that there must exist an $a \in Z_q$ such that $h = g^a \bmod p$. Then you can show that an honestly generated commitment $c = g^m h^r \bmod p$ could have been generated by any other message m' using a certain randomness r'.