Scribe: Cryptography and Network Security (Week 10, Thursday)

Chandan Kumar

9-Nov-2020

1 Topics Covered

1. Primality Testing
   (a) Prime Number Theorem
   (b) Monte-Carlo Algorithm
   (c) The Problem Composites
   (d) Quadratic Residue
   (e) Legendre Symbol & Jacobi Symbol
   (f) Solovay Strassen Algorithm

2 Primality Testing

2.1 Overview

In RSA algorithm parameter selection requires n = p*q, such that p and q are large prime numbers approximately of 512 bit in size each. So it is necessary to generate large random numbers and check for it primality. Though Agrawal, Kayal and Saxena in 2002 proved that there exist a polynomial time deterministic algorithm for primality testing but still randomized polynomial time Monte Carlo algorithm like Solovay-Strassen algorithm and Miller-Rabin algorithm are widely used technique for primality testing nowadays. These algorithms give an answer in time that is polynomial to $O(logn)$ which is the number of bits required to store n. However, there is a probability that algorithm may claim that n is prime when it is not, in such cases the number is called as pseudo-prime number.

2.2 Prime Number Theorem

This theorem calculates the number of prime numbers $\pi(N)$ which are less than equal to a given number N. It also estimates the probability of choosing a prime number among all the numbers less than equal to N.

$$\pi(N) = \frac{N}{lnN} \tag{1}$$

So, the probability that a chosen random number between 1 and N is prime can be given by $\frac{\frac{N}{lnN}}{N} = \frac{1}{lnN}$.
For a 512 bit large number, we have on average, out of 355 random integers from N of the appropriate size, one will be prime (of course, if we restrict our attention to odd integers, the probability doubles, to about 2/355). This probability shows that parameter generation for RSA is practical and can be easily employed.

2.3 Monte Carlo Algorithm

Monte Carlo Algorithm comes in two flavour, one is a *yes-biased monte carlo* algorithm where as other is *no-biased monte carlo*algorithm. A *yes-biased monte carlo* is an algorithm for a decision problem in which a yes answer is always correct. It has an error probability equal to ϵ, if for any instance in which the answer is "yes", but algorithm gives an answer in "no" with probability at most ϵ. The n0-biased monte carlo has almost reverse functionality than what is discussed for *yes-biased monte carlo*.

2.4 The Problem composites

The problem composites is a decision problem which answers whether a given number n is composite or not without computing the factors of n. The decision version of this problem is represented in Table 1

Problem	**Composites**
Instance	A positive integer n >= 2.
Question	Is n composite?

Table 1: Composite problem

If the output given by monte carlo yes-biased is yes then n is surely composite. However if n is composite then the algorithm outputs yes with a probability of at least 1/2.

2.5 Quadratic Residue

Definition: Suppose p is an odd prime then a is said to be a quadratic residue modulo p if $a \not\equiv 0 \pmod{p}$ and the congruence $y^2 \equiv a \pmod{p}$ has a solution $y \in Z_p$. a is defined to be a quadratic non-residue modulo p if $a \not\equiv 0 \pmod{p}$ and a is not a quadratic residue modulo p.

Let us understand quadratic residue with the help of an example. Consider 11 as the prime number.

$$1^2 = 1$$
$$2^2 = 4$$
$$3^2 = 9$$
$$4^2 = 5$$
$$5^2 = 3$$
$$6^2 = 3$$
$$7^2 = 5$$
$$8^2 = 9$$
$$9^2 = 4$$
$$10^2 = 1$$

So, the quadratic residue from the above example can be listed as {1, 3, 4, 5, 9} and similarly, quadratic non residue can be given by { 2, 6, 7, 8, 10 }. The two important point that should be noted here is,

1. The quadratic residue forms a palindrome. eg { 1, 4, 9, 5, 3, 3, 5, 9, 4, 1}.
2. There are exactly (p-1)/2 quadratic residues.

2.5.1 The QR Problem

The QR decision problem asks given an odd prime number and an integer a, we have to find, is a a quadratic residue modulo p, refer to Table 2.

Problem	**Quadratic Residue**
Instance	A positive integer n >= 2.
Question	Is n composite?

Table 2: QR problem

In order to solve this decision problem, Euler came up with a polynomial time deterministic algorithm discussed below.

2.5.2 Euler's Criterion

Let p be an odd prime. Then a is a quadratic residue modulo p if and only if

$$a^(p-1)/2 \equiv 1(modp) \tag{2}$$

. Also, it can be seen that this statement keeps relevance with fermat's theorem which says that $a^(p-1) \equiv 1(modp)$. Taking square root both sides will produce another result which is, $a^(p-1)/2 \equiv -1(modp)$ is a non quadratic residue. The time complexity for Euler's criterion is $O(logp)^3$.

2.6 Legendre Symbol & Jacobi Symbols

Suppose p is an odd prime. For any integer a, define the Legendre symbol $\left(\frac{a}{p}\right)$ as follows:

$$\left(\frac{a}{p}\right) = \begin{cases} 0 & \text{if } a \equiv 0(\text{mod p}), \\ 1 & \text{if } a \text{ is a quadratic residue modulo p} \\ -1 & \text{if } a \text{ is a quadratic non-residue modulo p} \end{cases}$$

Suppose p is an odd prime, then $\left(\frac{a}{p}\right) \equiv a^{(p-1)/2}(modp)$.

Similarly, Jacobi symbol can be defined as follows, Suppose n is an odd prime positive integer and the prime power factorisation of n is

$$n = \prod_{i=1}^{k} p_i^{e_i} \tag{3}$$

Let a be an integer then jacobi symbol $\left(\frac{a}{p}\right)$ is defined to be,

$$\left(\frac{a}{p}\right) = \prod_{i=1}^{k} \frac{a}{p_i}^{e_i} \tag{4}$$

2.7 Solovay Strassen Algorithm

The decision problem we are solving in Solovay-Strassen Algorithm is "Is n composite?". Note that whenever algorithm says "yes", the answer is correct ans whenever is says "no" then there is an error probability associated with that which is equal to at most 1/2. Please refer to following algorithm table for more clear view.

Algorithm 1: SOLOVAY-STRASSEN ALGORITHM.

```
choose a random integer a such that 1 <= a <= n − 1.
x ← (a/n)
if x == 0 then
    return ("n is composite")
y ← a^((n−1)/2)(modn)
if x ≡ y(modn) then
    return ("n is prime")
```

3 Conclusion

This lecture introduces some definition, symbolic representation, to prove that primality testing can be in polynomial deterministic way.

References

References include class lecture and Furozon book's chapter on RSA.