C&O 487/687: Assignment #4

Due date: November 14, 2023 (Tuesday)

Note: Some questions use randomization to customize to you specifically. Please include your max-8-character UW user id (`g66xu`) at the beginning of your answer so we can look up your custom solution.

Throughout this assignment, you make use of the following facts about number theory:

- Fact 1: If $a, b \in \mathbb{Z}_p^*$ and p is prime, then a and b are squares modulo p if and only if ab is a square modulo p.
- Fact 2: If p is prime, then exactly half the elements in $\mathbb{Z}_p^*$ are squares.
- Fact 3: There exists an efficient algorithm for determining if an element a has a square root modulo p.
- Fact 4: There exists an efficient algorithm for computing square roots modulo p, if it exists.

1. [9 marks] **Hybrid encryption.**

   Recently, the University of Waterloo LEARN server was hacked by a group calling themselves "Really Super Awesome Association of Engineering Students" (RSA-AES), an maniacal organization bent on world domination. The first phase of their evil plan is to ensure that engineering students have better grades and hence get better co-op jobs than math students, so they hacked into the LEARN server and encrypted this CO 487 assignment in hopes of causing you to get a lower mark in this course. Luckily for you, they made some mistakes during encryption because they did not take CO 487.

   Your first task for this assignment is to decrypt the encrypted assignment PDF so that you can actually do the rest of your assignment.[1]

   I managed to capture some logs showing some of the Python code they used to encrypt the assignments, which you might find helpful in figuring out how to break their encryption. This is in the file `a4q1_attacker_log.py` on LEARN.

   You might have to do some number theory operations in Python as you try to solve this question. Here are a few functions that might be helpful:

   - `pow(x, y, z)`: computes $x^y \bmod z$
   - `isprime(x)`: determines whether the integer x is prime; see `https://docs.sympy.org/latest/modules/ntheory.html#sympy.ntheory.primetest.isprime`
   - `nextprime(x)`: returns the next prime number larger than x; see `https://docs.sympy.org/latest/modules/ntheory.html#sympy.ntheory.generate.nextprime`
   - `factorint(x)`: attempts to factor the integer x, and if successful returns the factors and their multiplicities; see documentation at `https://docs.sympy.org/latest/modules/ntheory.html#sympy.ntheory.factor_.factorint`
   - `mod_inverse(x, z)`: computes $x^{-1} \bmod z$
   - `decimal.getcontext().prec = 100; decimal.Decimal(x).sqrt()`: compute high-precision square roots of real numbers

   `pow` is included with Python by default. To get `decimal`, you have to put `import decimal` at the top of the program. To get the rest of the functions, you have to install an additional library called `sympy` (try running `pip3 install sympy`) and then importing those functions into your program using the following source code:

[1] You can also download the decrypted version of (most of) the assignment from LEARN if you want to get started on other questions, but the decrypted version from LEARN is missing a part (worth a few marks) that can only be obtained by decrypting your own customized encrypted assignment.

```
from sympy import mod_inverse
from sympy.ntheory import isprime, nextprime
```

Note that all of the required packages are installed on the UW math Jupyter notebook server `https://jupyter.math.uwaterloo.ca`, but you need to make sure you use the "Python 3 extra" kernel.

(a) [4 marks] By inspecting `a4q1_attacker_log.py`, describe 4 cryptographic mistakes made by the hackers and what you would do differently.

(b) [4 marks] Describe the procedure you used to decrypt the file. In your description, focus on the main cryptographic break you found. Submit any code you write through Crowdmark, as you would a normal assignment – as a PDF or screenshot. We will be reading it, but not executing it.

(c) [1 marks] To prove that you successfully decrypted the file, copy the following random number into your solution: 90182

Please include your max-8-character UW user id (`g66xu`) at the beginning of your answer so we can look up your custom solution.

2. [6 marks] **RSA prime generation**

Note: this is the question 6 from assignment 3 that was moved to assignment 4. Even if you already submitted this to Crowdmark for A3, please submit again for A4.

Alice and her friend Alicia are finding that generating primes for their RSA modulus is quite slow and are looking for ways to speed things up.

(a) [2 marks] Alice tells Alicia there is no need to generate two distinct primes p and q. Instead, she generated a single prime p and used $N = p^2$ as her modulus. Is this a good way to speed things up? Why or why not?

(b) [2 marks] Alicia has a different idea. She tells Alice that they can each generate a prime and then multiply them together to get a shared modulus $N = pq$. Alice doesn't like this idea because she doesn't want Alicia to read her conversations with Bob. Alicia tells her they can just use different exponents. Explain why this is a bad idea and show how an attacker could recover any message m that Bob sends to both Alice and Alicia by using Euclid's algorithm on the public exponents.

(c) [2 marks] Alice agrees to Alicia's idea but still has some reservations so instead of using $N = pq$ as her modulus, she generates another prime r and uses $M = pr$ as her modulus instead. Explain why this is a big mistake.

3. [9 marks] **Diffie–Hellman and discrete logarithms.**

Let G be a cyclic group of prime order q, with generator g; this information is publicly known. Consider the following three computational hardness assumptions:

- **Decisional Diffie–Hellman (DDH):** Given either (g^a, g^b, g^{ab}) or (g^a, g^b, g^c), where a, b, and c are random, distinguish whether you were given a triple of the first form or a triple of the second form.
- **Computational Diffie–Hellman (CDH):** Given g^a and g^b, where a and b are random, compute g^{ab}.
- **Discrete Logarithm Problem (DLP):** Given g^a, where a is random, compute a.

