Encryption Algorithm Based on Position

Mrs. Anagha Vivek Dudgikar
MM College Of Engineering, Pune – India

Abstract

Secure data communication is the main aspect of this study. The algorithm developed as a part of this study, is independent of the keys, due to which higher security means can be achieved. This algorithm is easy to implement on any machine. The cipher characters derived never get repeated for same plain text characters. The complete file also can be encrypted by using this technique.

Keywords: Position Based, cipher text, encryption, decryption.

1. Introduction

Cryptography in simple words is an `art or science of protecting data'. But it's an art which is yet to achieve perfection. The major challenge being faced by it is the intelligence and computational capability of the hacker.[2]All these years the scientists have been working on increasing the computational complexity, but the recent concept of position based cryptography has added a complete new dimension to this field. The strength of this cryptographic technique is in different character replacement for the same character in plain text, which helps in providing a highly secure data transfer. [5]

2. Proposed System

The purpose of the intended study is to develop an encryption and decryption algorithm. Algorithms are developed in such a way that they would be easy to understand & implement.

The process of encryption is kept simple:

- For given plain text characters, numbers starting from zero (0) onwards get assigned for Position representation.
- Once Position based encryption Algorithm gets applied, plain text characters get replaced with cipher text by using Table-I & Table-II mentioned below.
- The algorithm takes care of replacing same plain text characters, with different cipher characters as a part of conversion which builds required complexity.
- We need to refer Rows in tables as position of characters whereas column represents character itself.

- The cipher text letter is derived as an intersection of the row & column for the chosen character in consideration.
- If original character is ‘blank space’, the counter for position is reset to 0.
- The special characters get handled by using Table-II.

Decryption is also equally simple.

- For derived cipher text character, intersection of row and column is checked and Original plain text character is obtained.

Distinct features of developed algorithm:

- Without knowing rules & tables, it is difficult to decode.
- Time & Space complexity O (n^2) where n = 26.
- Case sensitivity is maintained

3. Example

Consider original text sentence as: *‘Several methods can be used to encrypt data’.*

Let’s take first two words of above sentence for example purpose.

Positions will be represented as below

S	e	v	e	r	a	l		m	e	t	h	o	d	s
0	1	2	3	4	5	6		0	1	2	3	4	5	6

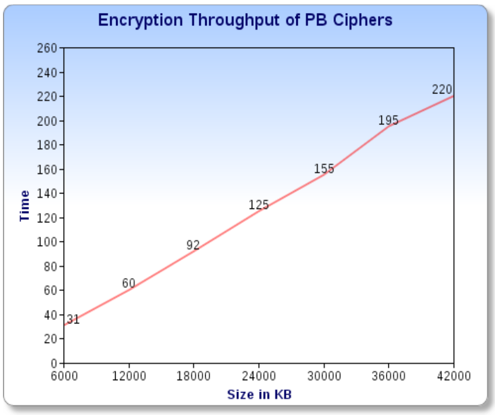
Encryption:
A. Several
1. char[x] = [y+25-f]%26
   char[x] = [0+25-18]%26 = 7
   char[x] = ’H’
2. Char[x] = [y+25-f] %26
   Char[x] = [1+25- 4] %26 = 22
   Char[x] = ’w’
3. Char[x] = ’g’
4. Char[x] = ’y’
5. Char[x] = ’m’
6. Char[x] = ’e’
7. Char[x] = ’u’

B. methods:-
1. Char[x] = [0+25-12] %26 = 13 = ’n’
2. char[x] = [1+25-4] %26 = 22 = ’w’
3. char[x] = [2+25-19] %26 = 8 = ’i’
4. Char[x] = [3+25-7] %26 = 21 = ’v’
5. Char[x] = [4+25-14] %26 = 15 = ’p’
6. Char[x] = [5+25-3] %26 = 1 = ’b’
7. Char[x] = [6+25-18] %26 = 13 = ’n’

