PERANCANGAN PROGRAM APLIKASI KRIPTOSISTEM MENGGUNAKAN ALGORITMA *SQUARE* YANG DIMODIFIKASI DAN FUNGSI *HASH* SHA-1

Marzuki Silalahi, Tumpal P, Herlina Susanti
Dosen FASILKOM – UIEU
Dosen Universitas Tarumanagara, Jakarta
Mahasiswa Universitas Tarumanagara, Jakarta
tumpal@plasa.com

Abstract

Security and confidentiality of data or information has become an important issue within each person or in an organization. Especially, if this concerned data is in a computer connected with public network, such as LAN. The data will be vulnerable of computer crime done by unauthorized person. This article will discussed about designing a cryptographic application by using a combination between symmetric algorithm SQUARE and hash function SHA-1, to provide not only security but also authenticity in message delivery through network. Symmetric algorithm SQUARE with 128 bits key length will provide security of message delivery through network, by encrypting the message into an incomprehensible form, so it can not be accessed by unauthorized person, only by the one who also knows the key. While using a hash algorithm SHA-1 (Secure Hash Algorithm), user can produce message digest and use it as a digital signature of the message which provide authenticity, by then user will know if the message is really come from a trusted person.

Keywords: *Cryptosystem, SQUARE, SHA-1, Message Authentication, Conventional Encryption, Computer Security, Digital Signature, Hash Algorithm, Symmetric-Key Algorithm, Public-Key Encryption.*

Pendahuluan

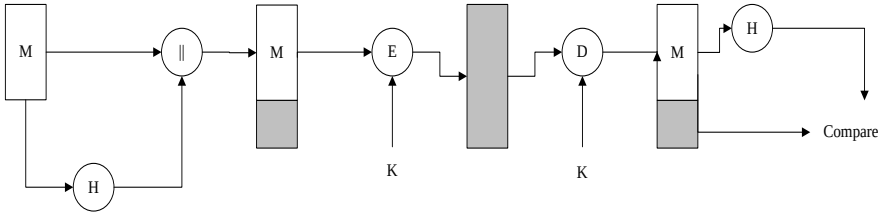
Masalah keutuhan dan kerahasiaan data telah menjadi salah satu aspek penting dari sistem informasi, karena semakin banyak kejahatan komputer yang timbul. Untuk menghindari terjadinya kejahatan komputer, maka diperlukan suatu sekuriti (keamanan) yang baik, sehingga data yang terdapat pada komputer menjadi lebih aman. Salah satu cara yang paling baik adalah dengan menggunakan kriptografi. Terdapat berbagai macam algoritma dalam kriptografi, namun tidak semua mampu memberikan jaminan keamanan dan kerahasiaan yang baik terhadap pesan. Beberapa algoritma yang digunakan dalam program aplikasi kriptografi ini adalah algoritma enkripsi *Square* dan fungsi *hash* SHA-1. Pemilihan kedua algoritma ini, dengan keunggulannya masing-masing dapat saling melengkapi, karena tidak hanya dapat memberikan jaminan kerahasiaan namun juga autentikasi karena menghasilkan *digital signature* dari sebuah pesan.

Tujuan dari perancangan ini adalah untuk merancang suatu program aplikasi kriptosistem menggunakan algoritma enkripsi *Square* dan fungsi *hash* SHA-1 yang dapat memberikan suatu sekuriti yang lebih baik pada sebuah komputer, dapat menjamin keamanan dan kerahasiaan data, dan memberikan jaminan autentifikasi dari pengiriman data dalam jaringan.

Rancangan yang dibuat adalah sebuah program aplikasi komputer yang dapat memberikan keamanan data pada komputer dan menjamin autentifikasi dari data tersebut dalam pengiriman melalui jaringan. Program tersebut merupakan sistem enkripsi, dekripsi, dan autentifikasi dengan menggunakan algoritma enkripsi *Square* dengan kunci simetrik (*symmetric key*) dan fungsi *hash* SHA-1. Program aplikasi ini menggunakan bahasa pemrograman Visual Basic 6.0 dan Microsoft Access 2003.

