Prime Numbers: Foundation of Cryptography

Sonal Sarnaik and Basit Ansari

Abstract Prime number plays a very important role in cryptography. There are various types of prime numbers and consists various properties. This paper gives the detail description of the importance of prime numbers in cryptography and algorithms which generates large/strong prime numbers. This paper also focuses on algorithms which find prime factors and tests whether the entered number is prime number or not.

Keywords Prime numbers · Primality testing · Prime number generation

1 Introduction

Exchange of information or data plays a very vital role nowadays. There are various ways through which this data is exchanged. Today's most common way is to communicate through some electronic medium for exam Internet. We perform many important tasks through the internet such as online shopping, online banking, personal data share, etc. So it is very important to make this communication very secure so that an attacker will not be able to get access to the data. Currently, there are various security measures to make this communication secure one of the method is to use Cryptography [1–4]. Cryptography focuses on the concept that "security can be achieved by hiding the data or converting it into some unreadable form," so cryptography is the study of mathematical science which is used to convert the data in some incomprehensible form which gives security to the data [1–3], i.e., cryptography is the art of secret writing [4]. Secret writing is achieved by applying the key to the original data which converts original data into unreadable data (called as Encryption) and unreadable data to original data (called as Decryption) [1–4]. This

S. Sarnaik (✉) · B. Ansari
Marathwada Institute of Technology, Aurangabad, India
e-mail: sonalsarnaik141@gmail.com

B. Ansari
e-mail: basit.ansari@sycet.org

M. U. Bokhari et al. (eds.), *Cyber Security*, Advances in Intelligent Systems and Computing 729, https://doi.org/10.1007/978-981-10-8536-9_31

task is achieved by applying key at sender and key at receiver for encryption and decryption. Two types of keys are mostly used in cryptography, symmetric key and asymmetric key [3, 4]. The intensity of the security will completely rely on the type of key is used. An asymmetric key is much stronger then symmetric key as in asymmetric key two different keys are used one for encryption (encryption key is publically declared) and one for decryption (Decryption key is private only known to receiver) whereas in symmetric key, the same key is used to encryption and for decryption [1–4]. An asymmetric key is also called a public-key cryptosystem [3–5].

There are various algorithms which are used to provide security to the data. Basically, all the concepts of cryptography based on the modular arithmetic concepts, Number systems, Groups rings, Fields, etc [4, 5]. This paper focuses on the concepts of the number system and in which prime number which plays a vital role in Cryptography. If we consider about asymmetric key then the calculation of key completely depends on the prime number and its factors [3–5].

2 Prime Numbers

The numbers which are divisible by itself or by 1 are called as prime numbers and other numbers are called as composite numbers. Examples: 2, 3, 5, 7, 11, 13, 17, 19, etc., are prime numbers which are divisible by only one or by itself and rest of the numbers such as, 2, 4, 9, 10, 12, 14, etc., are composite numbers [4, 6–10]. The securities of cryptographic algorithms are depending on prime numbers and its length. There are various type of prime numbers such as Balance prime, Circular Prime, Long Prime, Mersenne Prime, Minimal Prime, Strong Prime, Palindromic Prime, Permutable Prime, Twin Prime, Unique Prime, Wilson Prime, Regular Prime, Integer Sequence Prime, Higgs Prime, etc. All these types of prime numbers have different properties and it is used in cryptography depending on its properties. The main type of prime numbers which plays a vital role in cryptography are strong prime numbers. A strong prime is a prime number with certain special properties. A number p is a strong prime number if it satisfies following conditions [2–4]:

- p is large prime number
- $p - 1$ must have large prime number factor, say $a1$ $q1$, where $p = a1 * q1 + 1$
- $q1$ must have large prime factors say $a2$ $q2$, where $q1 = a2 * q2 + 1$
- $p + 1$ must have large prime factors say $a3$ $q3$, where $p = a3 * q3 - 1$.

3 Importance of Prime Number in Cryptography

Strong primes are basically used in public-key cryptography to make encryption key and decryption key more secure. Algorithms such as RSA algorithms, Taher and ElGamal algorithms, elliptical curve cryptography, etc., uses strong prime numbers for the encryption key and decryption key generation [2–5]. For example, RSA algorithm uses two types of key, public key (also called as an encryption key) and private key (also called as decryption key) [2–4]. The public key is used to encrypt data; this key is publically declared and known to all [4, 5].