(a) Show that, if the DDH problem is hard, then the CDH problem must also be hard.
Hint: Prove the contrapositive. Assume that there exists an algorithm, $\mathcal{A}$, which can successfully solve CDH, and write an algorithm, $\mathcal{B}$, that has blackbox access to $\mathcal{A}$, which can solve the DDH problem.

(b) Show that, if the CDH problem is hard, then the discrete logarithm problem must also be hard.

Hint: Prove the contrapositive. Assume that there exists an algorithm, $\mathcal{A}$, which can successfully solve DLP, and write an algorithm, $\mathcal{B}$, that has blackbox access to $\mathcal{A}$, which can solve the CDH problem.

(c) It is not the case that DDH is hard in every group. Suppose our adversary has an oracle, O, which tells them whether a given input is a square modulo p (where p is prime). For instance, if $p = 7$, then $O(2) = true$ since $9^2 \equiv 2 \mod 7$, but $O(3) = false$, since 3 is not a square modulo 7. Write an adversary $\mathcal{A}$, with access to O,[2] which can solve DDH in the group $\mathbb{Z}_p^*$ (for prime p). *Hint: Your adversary does not have to succeed with probability 1; it is enough that they succeed with probability $\frac{1}{2} + x$, where x is non-negligible.*

4. [7 marks] **IND-CPA/IND-CCA security and Elgamal encryption.**

In this question, you will analyze the Elgamal cryptosystem from a provable security perspective.

(a) [1 marks] Explain why, in the IND-CPA and IND-CCA security experiments for public key encryption, the adversary is not given access to an encryption oracle.

(b) [1 marks] Write the IND-CPA and IND-CCA security experiments for the Elgamal public key encryption scheme, including all oracles.

(c) [3 marks] Show that Elgamal encryption satisfies IND-CPA security, assuming that the decisional Diffie–Hellman (DDH) problem is hard.

Hint: Prove the contrapositive. Assume that there exists an algorithm, $\mathcal{A}$, which can win the IND-CPA game for Elgamal, and write an algorithm $\mathcal{B}$, that has blackbox access to $\mathcal{A}$, which can solve the DDH problem.

(d) [2 marks] Show that Elgamal does not satisfy IND-CCA security.

Hint: Write an algorithm, $\mathcal{A}$, which can win the IND-CCA game with non-negligible probability.

5. [8 marks] **Backdoored PRG.**

A *pseudorandom generator* (PRG) is a deterministic function that expands a short random seed into a longer random looking sequence. We will construct a pseudorandom generator from a deterministic function f that takes as input a state s_{i-1} and produces a new state s_i and outputs $t_i \in S$ for some set S.

To produce a long pseudorandom sequence of ℓ numbers from set S, start with an initial seed s_0, iterate the function as many times as needed by computing

$$(s_i, t_i) \leftarrow f(s_{i-1})$$

for $i = 1, \ldots, \ell$, and output the final output $t_1, \ldots, t_\ell \in S^\ell$

We will design a PRG based off of elliptic curves. The following information is public:

- A prime number p such that $p \equiv 3 \mod 4$.
- Values A and B such that $4A^3 + 27B^2 \neq 0$ representing an elliptic curve $E : y^2 = x^3 + Ax + B$.
- Two points P, Q over the elliptic curve $E(\mathbb{F}_p)$.

[2] Giving the adversary such an oracle is not as outlandish as it may seem. The Legendre symbol is an efficiently computable function from elementary number theory which returns 1 if the input is a square modulo p, -1 if the input is not a square modulo p, and 0 if the input is divisible by p. So, in the real world, every adversary has access to O!

Our PRG is based on the following function f, defined as follows:

$$f(s_{i-1}) = (s_i, t_i)$$

where

$$r_i = x(s_{i-1}P) \qquad s_i = x(r_iP) \qquad t_i = x(r_iQ)$$

Here, the function x just maps an elliptic curve point R to the x-coordinate of the point R.

(a) [2 marks] For $p = 19$, $A = 2$, and $B = 3$, $P = (1, 14)$, $Q = (3, 13)$ and $s_0 = 2$ produce the output for $\ell = 3$ (Produce the values t_1, t_2, t_3). You may use your own code here to implement the elliptic curve addition formula, or may do it by hand. You will not receive credit for giving t_1, t_2, t_3 without showing work.

(b) [3 marks] It is possible to show that, for well-chosen parameters, this PRG is secure under the assumption that the elliptic curve Diffie–Hellman problem is hard. If, however, the adversary knows a value c such that $P = cQ$, called a backdoor, the PRG is not secure. Suppose the value of t_1 had been published. Explain how, for arbitrary parameters meeting the conditions described, an adversary could use c and t_1 to find s_1, and use this to predict t_2. Justify your response.

Hint: Consider a point $P^* \in E(\mathbb{F}_p)$ with x-coordinate t_1.

(c) [2 marks] For $p = 103, A = 3, B = 4$, $Q = (2, 11)$, and $P = 3Q = (84, 68)$, the value of t_1 was 42. Use your attack from part (b) to find s_1 and predict t_2. For this part you may make use of online calculators and tools to do arithmetic; please indicate which tools or calculators you use.

(d) [1 marks] Suggest a countermeasure to the attack of part (b).

Academic integrity rules

You should make an effort to solve all the problems on your own. You are also welcome to collaborate on assignments with other students in this course. However, solutions must be written up by yourself. If you do collaborate, please acknowledge your collaborators in the write-up for each problem. *If you obtain a solution with help from a book, paper, a website, or any other source, please acknowledge your source. You are not permitted to solicit help from other online bulletin boards, chat groups, newsgroups, or solutions from previous offerings of the course.*

Due date

The assignment is due via Crowdmark by 8:59:59pm on November 14, 2023. Late assignments will not be accepted.