Please write your hackerrank username somewhere on your problem set.
Written problems are due at the beginning of class on Friday, October 2. Programming problems must be electronically submitted by 10:50am according to the instructions on problem set 1. The submission page for the programming questions is as follows.

`https://www.hackerrank.com/math158-problem-set-3`

Written problems

1. Prove that breaking Diffie-Hellman is at least as hard as breaking ElGamal (with the same parameters p, g). More precisely, show that if Eve has access to an oracle function `DH(A,B)` which takes two numbers $g^a\%p$ and $g^b\%p$ and instantly returns the number $g^{ab}\%p$, then she can (using this oracle) efficiently decipher any messages encrypted with ElGamal.

2. Textbook exercise 2.16 (same in the first edition).

3. We discussed in class why Bob should create a new ephemeral key (denoted k on page 72 of the textbook) each time he enciphers a message to Alice using her public key, otherwise he exposes himself to a known-plaintext attack from Eve. In this problem, we will see why he also should not just perform "small variations" on a previously-used ephemeral key.

   (a) Suppose that Bob previously used an ephemeral key k_1 to send Alice a message m_1, and that Eve knows what this message is (as well as the enciphered version that Bob sent to Alice). Suppose Bob now enciphers another message m_2 to Alice, using the ephemeral key $k_2 = k_1 + 1$. Explain how Eve can efficiently detect that this is how Bob has obtained k_2 from k_1 (without necessarily determining what k_1 is), and efficiently extract the message m_2.

   (b) Suppose instead that Bob obtains k_2 as $42k_1$. Explain how Eve can efficiently detect this, and extract the message m_2.

   (c) Suppose instead that Bob obtains k_2 as $k_1 - 5$. Explain how Eve can efficiently detect this, and extract the message m_2.

   (d) Suppose instead that Bob obtains k_2 as $13k_1 + 2$. Explain how Eve can efficiently detect this, and extract the message m_2.

   Problem 11 generalizes the ideas of this problem somewhat.

4. Recall that the *order of g modulo p* is defined to be the minimum positive number d such that $g^d \equiv 1 \pmod{p}$. It is often denoted $\mathrm{ord}_p(g)$. Prove that if e is any integer, then the orders of g and of g^e are related by the following formula.

$$\mathrm{ord}_p(g^e) \cdot \gcd(e, \mathrm{ord}_p(g)) = \mathrm{ord}_p(g)$$

5. Let p be any prime, and let g be an element of $(\mathbf{Z}/p)^\times$. Let q be another prime number. Show that $\mathrm{ord}_p(g) = q$ if and only if $g \not\equiv 1 \pmod{p}$ and $g^q \equiv 1 \pmod{p}$. Deduce that if g is a primitive root modulo p, and the prime q divides the number $p - 1$, then $g^{(p-1)/q} \pmod{p}$ is an element of order q.

6. Prove that If p, q are two primes such that $p \equiv 1 \pmod{q}$, then there exists an element $g \in \mathbf{Z}/p$ such that $\operatorname{ord}_p(g) = q$.

   Note. The importance of this fact comes from the fact that Diffie-Hellman (and related systems) are most secure when the order of the element g chosen is divisible by a large prime. One way to ensure this is to choose that large prime (q) in advance, then generate a prime p with $p \equiv 1 \pmod{q}$, and then identify the element g (which is now guaranteed to exist, by the choice of p).

7. Use the Babystep-Giantstep algorithm to compute each of the following discrete logarithms. Show your calculations in a table as on page 83 of the textbook.

   (a) $\log_{10}(13)$ for the prime $p = 17$.

   (b) $\log_{15}(16)$ for the prime $p = 37$.

   (c) $\log_{5}(72)$ for the prime $p = 97$.

Programming problems

8. Write a program to solve the discrete logarithm problem for 16-bit primes. This problem serves as a warm-up (and a way to test more efficient algorithms); a fairly naive method should work.

9. Write a program to solve the discrete logarithm problem for 36-bit primes.

10. Write a program which deciphers a message enciphered with ElGamal, given the parameters p, g and your private key.

11. Write a program that deciphers an message from Bob to Alice using ElGamal encryption, given the system parameters p, g, Alice's public key, knowledge of one previous plaintext that Bob has sent to Alice, as well as information about the nature of Bob's poorly chosen "random" number generator. See the problem statement on hackerrank for details.