Perancangan

Komponen Rancangan adalah unit enkripsi dekripsi dengan *algoritma Square,* unit hash dengan fungsi hash SHA-1, unit pengiriman dan penerimaan file dalam jaringan, dan unit pembanding. Spesifikasi Rancangan dari Unit Enkripsi dan Dekripsi, menggunakan algoritma SQUARE yang merupakan 128-bit *symmetric block chiper* dengan panjang key 128-bit. Metode operasi yang digunakan adalah metode operasi Electronic CodeBook (ECB). Unit Hash menggunakan algoritma Secure Hash Algorithm (SHA-1), untuk mengkompresi pesan yang dapat berbentuk teks apa saja (karakter ASCII) dengan panjang tidak terbatas, hingga menghasilkan message digest dengan panjang 160-bit yang kemudian disambungkan dengan pesan asli. Unit Pengiriman dan Penerimaan File di dalam Jaringan Sistem pengiriman dan penerimaan file ini hanya bisa dilakukan di dalam *Local Area Network* (LAN). Unit Pembanding digunakan oleh penerima pesan untuk memeriksa keutuhan dan keaslian pesan yang diterima dengan membandingkan nilai *message digest* yang dihasilkan sebelum dan sesudah proses pengiriman. Sehingga dapat diketahui apakah pesan tersebut tidak mengalami perubahan saat proses pengiriman, masih asli, dan benar-benar berasal dari pihak pengirim pesan. Adapun struktur yang digunakan dalam program aplikasi kriptografi ini dapat digambarkan sebagai berikut:



Sumber: Williams Stallings, 1995

Gambar 1. Struktur *Message Authentication* dan *Hash Function*

Algoritma *Square*

Tahap-tahap pada algoritma ini adalah:

1. Transformasi Linear θ (Tahap *diffusion*)
   $\theta : b = \theta(a) \Leftrightarrow b_{i,j} = c_j a_{i,0} \oplus c_{j-1} a_{i,1} \oplus c_{j-2} a_{i,2} \oplus c_{j-3} a_{i,3}$ dengan $c(x) = \oplus_j c_j x^j$, $b = \theta(a) \Leftrightarrow b_i(x) = c(x) a_i(x) \bmod 1 \oplus x^4$ untuk $0 \le i < 4$
   Inverse dari θ yang sesuai dengan polynomial $d(x)$ adalah sebagai berikut:
   $d(x)c(x) = 1 \pmod{1 \oplus x^4}$
2. Transformasi Nonlinear γ (Tahap nonlinear): $\gamma : b = \gamma(a) \Leftrightarrow bi,j = S\gamma(ai,j)$
3. Permutasi Byte π (Tahap *dispersion*)
   $\pi : b = \pi(a) \Leftrightarrow b_{i,j} = a_{j,i}$
4. Penambahan Round Kunci σ
   $\sigma[k^t] : b = \sigma[k^t](a) \Leftrightarrow b = a \oplus k^t$

HASH dengan Algoritma SHA-1

Proses SHA-1 memiliki beberapa tahapan sebagai berikut:

1. Menambahkan bit-bit tambahan (*padding bits*).
2. Penambahan panjang. Panjang total ***n****512-bit dengan ***n*** > 0 dan dapat didefinisikan sebagai tahapan dari ***n*** blok, M_1 , M_2,..., M_n.
3. Menginisialisasi *buffer message digest*
   Word A = 67 45 23 01
   Word B = EF CD AB 89
   Word C = 98 BA DC FE
   Word D = 10 32 54 76
   Word E = C3 D2 E1 F0
4. Memproses pesan dalam blok 512-bit
   Hasil keluaran dari putaran keempat (tahap ke-80) ditambahkan pada *input* putaran pertama $\boldsymbol{CV_q}$ untuk menghasilkan $\boldsymbol{CV_{q+1}}$. Penambahan dilakukan secara independen pada setiap lima karakter pada *buffer* dengan setiap karakter pada $\boldsymbol{CV_q}$ dengan menggunakan penambahan modulo 2^{32}.
5. Hasil keluaran.
   $CV_0 = IV$ (1)
   $CV_{q+1} = SUM_{32}(CV_q, ABCDE_q)$ (2)
   $MD = CV_L$ (3)

Dimana:

IV = nilai inisial *buffer* ABCDE, didefinisikan tahap ke-3.
ABCDE*q*=hasil keluaran putaran akhir memproses blok pesan ke-*q*.
L = jumlah blok pesan (termasuk *padding* dan panjang *field*).
Sum_{32} =penambahan modulo 2^{32} bit yang dilakukan terpisah pada setiap karakter pada pasangan *input*.
MD =nilai *message digest* akhir.

