HW 4 CMSC 456. Morally DUE Sep 30
SOLUTIONS
NOTE- THE HW IS EIGHT PAGES LONG

1. (0 points) READ the syllabus- Content and Policy. READ my NOTES on ciphers and on English. What is your name? What is the day and time of the first midterm?

2. (25 points) Alice and Bob are going to use Diffie-Hellman. Bob wants to save some time so instead of picking a RANDOM $b \in \{\frac{p}{3}, \frac{2p}{3}\}$ he picks a b that is a power of 2 because he thinks that for such b, g^b will be easier to compute. (Alice still picks $a \in \{\frac{p}{3}, \frac{2p}{3}\}$ at random.)

   (a) (10 points) Bob is right! Computing g^b IS easier if b is a power of 2. Explain why.

   (b) (15 points) If Eve knows that Bob is choosing only powers of 2, she can find the shared secret in time $(O(\log p))^c$ for some c. Show how. What is c?

SOLUTION TO PROBLEM TWO

(a) Bob is right! Computing g^b IS easier if b is a power of 2. Explain why.

ANSWER:

Recall that repeated squaring for g^b takes

$$\lfloor \log_2(b) \rfloor + \text{ (number of 1's in } b \text{ in binary)-1}$$

mults. But if b is a power of 2 then there is only one 1 in b in binary. So only $\lfloor \log_2(b) \rfloor$ mults.

(b) Eve can now find the shared secret in time $O(\log p)^c$. Show how. What is c?

ANSWER:

Eve knows that $b \in X = \{2^0, 2^1, \ldots, 2^{\lfloor \log_2 p \rfloor}\}$.

Eve sees p, g, g^a, g^b.

Eve computes g^x for all $x \in X$. Each of these takes $O(\log p)$ mults, and there are $O(\log p)$ elements of X, so thats $O(\log p)^2$ mults. For one of those x you will see that $g^x = g^b$, so Eve finds out what b is. Once Eve knows b, she computes $(g^a)^b = g^{ab}$, so she has the secret. This last step took another $O(\log p)$ mults, so still

$$O(\log p)^2 \text{ mults, so } c = 2.$$

One can be a bit cleverer and get $c = 1$.

END OF SOLUTION TO PROBLEM TWO

GOTO NEXT PAGE

3. (35 points) This is a programming problem. For this problem, you will be writing *two* programs, which are described below. **WARNING:** This problem is FIVE pages long.

   (a) (15 points) Your first program will deal with primality testing. This program will input two lines and will output two lines.

      i. Begin by inputting two lines from standard input. The first line will contain a positive integer $p \geq 7$. The second line will contain a path to a text file containing a bunch of integers, one on each line. You will use the numbers in this file to generate the random numbers used for primality testing (see note below).

      ii. For the first part of this program, you will run the primality test algorithm described in the lecture slides to determine (up to a high degree of certainty) whether p is prime. For each trial, you will generate a random $a \in \{2, \ldots, p-1\}$ and will compute whether $a^p \equiv a \pmod{p}$. If all the trials pass, output "true" to indicate that p is suspected to be prime. If one of the trials fails, output "false" to indicate that p is known not to be prime. You must run EXACTLY $\lfloor \log_2 p \rfloor$ trials, even if some trial fails early.

      iii. (If your program previously outputted "false", then output "false" again and skip this part.) Now, you will use the same primality test algorithm to determine (up to a high degree of certainty) if $(p-1)/2$ is prime. In other words, you will determine if p is a "safe prime". Output "true" or "false" as before on a new line.

GO TO NEXT PAGE FOR MORE INFO ON THIS PROBLEM

NOTE: The primality testing algorithm requires you to compute powers modulo some number. When doing so, you MUST use repeated squaring. You MAY NOT use library code (i.e., no "pow" or "modpow" functions) to do this. If you do not follow these directions, you will LOSE points.

NOTE: Because of how the autograder grades your output, you MUST generate random numbers in the following specific way. If you do not follow these directions carefully, your output may be marked as incorrect. Whenever you need to generate a random integer in the half-open interval $[b, c)$, you must (1) read the next available integer k from the file specified by the input path, and then (2) compute

$$a = (k \bmod (c - b)) + b$$

as your random number. Once you have done this, discard k and move to the next number in the file. You may assume all numbers in the file are $< 2^{63}$.

GO TO NEXT PAGE FOR MORE INFO ON THIS PROBLEM

(b) (15 points) Your second program will deal with finding large "probable prime" numbers. This program will input two lines and will output six lines.

i. Begin by inputting two lines from standard input. The first line will contain a positive integer $5 \leq L \leq 60$. The second line will contain a path to a text file containing a bunch of numbers, as with the previous program.

ii. First, you will do the following. Generate $(L-1)$ bits randomly, and then append a 1-bit to the left, yielding an L-bit number k. If k is determined to be a safe prime (using the same algorithm as in your first program), then output k on the first line. Otherwise, repeat the process until you get a safe prime. Output on the second line how many random k's you generated for this part.

iii. Next, you will do the following. Generate $(L-2)$ bits randomly, and then append a 1-bit to both the left and right, yielding an L-bit odd number k. If k is determined to be a safe prime, then output k on the third line. Otherwise, repeat the process until you get a safe prime. Output on the fourth line how many random k's you generated for this part.