Private key is used to decrypt the data by the receiver, as the name suggests this key is private and no one else can use this key [4, 5]. The main security point in RSA is completely depending on two prime numbers chosen by the sender used for public-key generation and private-key generation [2–5, 11].

RSA algorithm:

Step 1: Sender selects two large prime number p and q
$n = p * q$
Calculate $\varphi(n) = (p - 1) * (q - 1)$

Step 2: Choose d such that it satisfies following condition.

- $\text{Gcd}(d, \varphi(n)) = 1$
- $\text{Max}(p, q)$
- d must be prime no

Step 3: Find e such that it satisfies:

- $e.\ d \equiv 1 \bmod \varphi(n)$
- $e > \log_2(n)$
- $\text{Gcd}(e, \varphi(n)) = 1$

Step 4: $C = M^{e} \bmod n$
Step 5: $M = C^{d} \bmod n$

Where M is original text, C is Ciphertext, p and q both are prime numbers, n is the product of two prime numbers, $\varphi(n)$ is Euler's totient function, e is the Encryption key, d is the Decryption key.

In above demonstration, we can easily understand that if p and q are sufficiently large then complexity other computations will be increased and so decryption key will be very difficult to find out by any third user. This has been shown in [11]. If encryption key and n are known then by applying various factorization methods it is very easy to find prime factors of n, through which decryption key is easy to find [11].

4 Primality Testing

There are various tests which will give us result that whether the entered number is a prime number or not. Methods such as Fermat little's theorem, Miller Rabin, Solovay Strassan [2–6]. An old method of primality checking on the given number is trial and error method, where number "n" will be divided by all possible m from 2 to n, if n gets divided by m then the number is not prime number else number is a prime number [2–5].

Example:
n = 13 then **m** = 2, 3, …12

n mod m =?	13 mod 7 = 6
13 mod 2 = 1	13 mod 8 = 5
13 mod 3 = 1	13 mod 9 = 4
13 mod 4 = 1	13 mod 10 = 3
13 mod 5 = 3	13 mod 11 = 2
13 mod 6 = 1	13 mod 12 = 1

In this example, n is not divisible by all possible m up to $n - 1$. Hence we can say that the given number n is called a prime number. This method is very much time consuming and hence it is not used. Following are the algorithms/Methods which are mostly used to identify whether the entered number is prime or not.

i. **Fermat little theorem**

A different method is used to check primality of a given number is Fermat little theorem. According to Fermat's Little Theorem any number "p" who is prime and any number "a", where $p \nmid a$ (a is not divisible by p), $a^{p-1} = 1 \pmod{p}$ [4].

Example:
$a = 15$, $p = 37$

$\boldsymbol{a^{p-1} = 1}$ **(mod $\boldsymbol{p}$).**
$\boldsymbol{15^{37-1}}$ ***mod 37 = 1*** $1 = 1 \bmod p$
so, 37 is a prime number.

ii. **Solovay–Strassen Algorithm**

This algorithm is based on Monte Carlo algorithm with error probability at most of half. It uses Legendre Jacobi symbol $\left(\frac{a}{n}\right)$, where n is the odd composite number which can be factorized. Instead of factorizing, it can be solved by using some concepts of number theory and some other properties [3, 4].

Properties such as

a. Legendre's Symbol: This property will be applicable only if n is prime number.

$$\left(\frac{a}{n}\right) = 0, \text{ if } m = 0 \mod n$$
$$\left(\frac{a}{n}\right) = 1, \text{ if } x^2 \bmod n = m \text{ but } m \neq 0 \mod n\ (x \text{ is some value})$$
$$\left(\frac{a}{n}\right) = -1, \text{ otherwise}$$

b. Bimultiplicativity:

$$\left(\frac{m1m2}{n}\right) = \left(\frac{m1}{n}\right)\left(\frac{m2}{n}\right) \text{ or } \left(\frac{m}{n1n2}\right) = \left(\frac{m}{n1}\right)\left(\frac{m}{n2}\right)$$