Decryption:
A. Cipher text : ‘Hwgymeu’
1. Char [pos] = [y+25-x] %26
   Char [pos] = [0+25-7] %26 = 18
   Char[x] = ’S’
2. Char pos] = [y+25-x] %26
   Char [pos] = [1+25-22] %26 = 4
   Char [x] = ’e’
3. Char [pos] = [2+25-6] %26 = 21 = ’v’
4. Char [pos] = [3+25-24] %26 = 4 = ’e’
5. Char [pos] = ’r’
6. char [pos] = ’a’
7. Char [pos] = ’l’

B. nwivpbn:
1. Char[x] = [0+25-13] %26 = 12 = ’m’
2. char[x] = [1+25-22] %26 = 4 = ’e’
3. Char[x] = [2+25-8] %26 = 19 =’t’
4. Char[x] = [3+25-21] %26 = 7 = ’h’
5. Char[x] = [4+25-15] %26 = 14 = ’o’
6. char[x] = [5+25-1] %26 = 3 =’d’
7. Char[x] = [6+25-13] %26 = 18 =’s’

4. Performance



The throughput was measured while varying the file size from 5MB to 40 MB, since most file system accesses are within this range in actual environment.

5. Conclusion And Future Work

The purpose of this study is to provide a secured & computation efficient storage solution for cryptography environment. The implementation of the developed algorithms is simpler as system requirements (both hardware and software) are not significant.

Numbers and Picture files are not in scope of this work. If the user requirement is specifically for Key based algorithm, above algorithm would needs to be suitably modified.

6. References

[1] Mandeep Kaur, Manish Mahajan
Using encryption Algorithms to enhance the Data Security in Cloud Computing
Volume 01 – No.12, Issue: 03 Page 56-59
International Journal of Communication and Computer Technologies

[2]Sakthi Vignesh S. Sudharssun K.J.Jegadish Kumar
Limitations of Quantum & the Versatility of Classical Cryptography: A Comparative Study Environmental and Computer Science, 2009. ICECS'09. Second International Conference on 28-30 Dec. 2009 Page(s):333 - 337 E-ISBN: 978-1-4244-5591-1

[3] Uma Somani, Kanika Lakhani, Manish Mundra Implementing digital signature with RSA Encryption Algorithm to enhance the data security of cloud in Cloud Computing , 2010.

[4]www.schneier.com/blog/archives/2012/11/encryption_in_c.html

[5] W. Diffie and M.E. Hellman, “New Directions in Cryptography,”
IEEE Transactions on Information Theory, v. IT–22, n. 6, Nov 1976,
pp. 644–654.

[6] W.F. Ehrsam, C.H.W. Meyer, and W.L. Tuchman, “A Cryptographic
Key Management Scheme for Implementing the Data Encryption
Standard,” IBM Systems Journal, v. 17, n. 2, 1978, pp. 106–125

[7] Information Security Management Handbook by Harold F. Tipton, Micki Krause

[8] S J Shepherd. Public Key Stream Ciphers Published in Security and Cryptography Applications to Radio Systems, IEE Colloquium on Date of Conference: 1994
Page(s): 10/1 - 10/7