GO TO NEXT PAGE FOR MORE ON THIS PROBLEM

iv. Lastly, you will do the following. Generate $(L-3)$ bits randomly, and then append a 1-bit to the left, yielding an $(L-2)$-bit number m. Then, obtain $k = 6m + 5$, which is roughly L bits long. If k is determined to be a safe prime, then output k on the fifth line. Otherwise, repeat the process until you get a safe prime. Output on the sixth line how many random m's you generated for this part.

NOTE: To generate L bits, take the next integer k from the input file and compute $k \pmod{2^L}$. The bits of this integer are the bits that you want. Discard k and move to the next number in the file as usual. When generating random values, keep in mind that you should only be passing through the file once per run of your program.

(c) (5 points) Run your second program on values $L = 10, 20, 30, 40$. You may generate your random numbers however you want for this part. Record (in the table below) the average of how many tries it takes to obtain a safe prime over 10 trials for each method and value of L. Report this table along with the rest of your homework. Your code should be uploaded separately (see below).

L	Naive Method	Odd Method	$6k$-method
10			
20			
30			
40			

GO TO NEXT PAGE FOR MORE ON THIS PROBLEM

You are free to choose from various programming languages to complete this problem. By default, we support C, C++, Java, Python2/3, and Ruby. Ask on Piazza if you want more options. You will be submitting all code files you used to complete this problem to the Gradescope assignment called "hw04 - problem 3". Since you will probably want to submit multiple files, you should merge all files into a single zip file and submit that zip file to Gradescope. Upon submission, your code will be automatically run on a Linux machine and tested against various test cases to ensure correctness. You are allowed to submit your code as many times as you want.

Regardless of the language you choose, your submission must include TWO bash scripts called `run1` and `run2` (no file extensions) for running your first and second programs respectively. These scripts must begin with the shebang `#!/usr/bin/env bash` on the very first line. These scripts will be run each time the autograder tries to run your code, so add to these files any commands that are needed to run your code. This gives you greater flexibility regarding how you want to organize your code. Additionally, if you are using a non-scripting language such as Java, also upload a bash script called `build`, also with shebang. This script will be called once upon submission to compile your code before execution.

If you have any questions or confusions, or if you encounter any technical difficulties, feel free to ask for help on Piazza.

SOLUTION TO PROBLEM THREE

Omitted

END OF SOLUTION TO PROBLEM THREE

GOTO NEXT PAGE FOR NEXT PROBLEM FINALLY

4. (25 points) Find $A \in \mathbb{N}$ ($\mathbb{N}$ is the nonnegative integers) and $X \subseteq \{0, 1, 2, \ldots, A-1\}$ such that

$$\begin{aligned}\{n \in \mathbb{N} :&(n \not\equiv 0 \pmod 2) \ \wedge \\ &(n \not\equiv 0 \pmod 3) \ \wedge \\ &(n \not\equiv 0 \pmod 5) \ \wedge \\ &(n \not\equiv 0 \pmod 7)\}\end{aligned}$$

$$= \{n \in \mathbb{N} : (\exists k \in \mathbb{N}, i \in X)[n = Ak + i]\}.$$

(Note: IF the problem was about $2, 3$ instead of $2, 3, 5, 7$ then $A = 6$ and $X = \{1, 5\}$.)

(a) (10 points) What is A? Make it as small as possible.

(b) (15 points) List all of the numbers in X that are NOT 1 and NOT prime. Justify your answer.

(Note: IF the problem was about $2, 3$ instead of $2, 3, 5, 7$ then for (a) the answer is 6 and for (b) the answer is $\emptyset$.)

SOLUTION TO PROBLEM FOUR

a) $A = 2 \times 3 \times 5 \times 7 = 210$

b) If $i \in X$ then i cannot be a multiple of 2,3,5, or 7. If i is not prime its only prime factors are all ≥ 11. So we seek all numbers x such that (1) x is a products of primes ≥ 11, and (2) $x < 210$.

We first look at numbers that have 11 as their lowest prime factor:

$11 \times 11 = 121 < 210$

$11 \times 13 = 143 < 210$

$11 \times 17 = 187 < 210$.

$11 \times 19 = 209 < 210$.

Clearly if we multiply 11 by anything ≥ 20 then we'll get a number > 210. Note that also means we can't use, for example 11×11^2.

We now look at numbers that have 13 as their lowest prime factor:

$13 \times 13 = 169 < 210$

$13 \times 17 = 221$ WHOOPS, already too big.

SO the set of NON-primes in X is

$$\{121, 143, 187, 169, 209\}$$

END OF SOLUTION TO PROBLEM FOUR

GOTO NEXT PAGE

5. (15 points) Alice and Bob are going to do Diffie-Hellman with $p = 89$ and $g = 30$.

   (a) (5 points) Alice picks $a = 2$ and Bob picks $b = 5$. What is the shared secret key?

   (b) (5 points) Alice picks $a = 5$ and Bob picks $b = 2$. What is the shared secret key?

   (c) (5 points) You should have gotten the same answer in parts 1 and 2 above. Is this a coincidence? Explain.

SOLUTION TO PROBLEM FIVE

Omitted.

END OF SOLUTION TO PROBLEM FIVE