PCLA: A New Public-key Cryptosystem Based on Logarithmic Approach

Archana Raghuvamshi[1], Prof.P.Premchand [2], P. Venkateswara Rao [3]

[1]Department of Computer Science, Adikavi Nannaya University
Rajahmundry-533105, Andhra Pradesh, India

[2]Department of Computer Science and Engineering, University College of Engineering
Osmania University, Hyderabad-500007, Andhra Pradesh, India

[3]Department of Computer Science, Adikavi Nannaya University
Rajahmundry-533105, Andhra Pradesh, India

Abstract

A public key cryptosystem is an asymmetric cryptosystem where the key consist of a public key and a private key. As the public key is public (known to all), can be used to encrypt the messages. The private key, kept as secret, can be used to decrypt the messages by the owner. We describe a new public key cryptosystem based on logarithmic approach. PCLA with its simplicity, it's very easier to create keys and can be use securely with low memory requirements. PCLA encryption and decryption use a mixing system suggested by logarithmic approach combined with a clustering principle based on elementary mathematical theory. The security of the PCLA cryptosystem comes from the interaction of the logarithmic mixing system with the independence of relatively prime integer's p and q.

Keyword: *Public Key Cryptosystem, Encryption, Decryption, Logarithmic Approach*

1. Introduction

A cryptosystem refers to a suite of algorithms needed to implement a particular form of Encryption and Decryption. Typically, a cryptosystem consists of three algorithms: one for *key* generation, one for encryption, and one for decryption. The term *cipher* (sometimes *cypher*) is often used to refer to a pair of algorithms, one for encryption and one for decryption. Therefore, the term "cryptosystem" is most often used when the key generation algorithm is important. For this reason, the term "cryptosystem" is commonly used to refer to *public key* techniques; however both "cipher" and "cryptosystem" are used for *symmetric* key techniques.

There has been considerable interest in the creation of efficient and computationally inexpensive public key cryptosystems since Diffie and Hellman [1] explained how such systems could be created using one-way functions. Currently, the most widely used public key system is RSA, which was created by Rivest, Shamir and Adelman in 1978 [5] and is based on the difficulty of factoring large numbers.

In this paper we describe a new public key cryptosystem based on logarithmic approach. The Security of this new public key cryptosystem comes from the interaction of the Logarithmic mixing system with independence of p and q values. The paper is organized as follows, Section 2 describes the background of the cryptosystem, Section 3 explains the description of a proposed PCLA algorithm, Section 4 describes our implementaion work, Section 5 explaines the results of our proposed work.

2. Background

Cryptography is about communication in the presence of an adversary. It encompasses many problems (encryption, authentication, key distribution to name a few). The field of modern cryptography provides a theoretical foundation based on which we may understand what exactly these problems are, how to evaluate protocols that purport to solve them, and how to build protocols in whose security we can have confidence.

The idea of a Public Key Cryptosystem (PKC) was proposed by Diffie and Hellman in their pioneering paper [2] in 1976 who, influenced by Ralph Merkle's work on public-key distribution, disclosed a method of public-key agreement. This method of key exchange, which uses exponentiation in a finite field, came to be known as Diffie–Hellman key exchange. This was the first published practical method for establishing a shared secret-key over an authenticated (but not private) communications channel without using a prior shared secret. Merkle's public-key-agreement technique became known as Merkle's Puzzles, and was invented in 1974 and published in 1978. In 1997, it was publicly disclosed that asymmetric key algorithms were developed by James H. Ellis, Clifford Cocks, and Malcolm Williamson at the Government Communications Headquarters (GCHQ) in the UK in 1973.[3] The researchers independently developed Diffie–Hellman key exchange and a special case of RSA.

The GCHQ cryptographers referred to the technique as "non-secret encryption". This work was named an IEEE Milestone in 2010[4]. A generalization of Cocks's scheme was independently invented in 1977 by Rivest, Shamir and Adleman, all then at MIT. The later authors published

their work in 1978, and the algorithm appropriately came to be known as RSA. RSA uses exponentiation modulo a product of two large primes to encrypt and decrypt, performing both public key encryption and public key digital signature, and its security is connected to the presumed difficulty of factoring large integers, a problem for which there is no known efficient (i.e., practicably fast) general technique. Though many researchers have tried evolutionary computing approaches for designing a cryptosystem, it was rare to see in the literature that researchers used any basic logorithmic approach for designing a public key crypto system.Hence it is encourage to desing the new cypto sytem based on logorithmic approach.

