COMS BC3262 March 30, 2026

Problem Set 4

Professor: Eysa Lee *Due: April 16, 2026*

Guidelines. Solutions will be graded for correctness and clarity. If you do not know how to solve a problem, you may write **"I don't know how to do this"** to receive 20% credit for any problem not marked as extra credit. In your proofs, justify every step and, if needed, write down definitions/theorems from class or the textbook and that you rely on. When refuting a claim, provide a counterexample and show that it does not satisfy the required definition.

Collaboration. You are allowed to discuss problems with other students in the class. However, you should write up your solutions on your own and may not read or copy the solutions of others. You should try solving each problem on your own before discussing with others. **If you work with others on a problem, you must note with whom you discussed the problem at the beginning of your solution write-up.** For example, "I discussed Problem 1 with Alice and Bob and Problem 2 with Carol." You may not use external resources apart from the textbooks.

Generative AI Policy. The use of generative AI for anything related to assignments is prohibited in this class. **It is always prohibited to input any substantial portion of any assignment into any AI resource.** It is also always forbidden to incorporate any substantive portion of an AI resource's output into your answers, even if you paraphrase it or rewrite it in your own words.

Problem 1 (Discrete Log vs CDH vs DDH) 10 pts

Let $(\mathbb{G}, g, q) \leftarrow \mathcal{G}(1^n)$ be a group generation algorithm that generates a cyclic group $\mathbb{G} = \langle g \rangle$ with generator g of order $|\mathbb{G}| = q$. You can assume that $q \geq 2^n$ for the security parameter n.

- Show that if the DDH assumption holds then this implies that the CDH assumption holds which in turn implies that the Discrete Logarithm assumption holds over this group.
- Show that if the discrete logarithm assumption does *not* hold in this group then the Diffie-Hellman Key Exchange protocol is insecure over this group.

Problem 2 (Playing with ElGamal Ciphertexts) 10 pts

Let $(\mathbb{G}, g, q) \leftarrow \mathcal{G}(1^n)$ be a group generation algorithm that generates a cyclic group $\mathbb{G} = \langle g \rangle$ with generator g of order $|\mathbb{G}| = q$. Recall that the ElGamal encryption scheme has public key $pk = h = g^x$ and $sk = x$. The encryption procedure computes $\mathsf{Enc}(pk, m) = (g^r, h^r \cdot m)$ where $r \leftarrow \mathbb{Z}_q$.

- (Plaintext Multiplication) Show that given a public key pk and any two ElGamal ciphertexts c_1, c_2 encrypting some unknown messages $m_1, m_2 \in \mathbb{G}$ respectively under the public key pk, we can efficiently create a new ciphertext c^* encrypting $m^* = m_1 \cdot m_2$ under pk without needing to know sk, m_1, m_2.
- (Re-randomization) Given a public key pk and an ElGamal ciphertext c encrypting some unknown messages $m \in \mathbb{G}$, show how to create another ciphertext $c' \neq c$ that encrypts the same message m under pk but with fresh independent randomness. In other words c' should have the same distribution as a fresh and independently generated encryption of m under pk.

Problem 3 (Pseudorandom Generators) 10 pts

Let $(\mathbb{G}, g, q) \leftarrow \mathcal{G}(1^n)$ and let $h \leftarrow \mathbb{G}$ be a random group element. Assume that the values $pub = (\mathbb{G}, g, q, h)$ are made public to everyone, including the adversary. Consider the following candidate pseuodrandom generator

$$G(x) = (g^x, h^x)$$

where the seed is a random $x \leftarrow \mathbb{Z}_q$ and the output consists of 2 group elements. (This is slightly different than the one we mentioned in class.)

- Show that this is a secure pseduorandom generator under the DDH assumption, meaning that for a random $x \leftarrow \mathbb{Z}_q$, the output $G(x)$ is indistinguishable from two uniformly random group elements (even to an adversary who is given the public values pub).
- Now consider the case where the public values $pub = (\mathbb{G}, g, q, h)$ are themselves chosen by an adversary, but they must still be chosen from the correct distribution. Show that an adversary can sample pub in a way that allows her to easily break the pseudorandomness property. In particular, the adversary can distinguish $G(x)$ for a random $x \leftarrow \mathbb{Z}_q$ from two truly random group elements.
  (hint: let the adversary choose $h = g^y$ for a random y that she knows.)

(Note: One of the Snowden revelations showed that the NSA tried to use (a more complex variant of) the above trick. They pushed for the standardization of a pseudorandom generator that was a variant of the above and they chose the value *pub*. It is widely believed that they chose this value in a way that would allow them to distinguish the outputs from random, while ensuring that nobody else could. See `https://en.wikipedia.org/wiki/Dual_EC_DRBG`)

Problem 4 (Amplifying a Discrete Log Attack) 10 pts

Fix some cyclic group $\mathbb{G}$ of order q with generator g. Suppose that for this group, there exists a randomized algorithm $\mathcal{A}$ that runs in time T and solves the discrete logarithm problem *on average* with probability ε meaning that

$$\Pr_{x \leftarrow \mathbb{Z}_q}[\mathcal{A}(g^x) = x] \geq \varepsilon$$

Note that this probability is over a random $x \leftarrow \mathbb{Z}_q$ and also any additional randomness used by $\mathcal{A}$. Assume that multiplication and exponentiation in the group can be performed in $O(T)$ time.

- Show that there is also a randomized algorithm $\mathcal{B}$ that runs in time $O(T)$ and solves the discrete logarithm problem *in the worst* case with probability ε, meaning that for *every* $x \in \mathbb{Z}_q$ we have

  $$\Pr[\mathcal{B}(g^x) = x] \geq \varepsilon.$$

  (hint: $\mathcal{B}$ needs to call $\mathcal{A}$ on a random value to have any guarantee that $\mathcal{A}$ succeeds. Think how to add some randomness to g^x to derive a random group element whose discrete log would reveal x.)

- Show that for every ℓ, there is also a randomized algorithm $\mathcal{B}'$ that runs in time $O(T \cdot \ell)$ and solves the discrete logarithm problem *in the worst* case with probability $\delta := 1 - (1 - \varepsilon)^\ell$, meaning that for *every* $x \in \mathbb{Z}_q$ we have

  $$\Pr[\mathcal{B}'(g^x) = x] \geq \delta.$$

  For example if $\mathcal{A}$ succeeds with probability $\varepsilon = 1/2$ then by setting $\ell = 10$ we require $\mathcal{B}'$ to succeed with probability $1 - 1/2^{10} = 1 - 1/1024 > .999$.
  (hint: have $\mathcal{B}'$ run $\mathcal{B}$ many times)

Note: The above shows that if the discrete logarithm problem is even slightly hard in the worst case (there is no randomized algorithm that can solve it for every x with $> .999$ probability) then it is also hard in the average case (no algorithm can solve it for a random x with non-negligible probability).