Fungsi kompresi SHA-1 pada setiap 80 tahap dalam memproses blok 512-bit memiliki bentuk perhitungan:

$$\mathbf{A, B, C, D, E \leftarrow (E + \mathit{f}(t,B,C,D) + S^5(A) + W_t + K_t), A, S^{30}(B), C, D} \quad (4)$$

Dimana:

A,B,C,D,E =lima karakter dalam *buffer*
T =jumlah tahap, $0 \le t \le 79$,
S^k =rotasi kekiri dari 32-bit dengan argumen bit-*k*,
W_t =karakter 32-bit yang dihasilkan dari blok masukan 512-bit,
K_t =konstanta penambahan, dengan menggu-

nakan empat jangkauan nilai seperti yang telah dijelaskan sebelumnya.

+ =modulo penambahan 2^{32}

Sebuah urutan fungsi logika $f_0, f_1,..., f_{79}$ digunakan dalam SHA-1. Masing-masing $\mathbf{f_t}$, 0 <= t <= 79, beroperasi pada tiga buah buffer 32-bit B, C, D, dan menghasilkan sebuah karakter 32-bit sebagai *output*-nya, yang dapat didefinisikan sebagai berikut:

f_t(B,C,D) = (B AND C) OR ((NOT B) AND D) (0 <= t <= 19)

f_t(B,C,D) = B XOR C XOR D (20 <= t <= 39)

f_t(B,C,D) = (B AND C) OR (B AND D) OR (C AND D) (40 <= t <= 59)

f_t(B,C,D) =B XOR C XOR D(60<=t<=79)

Hal ini mengindikasikan bagaimana nilai karakter 32-bit $\boldsymbol{W_t}$ dihasilkan dari pesan dengan panjang 512-bit. 16 nilai pertama dari $\boldsymbol{W_t}$ diambil langsung dari 16 karakter dari blok terakhir. Nilai yang tersisa didefinisikan sebagai berikut:

$$W_t = S^1 (W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3}) \quad (5)$$

Dalam 16 tahap pertama dari proses, nilai $\boldsymbol{W_t}$ sama dengan karakter yang bersangkutan pada blok pesan. Untuk 64 tahapan yang tersisa, nilai $\boldsymbol{W_t}$ dihasilkan dari *shift* kiri satu bit dari hasil XOR keempat nilai $\boldsymbol{W_t}$ putaran sebelumnya.

Hasil Perancangan dan Pembahasan

Perhitungan Algoritma SQUARE

Plaintext: HerlinaSusanti

Key: FTIUntar20042005

Setelah *plaintext* dan *key* diubah dalam bentuk biner, maka proses selanjutnya adalah menjalankan tahap-tahap yang ada pada algoritma *Square*. Tahap-tahap tersebut adalah:

Tahap Linier, lanngkah-langkahnya:

1. XOR Plantext dengan Key

Hasilnya:

00001110	00110001	00111011	00111001
00000111	00011010	00000000	00100001
01000111	01000011	01010001	01011010
01000110	01011001	00110000	00110101

2.XOR Baris

Hasilnya:

00111101	00111101	00111101	00111101
00111100	00111100	00111100	00111100
00001111	00001111	00001111	00001111
00011010	00011010	00011010	00011010

Tahap Nonlinear, langkah-langkahnya:

1. XOR hasil tahap linear dengan KEY

00111101	00111101	00111101	00111101
01000110	01010100	01001001	01010101
01111011	01101001	01110100	01101000
00111100	00111100	00111100	00111100
01101110	01110100	01100001	01110010
01010010	01001000	01011101	01001110
00001111	00001111	00001111	00001111
00110010	00110000	00110000	00110100
00111101	00111111	00111111	00111011
00011010	00011010	00011010	00011010
00110010	00110000	00110000	00110101
00101000	00101010	00101010	00101111