The setup for a public-key cryptosystem is of a network of clients c_1, c_2….c_n rather than an single pair of clients. Each client c *in* the network has a pair of keys < *Pc; Sc* > associated with him, the *public key Pc* which is published under the clients name in a "public directory" accessible for everyone to read, and the private-key *Sc* which is known only to *c*. The pairs of keys are generated by running a *key-generation* algorithm. To send a secret message *M* to *c* everyone in the network clients the *same* exact method, which involves looking up *Pc*, computing $E_{Pc}(M)$ where *E* is a public encryption algorithm, and sending the resulting cipher text *C* to *c*. Upon receiving cipher text *C*, client *c* can decrypt by looking up his private key *Sc* and computing $D_{Sc}(C)$ where *D* is a public decryption algorithm. Clearly, for this to work we need that $D_{Sc}(E_{Pc}(M)) = M$. A particular PKC is thus defined by a triplet of public algorithms (*G;E;D*), the key generation, encryption, and decryption algorithms.

Public key cryptography has 2 main branches:

- *Public key encryption*: This is used to preserve *confidentiality* of a message.In this branch, a message is encryted with a public key of a recipient so that only owner of a mathing private key can decrypt the message.

- *Digital signatures*: This is used for *authenticity* of a message i.e, the part of the message has not been tampered with.Here in this brach, the message is singed by the senders private key so that those (recipent's) having the public key of a sender can know that the message is singed by the owner of the private key

3. Description of a PCLA Algorithm

Notation: A PCLA Cryptosystem depends on three integer parameters (n, p, q). Note that p and q need not be prime, but we will assume that gcd (p, q) = 1 and q will always be considerably larger than p. n is base of logarithm. It could be any one of (e, 2, and 10).

Operation: The PCLA algorithm involves three steps: *key generation*, *encryption* and *decryption.*

Key generation: PCLA involves a **public key** and a **private key.** The public key can be known to everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted using the private key.

The keys for the PCLA algorithm are generated the following way: We choose any two distinct integer numbers *p* and *q*. For security purposes, the integers *p* and *q* should be chosen at random such that gcd(p, q) = 1, and q will always be considerably larger than p. Next we choose n, such that n Є (e, 2, 10)} and *n* is used as the base for both the public and private keys. We compute $a=\log_n(p)$, $b=\log_n(q)$ and finally we compute $e=\log_p(q)$, where $\log_p(q)=b/a$.Now *e* is released as the public key exponent. Here *e* is real number. On a typical computer system, a 'double precision' (64-bit) binary floating-point number has a coefficient of 53 bits (one of which is implied), an exponent of 11 bits, and one sign bit. Next we determine $d = \log_q(p)$, where $\log_q(p)=a/b$. *d* is kept as the private key exponent where *d* is also a real number.

The **public key** consists of the public (or encryption) exponent *e*. The **private key** consists of the private (or decryption) exponent *d* which must be kept secret.

Encryption

Alice transmits her public key *e* to Bob and keeps the private key secret. Bob then wishes to send message **M** $(M_1,M_2,….M_N)$ to Alice.

He first turns **M** into an integer m $(m1,m2….m_N)$ such that $0 < m_i < 255$ where i=1,2,…N by using an Extended ASCII code(Padding Scheme). He then computes the cipher text *c* corresponding to each m_i.

$$c = m^e$$

Note that here cipher text *c* will be in floating point representation.

Decryption

Alice can recover *m* from *c* by using her private key exponent *d* via computing.

$$m = round(c^{d})$$

Given *m*, she can recover the original message **M** by reversing the padding scheme.

(In practice, there are more efficient software for calculating c^d (Ex: MATLAB)).

4. Implementation

A New Public-key Cryptosystem based on Logarithmic Approach is implemented in JAVA. The new public-key algorithm starts from generating the public key 'e' and private key 'd'. Our algorithm starts by converting the plaintext by using generated public key into cipher text. And then in the later steps our program converts the cipher text into plaintext by using the generated private key.

Encryption

The certain restrictions are defined on the encryption algorithm:

1. Only the extended *ASCII* code is encrypted.
2. Base n should be one of the (e, 2, 10), where *n* is used as the base for both the encryption and decryption process.
3. Plain text has not more than 20 character length.