c. Invariance:

$$\left(\frac{m}{n}\right) = \left(\frac{m \bmod n}{n}\right)$$

d. Reciprocity: If, m and n are both odd positive numbers then

$$\left(\frac{m}{n}\right) = -1^{(m-1)(n-1)/4}\left(\frac{n}{m}\right)$$

e. Special Values:

$$\left(\frac{2}{n}\right) = -1^{\left(n^2-1\right)/8}, \left(\frac{1}{n}\right) = 1, \left(\frac{0}{n}\right) = 0$$

f. Euler's Theorem: If n is prime number then for any m,

$$m^{(n-1)/2} = \left(\frac{m}{n}\right) \bmod n$$

Input: Take any odd number "n"
Output: Number is prime of Not

```
Step 1: pick a random integer "a",
        Where a ≥ 1 and a ≤ n − 1.
Step 2: $z = \left(\frac{a}{n}\right)$ —by using Legendre Jacobi Symbol
        if $z = 0$, then write ("Entered number is composite")
```

Step 3:

$$y = a^{(n-1)/2} \bmod n$$

```
If $z \equiv y \bmod n$
Then
Write (“Entered number $n$ is prime”)
Else
Write (“Entered number $n$ is composite”)
```

Example:
Say n = 367

Step 1: **a = 21 where a is 1 < 21 < 366**
Step 2: $x = \left(\frac{a}{n}\right) = \left(\frac{21}{367}\right)$

$$\left(\frac{21}{367}\right) = \left(\frac{7*3}{367}\right) = \left(\frac{7}{367}\right) * \left(\frac{3}{367}\right) --- \text{By Bimultiplicativity propety} \tag{1}$$

$$\begin{aligned}
\left(\frac{7}{367}\right) &= (-1)^{(7-1)*(367-1)/4}\left(\frac{367}{7}\right) --- \text{By Reciprocity property} \\
&= (-1)\left(\frac{367}{7}\right) = (-1)\left(\frac{367 \bmod 7}{7}\right) \\
&= (-1)(-1) \ = 1 -- \text{By Invariance Property and Euler's Theorm} \\
\left(\frac{3}{367}\right) &= (-1)^{(3-1)*(367-1)/4}\left(\frac{367}{3}\right) --- \text{By Bimultiplicativity property} \\
&= (-1)\left(\frac{367}{3}\right) \\
&= (-1)\left(\frac{367\, mod\, 3}{3}\right) --- \text{By Invariance property} \\
\left(\frac{1}{3}\right) &= (-1) -- \text{By Special value property} \\
&= (-1)(1) = (-1)
\end{aligned} \tag{2}$$

Putting above values in Eq. 1,

$$x = \left(\frac{21}{367}\right) = \left(\frac{7 * 3}{367}\right) = \left(\frac{7}{367}\right) * \left(\frac{3}{367}\right) = (1)(-1) = -1$$
$$x = -1$$

Step 3:

$$\begin{aligned} &y = a^{\frac{n-1}{2}} \bmod n \\ &y = 21^{\frac{367-1}{2}} \bmod 367 \\ &y = 366 \quad (\text{Which is equal to} - 1 \bmod 367 = 366), \text{so} \\ &y = -1 \end{aligned} \tag{3}$$

from Eqs. 2 and 3

$$x \equiv y \bmod n$$

Hence, 367 is a prime number.

iii. **Miller–Rabin Algorithm**

```
Input:   Any odd number "n"
Output:  Number is prime of Not
Step 1:  $n - 1 = 2^i j$, where $n$ and $j$ both are odd.
         Pick a random number k, Where, $k \geq 1$ and $k \leq n - 1$.
Step 2:  Calculate $l = ka^j \bmod n$ if $l \equiv 1 \bmod n$ then write("Entered
         number is prime")
Step 3:  for $m = 0$ to $j - 1$
         do $l \equiv -1 \bmod n$
         Then return ("Entered number is prime")
         Else
         $l = l^2 \bmod n$
Step 4:  Write ("Entered number is Composite")
```