Hasilnya:

01111011	01101001	01110100	01101000
01010010	01001000	01011101	01001110
00111101	00111111	00111111	00111011
00101000	00101010	00101010	00101111

Tahap *dispersion*, langkah-langkahnya :

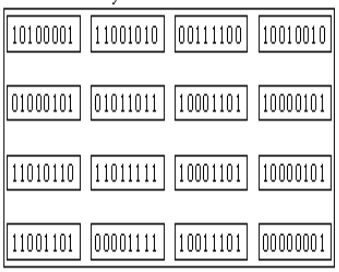
1. XOR hasil tahap nonlinear dengan KEY

10100001	01000101	11010110	11001101
01000110	01010100	01001001	01010101
11100111	00010001	10011111	10011000
11001010	01011011	11011111	00001111
01101110	01110100	01100001	01110010
10100100	00101111	10111110	01111101
00111100	10001101	10001101	10011101
00110010	00110000	00110000	00110100
00001110	10111101	10111101	10101001
10010010	10000101	10000101	00000001
00110010	00110000	00110000	00110101
10100000	10110101	10110101	00110100

Hasilnya:

11100111	00010001	10011111	10011000
10100100	00101111	10111110	01111101
00001110	10111101	10111101	10101001
10100000	10110101	10110101	00110100

2. Ubah urutan baris menjadi kolom dan sebaliknya, urutan kolom menjadi baris, maka hasilnya:



Hasil ini merupakan ciphertext untuk putaran (round) pertama. Pada putaran berikutnya dilakukan tahap-tahap yang sama seperti di atas. Perhitungan ini terus dilakukan sampai dengan putaran ke delapan.

Perhitungan Algoritma SHA-1

Pesan : abc
Kode Biner : 01100001 01100010 01100011.
Panjang pesan l = 24

Langkah 1 Penambahan padding bits

Penambahan bit "1" menjadi :
01100001 01100010 01100011 1
Penambahan 423 bit "0" menjadi
01100001 01100010 01100011 10000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

Langkah 2 Penambahan panjang (64-bit) pada pesan

Penambahan representasi dua karakter heksadesimal dari panjang pesan sebelum penambahan *padding* bits (l = 24), ditambahkan heksadesimal
00000000 00000018 menjadi
61626380 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000018

Langkah 3 Inisialisasi *buffer message digest*
A = 67452301 ; B = EFCDAB89 ;
C = 98BADCFE; D = 10325476 ;
E = C3D2E1F0.

Langkah 4 Memproses blok 512 bit
Pemetaan pesan pada blok 1 adalah
W[0] = 61626380 W[1] = 00000000
W[2] = 00000000 W[3] = 00000000
W[4] = 00000000 W[5] = 00000000
W[6] = 00000000 W[7] = 00000000
W[8] = 00000000 W[9] = 00000000
W[10] = 00000000 W[11] = 00000000
W[12] = 00000000 W[13] = 00000000
W[14] = 00000000 W[15] = 00000018

Nilai heksadesimal dari A,B,C,D,E setelah melalui pengulangan tahapan t dimana 0<t<79.

Tahapan	A	B	C	D	E
t = 0	0116FC33	67452301	7BF36AE2	98BADCFE	10325476
t = 1	8990536D	0116FC33	59D148C0	7BF36AE2	98BADCFE
t = 2	A1390F08	8990536D	C045BF0C	59D148C0	7BF36AE2
t = 3	CDD8E11B	A1390F08	626414DB	C045BF0C	59D148C0
t = 4	CFD499DE	CDD8E11B	284E43C2	626414DB	C045BF0C
t = 5	3FC7CA40	CFD499DE	F3763846	284E43C2	626414DB
t = 6	993E30C1	3FC7CA40	B3F52677	F3763846	284E43C2
t = 7	9E8C07D4	993E30C1	0FF1F290	B3F52677	F3763846
t = 8	4B6AE328	9E8C07D4	664F8C30	0FF1F290	B3F52677
t = 9	8351F929	4B6AE328	27A301F5	664F8C30	0FF1F290
t = 10	FBDA9E89	8351F929	12DAB8CA	27A301F5	664F8C30
t = 11	63188FE4	FBDA9E89	60D47E4A	12DAB8CA	27A301F5