Function for generating of the public and private key is follows:

```
/*Generator of Public and Private keys*/
/*Public key→e,
  Private key→d,
  p, q→ any two integer numbers ∋ gcd (q ,q)=1 and p<q,
  n→base such that n ∈ (e, 2, 10),
  a=log_n(p) and b=log_n(q) */
/*public key*/
 e=log_p (q), where log_p (q)=b/a
/*private key*/
 d = log_q (p), where log_q (p)=a/b.
```

The following steps describe the complete steps in PCLA Algorithm:

1. Choose two distinct integer numbers *p* and *q*.
   - For security purposes, the integers *p* and *q* should be chosen at random such that gcd(p, q) = 1, and q will always be considerably larger than p.
2. Choose n, such that n ∈ (e, 2, 10)}
   - *n* is used as the base for both the public and private keys
3. Compute $a=\log_n(p)$ and $b=\log_n(q)$.
4. Compute $e=\log_p(q)$, where $\log_p(q)=b/a$.
   - *e* is released as the public key exponent.
   - *e* is real number. On a typical computer system, a 'double precision' (64-bit) binary floating-point number has a coefficient of 53 bits (one of which is implied), an exponent of 11 bits, and one sign bit.
5. Determine $d = \log_q(p)$, where $\log_q(p)=a/b$.
   - *d* is kept as the private key exponent.
   - *d* is real number.
6. Bob then computes the cipher text *c* corresponding to each m_i
   Such that $c = m^e$
7. Alice can recover *m* from c by using her private key exponent d via computing, such that $m = round(c^d)$

5. Results and Discussion

Here is an example of PCLA encryption and decryption. The parameters used here are artificially small, but one can also use Open SSL to generate and examine a real key pair.

1. Choose two distinct interger numbers, such as $p = 5$, $q = 104729$.
2. Choose $n = 2$
3. Compute $a=\log_2(5)$
   $=2.3219$,
   And $b=\log_2(104729)$
   $=16.6763$.
4. Compute
   $e=\log_5(104729)$,
   Where $\log_5(104729)=b/a$
   $=(16.6763)/(2.3219) =7.1822$.
   Let $e=7.1822$ is released as the public key exponent.
5. Determine $d = \log_{104729}(5)$, where $\log_{104729}(5) = a/b$
   $= (2.3219)/(16.6763) =0.1392$.
   Let $d =0.1392$is kept as the private key exponent.

The **public key** is (*e*=7.1822). For a padded plaintext message *m*, the encryption function is $m^{7.1822}$.

The **private key** is (*d* =0.1392). For an encrypted ciphertext *c*, the decryption function is $c^{0.1392}$.

***The Encryption is done as follows*:**

Each padded plaintext message m is encrypted by using e=7.1822,

Table I
EXAMPLE OF A PCLA ENCRYPTION

Plaintext	Encryption [Public Key (e=7.1822)]	cipher text
255	$255^{7.1822}$	1.9242e+017
197	$197^{7.1822}$	3.0152e+016

The Decryption is done as follows:

Each number of the cipher is multiplied through the same number for d=0.1392 times and the result of this operation is plain text.

Table II
EXAMPLE OF A PCLA DECRYPTION

Cipher text	Decryption[Private key(d=0.1392)	plain text
1.9242e+017	ROUND([1.9242e+017)^0.1392]	255
3.0152e+016	ROUND([3.0152e+017)^0.1392]	197

In real life situations the random numbers selected would be much larger; given *e*, also from the public key; we could then compute *d* and so acquire the private key.

6. Conclusion

This paper presents new public key cryptosystem based on logarithmic approach. By selecting any of the predefined base and any two integer numbers p & q with some constraint we can generate public key 'e' and private key'd'. Here we use public key to encrypt the message and private key to decrypt the cipher text. Many researchers have been given various public key algorithms based on different techniques.

In this paper unlike the others we have proposed a new technique for public key cryptography by using some simple calculations. A working example has also been shown. We have imposed some restrictions on encryption part of our algorithm. Decryption is straight forward. Presented results show performance of used methods for PCLA algorithm for the plaintext size of 20 characters. These results are only approximation of public key because, the results can by understand as good, but generalized next carefully.

Appendix

```
/* Implementation of PCLA Algorithm */

/*
assert Math.log(Math.E) == 1
assert Math.log10(10000) == 4
def logn(base, val) { Math.log(val)/Math.log(base) }
assert logn(2, 1024) == 10
*/

import java.io.*;
import java.lang.*;
import java.math.*;

public class PCLA
{
public static void main(String args[])
{
            int p,q;
            double c,m;
            try
            {
          System.out.println("Enter two no p,q
                        ( q > p and gcd(p,q)=1 ) : ");
          BufferedReader br=new BufferedReader
                        (new InputStreamReader(System.in));
          String str=br.readLine();
          p=Integer.parseInt(str);
          str=br.readLine();
          q=Integer.parseInt(str);
          System.out.println("Enter plain text : ");
         str=br.readLine();
          m=Integer.parseInt(str);
          //System.out.println(p+" "+q);
          //e=2.718281828;
          double a=Math.log(p);
          double b=Math.log(q);
          System.out.println(a+" "+b);
         double e=b/a;
          double d=a/b;
          System.out.println(e+" "+d);
          //m=5;
          c=Math.pow(m,e);
          m=Math.round(Math.pow(c,d));
         System.out.println(" plain text :"+m);
          System.out.println(" Cipher text :"+c);
         System.out.println(" plain text :"+m);
    }

        catch(IOException e1){}
}
}
```

Acknowledgments

The research reported in this paper is the result of a team effort that been started shortly in our university. Dozens of researchers from all over the world published many papers on this subject, and we have greatly benefited from their ingenious ideas and beautiful insights. They are too numerous to list here, but we are gratefully acknowledge the contribution of all of them.