Example:
n = 131

```
Step 1: $n - 1 = 131 - 1 = 130 = 2^1 * 65$, So $k = 1$ and $m = 65$. Consider
        $a = 40$ where $1 \leq 40 \leq 130$
Step 2: $b = a^m \bmod n = 40^{65} \bmod 131 = 130$
        $b \equiv 1 \bmod 131$, hence go to next step
Step 3: for i = 0 to 1
            $b \equiv -1 \bmod n$
          $130 \equiv -1 \bmod 131$
          $130 \equiv 130$  ---condition is true hence 131 is prime
```

5 Prime Number Generation Algorithm

If we consider the example of RSA algorithm we can say that security of RSA completely depends on the two prime numbers, but is very difficult to find such strong prime numbers because if a prime number is week then the decryption key will easily break [11]. Similarly, there are various algorithms in cryptography which uses the prime number in the process of key generation. To avoid this difficult to find large or strong prime number, there are various prime number generation algorithms which gives a strong/large prime number as output [4, 12]. Algorithms such as a naive incremental generator, Random search for a prime Product of Primes, Modular search method, Williams–Schmidt algorithm for finding strong primes, Gordon's algorithm for finding strong primes, etc [3–10, 12]. If we discuss Gordon's algorithm for finding strong primes then the following algorithm and its output shows how it produces large and strong prime number from two small prime numbers [12].

```
Input: Two prime numbers $q$, $r$.
Output: a strong prime $p$ is generated.

  Step 1: Pick an integer $j_0$. Calculate and pick the first prime number in
          the sequence of $2 * j * r + 1$, Where, $j = j_0, j_0 + 1, j_0 + 2 \ldots$
          Denote this prime by $s = 2 * j * r + 1$
  Step 2: Calculate $l_0 = 2(q * s - 2 \bmod s)\, q - 1$.
  Step 3: Pick an integer $k_0$. Calculate and discover the first prime number
          in the sequence $l_0 + 2k * s * q$,
          Where $k = k_0, k_0 + 1, k_0 + 2 \ldots$
          Symbolize this prime by $p = l_0 + 2 * k * s * q$.
  Step 4: Write ($p$).
```

Example:

First prime number	Second prime number	Generated prime number
5	7	349
13	23	5641
157	163	557663
1511	1523	186832127
9941	9973	3034530013
10039	10753	1858419679
1435139	1255361	90549882804427
3413857	4281313	11418127264703807
7646137	8378239	27535998464638019

6 Integer Factorization Algorithms

It is easy to find Prime factors of a small number, such as 35 = 7 * 5, but the same task becomes difficult if we try on very large numbers. In public-key cryptography, it is very important to get prime numbers through factorization from a large composite number [4, 5, 8, 13–17]. Various algorithms are there which performs the task of finding prime factors of a large composite number, such as Number Field sieve, Quadratic sieve algorithm, Pollard P-1 algorithm, Pollard's rho algorithms, etc [3–5, 8, 16, 17]. We can get the original odd composite number by multiplying Prime factors with each other. Consider the following example Where, 8633 is an odd composite number and 89, 97 are two prime factors, By multiplying these two factors, we can get the original odd composite number, 89 * 97 = 8633.

Example: **8633 = 89 * 97,** Here **n** 8633, **p** 89 and **q** 97

i. **Pollard's rho algorithm**:

Integer factorization algorithms can be differentiated in two terms, Special-purpose algorithm and general purpose algorithm, Pollard's rho algorithm is an example of special-purpose factoring algorithm, which is used to find small prime factors of a composite integer. It is basically useful to find nontrivial factors [3, 4].

INPUT: Consider any odd composite integer **n**.
OUTPUT: a nontrivial factor *d*.

1. Consider two numbers i and j, where $i = 2$, $j = 2$.
2. For $k = 1, 2, \ldots$ do the following:

```
2.1 Calculate $i$ = $i$^2 + 1 mod $n$,
    $j$ = $j$^2 + 1 mod $n$,
    $j$ = $j$^2 + 1 mod $n$.
2.2 Calculate $d$ = GCD ($i - j$, $n$).
2.3 If $d$ is greater than 1 but less than $n$, Write ($p$) and terminate with
    success. (Second factor can be calculated by using $d1 = (n/d)$)
2.4 If $d$ is equal to $n$ then terminate the algorithm with failure
```

Here, p and q are the smallest prime factors of n. This algorithm uses polynomial function f with integer coefficient, i.e., $f(x) = x2 + c$, Where c can be any value from 1 but not $c \neq 0, -2$ [3–5, 13–17].

