CSE 470 CRYPTOGRAPHY AND COMPUTER SECURITY

FALL 2023

HOMEWORK 1

MERT GÜRŞİMŞİR

1901042646

INTRODUCTION

In cryptography, big prime numbers play a crucial role in ensuring the security of various cryptographic protocols. The security of many widely used encryption algorithms relies on the difficulty of factoring large numbers into their prime components. In particular, the widely adopted RSA algorithm, which is integral to secure communication over the internet, involves the use of large prime numbers for generating public and private keys. The security strength of RSA and other public-key cryptosystems is directly related to the size of the prime numbers involved; larger primes make it exponentially more challenging for adversaries to break the encryption by factoring. The use of big prime numbers adds complexity and computational difficulty to the task of deciphering encrypted data, making it a cornerstone in the protection of sensitive information in digital communication and transactions. As computing power advances, the cryptographic community continually explores larger prime numbers to maintain the resilience of encryption mechanisms against evolving threats.

The security of many widely used encryption algorithms relies on the difficulty of factoring large numbers into their prime components due to the mathematical properties of certain mathematical problems. One of the fundamental problems in number theory is the difficulty of factoring a large composite number into its prime factors, known as the integer factorization problem.

Public-key encryption systems, such as RSA (Rivest–Shamir–Adleman), are based on the asymmetry of this problem. While it's relatively easy to multiply two large prime numbers to obtain a large composite number, the reverse process—finding the prime factors of a large composite number—is computationally difficult, especially as the size of the numbers involved increases.

In RSA, a user generates a pair of keys: a public key, which can be freely distributed, and a private key, which must be kept secret. The security of the system relies on the fact that even with knowledge of the public key (which includes a large composite number), determining the private key (the prime factors of that composite number) is computationally infeasible within a reasonable amount of time, especially as the size of the primes increases.

The difficulty of factoring large numbers is not currently known to have an efficient solution using classical or quantum algorithms. Therefore, the security of encryption systems based on integer factorization rests on the assumption that factoring large numbers remains a computationally expensive task, providing a robust foundation for secure communication and data protection in various applications.

MILLER-RABIN

This primality test is probabilistic test that is discovered by Michael Oser Rabin and Gary Lee Miller. It tests if something is not prime.

Result of the test will be whether the number is composite or probably a prime number. Checks whether a specific property, which is known to hold for prime values, holds for the number under testing.

If test says “this number is prime”, actually it means “this number is probably prime”. We are almost sure but not completely sure.

However if we run the test and it says number is composite, than we are 100% sure this number is not prime.

At the beginning, we assume number is prime.

If we ever arrive at a false statement, that means that our original assumption was wrong. So n is composite.

There are two versions of this algorithm that I have found and I want to show both of them.

First algorithm

STEPS:

1. Find $n-1 = 2^k m$
    a. n: number we are testing
    b. k, m are whole numbers
2. Choose a: $1 < a < n-1$
3. Compute $b_0 = a^m \pmod n$, $b_i = b_{i-1}^2 \pmod n$

Example: Is 53 prime?

1. $n-1 = 2^k m$
    a. $n = 53$
    b. $52 = 2^k m$
        i. $\frac{52}{2^1} = 26$
        ii. $\frac{52}{2^2} = 13$
        iii. $\frac{52}{2^3} = 6.5$
        iv. 6.5 is not whole so we stop here.
        v. We take k as one before which is 2 at 2^2, and we take m as result which is 13
    c. $52 = 2^2.13$
        i. $k = 2$
        ii. $m = 13$
2. $1 < a < n-1$
    a. $1 < a < 52$
    b. Pick a = 2 (doesn't matter, better to choose small number)
3. $b_0 = a^m (\text{mod } n)$, $b_i = b_{i-1}^2$
    a. $b_0 = 2^{13} (\text{mod } 53) = 30 (\text{mod } 53)$
    b. We want to know if b0 is equal to +1 or -1
    c. If b0 equals either of these, it means n is probably prime
    d. Our b0 is 30.
    e. We have to move on the calculate b1.
    f. $b1 = b0^2 (\text{mod} 53) = -1 (\text{mod} 53)$
        i. We ask same question, is b1 equal to +1 or -1?
        ii. However, the implications at this point are different.
        iii. +1 → means that number is composite
        iv. -1 → means that number is prime (probably)
            1. Our case is this.
            2. Conclusion → 53 is probably prime.

What if b1 = 5?

- Then compute b2
    - $b2 = b1^2 (\text{mod } 53) = 25 (\text{mod } 53)$
    - Question is same, is b2 equal to +1 or -1?
    - +1 → composite
    - -1 → probably prime
- After b1, b2, b3… we work the same way.

There are some cases that you never get +1 or -1. If it loops and goes on forever, that means it's composite. That doesn't happen too often.