References

[1] R.J. McEliece, A public-key cryptosystem based on algebraic coding theory, JPL Pasadena, DSN Progress Reports 42-44 (1978), 114-116.

[2] W. Di±e and M. E. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, IT-22:644-654, November 1976.

[3] GCHQ website: http://www.gchq.gov.uk/history/pke.html

[4] "List of IEEE Milestones". *IEEE Global History Network*. IEEE. Retrieved 4 August 2011.

[5] Rabin, M.O.: Digital Signatures and Public-Key Encryptions as Intractable as Factorization,MIT, Technical Report, MIT/LCS/TR-212 (1979).
[6] Lecture Notes on Cryptography, by Shafi Goldwasser et al.
[7] A.Hosseinzadeh Namin, "Elliptic Curve Cryptography " , university of Windsor , April 2005
[8] Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, "Handbook of Applied Cryptography", CRC Press, 1997
[9] Burt Kaliski, "The Mathematics of the RSA Public-Key Cryptosystem", RSA Laboratories. April 9,2006.

Archana Raghuvamshi received her B.Sc Degree with mathematics Statistics and Computer Science from Osamani University,Hyderabad, India in 1997 and Masters of Computer Applications from Osmania University,Hyderabad,India in 2000.She received her M.Tech in Computer science and engineering from U.C.O.E., Osmania University,Hyderabad, India in 2006.During the academic years 2000-2004 she was on the faculty of the Department of Informatics and from the June'2006 to Dec'2006 she worked as a Assistant Professor in CVR College of Engineering,Hyderabad.Later from Dec'2006 to Jan'2009 worked as a Faculty Member in Department of CSE in ICFAITECH, Hyderabad.From Jan' 2009 to Aug' 2009 she has done her course work in IITM as part of the PhD which is registered in IITH.Since Oct' 2009 she is working as an Assistant Professor in Departement of Computer Science in Adikavi Nannaya university,Rajahmundry,Hyderabad.She has publised two research papers in IEEE Digital library and one in international conference proceedings.She is fellow of International Association of Engineers.Her current research includes information Security,Cryptography and Cryptanlysis.

Prof.P.Premchand received his B.Sc(Engg.) in Electrical Engineering from RIT Jamshedpur,India.He received his M.E in Computer Engineering from Andhra Univeresity, Visakhapatnam,India.He received his PhD in Computer Science and System Engineering from Andhra University,Visakhapatnam,India.He worked as a Lecturer in Dept. of Computer Science & Systems Engineering in Andhra University from 1985-1991. He worked as a Associate Professor in Dept. of Computer Science Engineering Osmania University from 1995-1998.He was as a Director in AICTE, New Delhi 1998-1999.He was on the Designation of Additional Controller of Examinations in Professional wing, Osmania University from 2001-2003. Now he is working as a Professor in Dept. of Computer Science Engineering, Osmania University from 1990. He worked as a Head, in Dept. of Computer Science Engineering Osmania University from 2005-2007.He was a Incharge Director 21stCentury Gurukulam (Pre-MSIT), Osmania University, Hyderabad from 2006-2007.He is a Chairman BOS in Faculty of Engineering since 2007.He has guided many M.Tech and PhD students for their award of Degree.His area of research includes Digital Image Processing, Crytptography, Network Security etc.

Pallipamu Venkateswara Rao received his B.Tech Degree with Computer Science and Engineering from Andhra University, Visakhapatnam, Andhra Pradesh, India in 2001 and M.Tech in 2001 and M.Tech in Information Technology from Andhra University, Visakhapatnam, Andhra Pradesh, India in 2004.He has total 8 years of teaching experience. Presently he is working as an Associate Professor in Department of Computer Science in Adikavi Nannaya university, Rajahmundry, Andhra Pradesh, India. He has published two research papers in IEEE Digital library and one in international conference proceedings. He is a life time member of Indian Society for Technical Education(ISTE) and a fellow of International Association of Engineers(IAENG).His current research includes Information Security, Cryptography and Network Security.