Example:

Odd composite number	First factor	Second factor
143	11	13
259	7	37
1927	41	47
391883	67	5849

ii. **Pollard p-1 algorithm**:

This algorithm is an example of special-purpose factoring algorithm to find used to find prime factors p and q from odd composite integer n. It uses smoothness bound concept which is calculated by $\sqrt{n}+1$ [3–5, 13–17].

```
INPUT: Consider odd composite integer $n$.
OUTPUT: A nontrivial factor of $d$.

Step 1. Calculate smoothness bound $B$.
Step 2. Pick random integer s, Where $2 \leq s$ and $s \leq n - 1$,
Compute $d = gcd(s, n)$. If $d \geq 2$ hen write ($d$ is first factor).
Step 3. For each prime $t \leq B$ do the following:
    3.1 Compute $l = \left|\frac{\ln n}{\ln t}\right|$
    3.2 Compute $s \leftarrow s^{tl}$ mod $n$
Step 4. Compute $d = gcd(s - 1, n)$.
Step 5. If $d = 1$ or $d = n$, then terminate the algorithm with failure.
Else, Write ($d$ is the first factor). (Second factor can be calculated by using
$d1 = (n/d)$)
```

Example:

Odd composite number	First factor	Second factor
87	3	29
553	7	79
2253	3	751
112579	103	1093

7 Conclusion

Prime number is a very important concept in cryptography. Because of its various features, it is used in almost all well-known algorithms of cryptography such as RSA, Taher and ElGamal, Diffie–Hell key exchange algorithms, etc. Also, various algorithms are there to generate prime numbers such as Gordon's algorithm for finding strong primes and various algorithms to check its primality. This paper focuses on such algorithms with various examples and its use in various cryptographic techniques.

Book References

1. Menezes B. Network security and cryptography: Cengage Learning, India, 2010, 432
2. Bose R. Information theory, coding and cryptography 2008, Tata Mc Graw hill
3. Menezes AJ, van Oorschot PC, Vanstone SA (2001) Handbook of applied cryptography, CRC Press, London, Oct 1996, 816
4. Stinson DR (2006) Cryptography: theory and practice, 3rd edn. CRC Press, London

Journal References

5. Rivest R, Shamir A, Adleman L (1978) A method for obtaining digital signature and publickey cryptosystem communications. ACM 21:120–126
6. Crandall R, Pomerance C (2001) Prime numbers, a computational perspective. Springer, New York
7. Joye M, Paillier P, Vaudenay S (2000) Efficient generation of prime numbers?, Springer-Verlag, 1965:34–354
8. Rivest RL, Silvermany RD. Are strong primes needed for RSA?
9. Agrawal M, Kayal N, Saxena N. Primes is in p
10. Wagsta SS Jr (2014) Is there a shortage of primes for cryptography?, 2(IX), Sep 2014, IJARET
11. Sarnaik S, Gadekar D, Gaikwad U. An overview to integer factorization and RSA in cryptography
12. Saouter Y. A (1995) new method for the generation of strong prime numbers, RR-2657, INRIA

13. Galbraith SD (2012) Towards a rigorous analysis of Pollard Rho. Mathematics of public key cryptography. Cambridge University Press, Cambridge, pp 272–273, ISBN 9781107013926
14. Yan Y (2008) Integer factorization attacks. Cryptanalytic attacks on RSA, Springer-Verlag, US, 255
15. Abubakar A, Jabaka S, Tijjani BI (2014) Cryptanalytic attacks on Rivest, Shamir, and Adleman (RSA) cryptosystem: issues and challenges, JATIT, Mar 2014, 61(1):37–43
16. Hawana B (2013) An overview and cryptographic challenges of RSA. IJERMT
17. Chalurkar SN, Khochare N, Meshram BB (2011) Survey on modular attack on RSA algorithm, IJCEM, Vol 14, Oct 2011, 106–110