t = 12	4607B664	63188FE4	7EF6A7A2	60D47E4A	12DAB8CA
t = 13	9128F695	4607B664	18C623F9	7EF6A7A2	60D47E4A
t = 14	196BEE77	9128F695	1181ED99	18C623F9	7EF6A7A2
t = 15	20BDD62F	196BEE77	644A3DA5	1181ED99	18C623F9
t = 16	4E925823	20BDD62F	C65AFB9D	644A3DA5	1181ED99
t = 17	82AA6728	4E925823	C82F758B	C65AFB9D	644A3DA5
t = 18	DC64901D	82AA6728	D3A49608	C82F758B	C65AFB9D
t = 19	FD9E1D7D	DC64901D	20AA99CA	D3A49608	C82F758B
t = 20	1A37B0CA	FD9E1D7D	77192407	20AA99CA	D3A49608
t = 21	33A23BFC	1A37B0CA	7F67875F	77192407	20AA99CA
t = 22	21283486	33A23BFC	868DEC32	7F67875F	77192407
t = 23	D541F12D	21283486	0CE88EFF	868DEC32	7F67875F
t = 24	C7567DC6	D541F12D	884A0D21	0CE88EFF	868DEC32
t = 25	48413BA4	C7567DC6	75507C4B	884A0D21	0CE88EFF
t = 26	BE35FBD5	48413BA4	B1D59F71	75507C4B	884A0D21
t = 27	4AA84D97	BE35FBD5	12104EE9	B1D59F71	75507C4B
t = 28	8370B52E	4AA84D97	6F8D7EF5	12104EE9	B1D59F71
t = 29	C5FBAF5D	8370B52E	D2AA1365	6F8D7EF5	12104EE9
t = 30	: 1267B407	C5FBAF5D	A0DC2D4B	D2AA1365	6F8D7EF5
t = 31	3B845D33	1267B407	717EEBD7	A0DC2D4B	D2AA1365
t = 32	046FAA0A	3B845D33	C499ED01	717EEBD7	A0DC2D4B
t = 33	2C0EBC11	046FAA0A	CEE1174C	C499ED01	717EEBD7
t = 34	21796AD4	2C0EBC11	811BEA82	CEE1174C	C499ED01
t = 35	DCBBB0CB	21796AD4	4B03AF04	811BEA82	CEE1174C
t = 36	0F511FD8	DCBBB0CB	085E5AB5	4B03AF04	811BEA82
t = 37	DC63973F	0F511FD8	F72EEC32	085E5AB5	4B03AF04
t = 38	4C986405	DC63973F	03D447F6	F72EEC32	085E5AB5
t = 39	32DE1CBA	4C986405	F718E5CF	03D447F6	F72EEC32
t = 40	FC87DEDF	32DE1CBA	53261901	F718E5CF	03D447F6
t = 41	970A0D5C	FC87DEDF	8CB7872E	53261901	F718E5CF
t = 42	7F193DC5	970A0D5C	FF21F7B7	8CB7872E	53261901
t = 43	EE1B1AAF	7F193DC5	25C28357	FF21F7B7	8CB7872E
t = 44	40F28E09	EE1B1AAF	5FC64F71	25C28357	FF21F7B7
t = 45	1C51E1F2	40F28E09	FB86C6AB	5FC64F71	25C28357
t = 46	A01B846C	1C51E1F2	503CA382	FB86C6AB	5FC64F71
t = 47	BEAD02CA	A01B846C	8714787C	503CA382	FB86C6AB
t = 48	BAF39337	BEAD02CA	2806E11B	8714787C	503CA382
t = 49	120731C5	BAF39337	AFAB40B2	2806E11B	8714787C
t = 50	641DB2CE	120731C5	EEBCE4CD	AFAB40B2	2806E11B
t = 51	3847AD66	641DB2CE	4481CC71	EEBCE4CD	AFAB40B2
t = 52	E490436D	3847AD66	99076CB3	4481CC71	EEBCE4CD
t = 53	27E9F1D8	E490436D	8E11EB59	99076CB3	4481CC71
t = 54	7B71F76D	27E9F1D8	792410DB	8E11EB59	99076CB3
t = 55	5E6456AF	7B71F76D	09FA7C76	792410DB	8E11EB59
t = 56	C846093F	5E6456AF	5EDC7DDB	09FA7C76	792410DB
t = 57	D262FF50	C846093F	D79915AB	5EDC7DDB	09FA7C76
t = 58	09D785FD	D262FF50	F211824F	D79915AB	5EDC7DDB
t = 59	3F52DE5A	09D785FD	3498BFD4	F211824F	D79915AB
t = 60	D756C147	3F52DE5A	4275E17F	3498BFD4	F211824F
t = 61	548C9CB2	D756C147	8FD4B796	4275E17F	3498BFD4