Second algorithm

- Fermat theorem:
  - $a^{n-1} = 1 \pmod n$ ------> if n is prime
  - We can write this → $a^{n-1} - 1 = 0 \pmod n$
  - If n is prime, n-1 must be even.
  - Difference of squares factorization:
    - $\left(a^{\frac{n-1}{2}} - 1\right)\left(a^{\frac{n-1}{2}} + 1\right) \equiv 0 (\mathrm{mod}\, n)$
    - $\left(a^{\frac{n-1}{4}} - 1\right)\left(a^{\frac{n-1}{4}} + 1\right)\left(a^{\frac{n-1}{2}} + 1\right) \equiv 0 (\mathrm{mod}\, n)$
      - We can do this if (n-1)/2 is even. If not we can stop at the previous step. If (n-1)/4 is still even, we can factorize more…
    - $\left(a^{\frac{n-1}{8}} - 1\right)\left(a^{\frac{n-1}{8}} + 1\right)\left(a^{\frac{n-1}{4}} + 1\right)\left(a^{\frac{n-1}{2}} + 1\right) \equiv 0 (\mathrm{mod}\, n)$
      …
    - $\left(a^{\frac{n-1}{2^k}} - 1\right)\left(a^{\frac{n-1}{2^k}} + 1\right) \ldots \left(a^{\frac{n-1}{2}} + 1\right) \equiv 0 (\mathrm{mod}\, n)$
      1. Now at the end we came here and $(n-1)/2^k$ is odd.
      2. If n is prime (that's our assumption at the beginning and we are looking for contradiction to determine its not prime), then n has to divide at least one of the terms in parantheses because left hand side is equivalent to 0(mod n) so left hand side is divisible by n. This fact is called *Euclid's Lemma* (A positive integer p>1 is a prime number if and only if p | ab implies that p|a or p| b, for all integers a and b).
      3. We check terms if divisible by n. If at least one of them is divisible by n, then our condition is satisfied and nothing is violated. n is probably prime. There is ¾ probability that n is prime.
         a. If we choose another value of a, then our chance to be wrong goes ¼ to 1/16.
         b. If we use 40 values of a, chance of going wrong goes to $(1/4)^{40}$. Small chance to be wrong.
      4. If none are divisible by n, then n is not prime. n is composite for sure.

Note that → 52 = 2 (mod10)

ERATOSTHENES

Ancient Greek method of finding prime numbers in an array or the hundreds chart created by Eratosthenes. This method is used for generating list of prime numbers up to some limit n.

Example:

- We can use this to identify prime numbers up to 100.
- Steps:
    - Cross the 1. bc prime number is any number greater than 1 and only divisible by itself and 1.
    - Circle 2, it is prime.
    - Cross all multiples of 2.
    - Circle 3, it is prime.
    - Cross all multiples of 3.
    - Circle 5, it is prime.
    - Cross all multiples of 5.
    - Circle 7, it is prime.
    - Cross all multiples of 7.

Now that we have found all of the multiples of 2 through 10, all we have left that are not circled are prime numbers. They are prime because they only have factors of 1 and themselves. 11*11 is greater than 100 so don't need to look multiples of it.

We have stopped at the square root of n (100), which is 10 because if we go further for example 11, 11*10 is already includes 10 that we have looked before and 11*11 is greater than 100.

ATKIN

Print all numbers smaller than or equal to the given limit like Eratosthenes.

Compared with the ancient Sieve of Eratosthenes, which marks off multiples of primes, Atkin does some preliminary work and then marks off multiples of squares of primes thus achieving a better theoretical asymptotic complexity. It is created in 2003 by A.O.L. Atkin and Daniel J. Bernstein.

The Sieve of Atkin addresses some of the limitations of the Sieve of Eratosthenes. It utilizes a set of mathematical insights to skip marking certain numbers as composite, reducing the overall number of operations needed. The algorithm involves modulo operations and is optimized for performance.

Step 1:

- Create a results list, filled with 2 and 3
- Create a sieve list with an entry for each positive integer; all entries in this list should initially be marked non-prime

Step 2:

- For each entry number n in the sieve list, with modulo-sixty remainder r:
    - If r is 1, 13, 17, 29, 37, 41, 49, or 53, flip the entry for each possible solution to $4x^2 + y^2 = n$
        - Perform n%12, if result is 1 or 5 then flip the status of number in list
    - ---
    - If r is 7, 19, 31,or 43, flip the entry for each possible solution to $3x^2 + y^2 = n$
        - Perform n%12, if result is 7 then flip the status of number in list
    - ---
    - If r is 12, 23, 47, or 59, flip the entry for each possible solution to $3x^2-y^2 = n$ when x>y
    - Effectively perform n%12, if result is 11 then flip the status of number in list
    - ---
    - If r is something else, ignore it completely

Step 3:

- Start with the lowest number in the sieve list.
- Take the next number in the sieve list, still marked prime.
- Include the number in the results list
- Square the number and mark all multiples of that square as non-prime. Note that the multiples that can be factored by 2, 3, or 5 need not be marked, as these will be ignored in the final enumeration of primes.
- Repeat steps four through seven.