Table I

	a	b	c	d	e	f	g	h	I	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
0	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l	k	j	i	h	g	f	e	d	c	b	a
1	a	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l	k	j	i	h	g	f	e	d	c	b
2	b	a	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l	k	j	i	h	g	f	e	d	c
3	c	b	a	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l	k	j	i	h	g	f	e	d
4	d	c	b	a	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l	k	j	i	h	g	f	e
5	e	d	c	b	a	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l	k	j	i	h	g	f
6	f	e	d	c	b	a	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l	k	j	i	h	g
7	g	f	e	d	c	b	a	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l	k	j	i	h
8	h	g	f	e	d	c	b	a	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l	k	j	i
9	I	h	g	f	e	d	c	b	a	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l	k	j
10	j	I	h	g	f	e	d	c	b	a	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l	k
11	k	j	I	h	g	f	e	d	c	b	a	z	y	x	w	v	u	t	s	r	q	p	o	n	m	l
12	l	k	j	I	h	g	f	e	d	c	b	a	z	y	x	w	v	u	t	s	r	q	p	o	n	m
13	m	l	k	j	I	h	g	f	e	d	c	b	a	z	y	x	w	v	u	t	s	r	q	p	o	n
14	n	m	l	k	j	I	h	g	f	e	d	c	b	a	z	y	x	w	v	u	t	s	r	q	p	o
15	o	n	m	l	k	j	I	h	g	f	e	d	c	b	a	z	y	x	w	v	u	t	s	r	q	p
16	p	o	n	m	l	k	j	I	h	g	f	e	d	c	b	a	z	y	x	w	v	u	t	s	r	q
17	q	p	o	n	m	l	k	j	I	h	g	f	e	d	c	b	a	z	y	x	w	v	u	t	s	r
18	r	q	p	o	n	m	l	k	j	I	h	g	f	e	d	c	b	a	z	y	x	w	v	u	t	s
19	s	r	q	p	o	n	m	l	k	j	I	h	g	f	e	d	c	b	a	z	y	x	w	v	u	t
20	t	s	r	q	p	o	n	m	l	k	j	I	h	g	f	e	d	c	b	a	z	y	x	w	v	u
21	u	t	s	r	q	p	o	n	m	l	k	j	I	h	g	f	e	d	c	b	a	z	y	x	w	v
22	v	u	t	s	r	q	p	o	n	m	l	k	j	I	h	g	f	e	d	c	b	a	z	y	x	w
23	w	v	u	t	s	r	q	p	o	n	m	l	k	j	I	h	g	f	e	d	c	b	a	z	y	x
24	x	w	v	u	t	s	r	q	p	o	n	m	l	k	j	I	h	g	f	e	d	c	b	a	z	y
25	z	a	b	c	d	e	f	g	h	I	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y

Table II

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
!	**"**	**#**	**$**	**%**	**&**	**'**	**(**	**)**	*****	**+**	**,**	**-**	**.**	**/**	**:**	**;**	**<**	**=**	**>**	**?**	**@**
@	?	>	=	<	;	:	/	.	-	,	+	*	)	(	'	&	%	$	#	"	!
?	>	=	<	;	:	/	.	-	,	+	*	)	(	'	&	%	$	#	"	!	@
>	=	<	;	:	/	.	-	,	+	*	)	(	'	&	%	$	#	"	!	@	?
=	<	;	:	/	.	-	,	+	*	)	(	'	&	%	$	#	"	!	@	?	>
<	;	:	/	.	-	,	+	*	)	(	'	&	%	$	#	"	!	@	?	>	=
;	:	/	.	-	,	+	*	)	(	'	&	%	$	#	"	!	@	?	>	=	<
:	/	.	-	,	+	*	)	(	'	&	%	$	#	"	!	@	?	>	=	<	;
/	.	-	,	+	*	)	(	'	&	%	$	#	"	!	@	?	>	=	<	;	:
.	-	,	+	*	)	(	'	&	%	$	#	"	!	@	?	>	=	<	;	:	/
-	,	+	*	)	(	'	&	%	$	#	"	!	@	?	>	=	<	;	:	/	.
,	+	*	)	(	'	&	%	$	#	"	!	@	?	>	=	<	;	:	/	.	-
+	*	)	(	'	&	%	$	#	"	!	@	?	>	=	<	;	:	/	.	-	,
*	)	(	'	&	%	$	#	"	!	@	?	>	=	<	;	:	/	.	-	,	+
)	(	'	&	%	$	#	"	!	@	?	>	=	<	;	:	/	.	-	,	+	*
(	'	&	%	$	#	"	!	@	?	>	=	<	;	:	/	.	-	,	+	*	)
'	&	%	$	#	"	!	@	?	>	=	<	;	:	/	.	-	,	+	*	)	(
&	%	$	#	"	!	@	?	>	=	<	;	:	/	.	-	,	+	*	)	(	'
%	$	#	"	!	@	?	>	=	<	;	:	/	.	-	,	+	*	)	(	'	&
$	#	"	!	@	?	>	=	<	;	:	/	.	-	,	+	*	)	(	'	&	%
#	"	!	@	?	>	=	<	;	:	/	.	-	,	+	*	)	(	'	&	%	$
"	!	@	?	>	=	<	;	:	/	.	-	,	+	*	)	(	'	&	%	$	#
!	@	?	>	=	<	;	:	/	.	-	,	+	*	)	(	'	&	%	$	#	"