t = 62	B66C020B	548C9CB2	F5D5B051	8FD4B796	4275E17F
t = 63	6B61C9E1	B66C020B	9523272C	F5D5B051	8FD4B796
t = 64	19DFA7AC	6B61C9E1	ED9B0082	9523272C	F5D5B051
t = 65	101655F9	19DFA7AC	5AD87278	ED9B0082	9523272C
t = 66	0C3DF2B4	101655F9	0677E9EB	5AD87278	ED9B0082
t = 67	78DD4D2B	0C3DF2B4	4405957E	0677E9EB	5AD87278
t = 68	497093C0	78DD4D2B	030F7CAD	4405957E	0677E9EB
t = 69	3F2588C2	497093C0	DE37534A	030F7CAD	4405957E
t = 70	C199F8C7	3F2588C2	125C24F0	DE37534A	030F7CAD
t = 71	39859DE7	C199F8C7	8FC96230	125C24F0	DE37534A
t = 72	EDB42DE4	39859DE7	F0667E31	8FC96230	125C24F0
t = 73	11793F6F	EDB42DE4	CE616779	F0667E31	8FC96230
t = 74	5EE76897	11793F6F	3B6D0B79	CE616779	F0667E31
t = 75	63F7DAB7	5EE76897	C45E4FDB	3B6D0B79	CE616779
t = 76	A079B7D9	63F7DAB7	D7B9DA25	C45E4FDB	3B6D0B79
t = 77	860D21CC	A079B7D9	D8FDF6AD	D7B9DA25	C45E4FDB
t = 78	5738D5E1	860D21CC	681E6DF6	D8FDF6AD	D7B9DA25
t = 79	42541B35	5738D5E1	21834873	681E6DF6	D8FDF6AD.

Sumber: Hasil Olahan Data

Langkah 5. Menghasilkan *message digest* 160-bit

A = 67452301 + 42541B35 = A9993E36
B = EFCDAB89+ 5738D5E1=4706816A
C = 98BADCFE+21834873 =BA3E2571
D = 10325476 + 681E6DF6 = 7850C26C
E= C3D2E1F0 + D8FDF6AD =9CD0D89D.

Dari hasil perhitungan di atas, dapat ditarik kesimpulan bahwa nilai *message digest* yang dihasilkan adalah: A9993E36 4706816A BA3E2571 7850C26C 9CD0D89D

Kesimpulan dan Saran

Dari hasil dan pembahasan yang diperoleh dari pembuatan program aplikasi ini aadalah sebagai berikut: 1.Perancangan program aplikasi kriptosistem dengan menggunakan kombinasi algoritma *Square* dan fungsi *hash* SHA-1 ini dapat memberikan sekuriti yang lebih baik terhadap sebuah komputer. 2. Program aplikasi ini dapat mempersulit orang yang tidak berhak dalam melakukan *interception* terhadap *file* karena *file* sudah dienkripsi. 3. Program aplikasi ini dapat menyediakan keamanan dan kerahasiaan data dan informasi dalam sebuah komputer maupun dalam transmisi melalui jaringan. 4. Program aplikasi ini dapat menyediakan autentikasi data dan informasi yang dikirimkan melalui jaringan. Sedangkan saran yang dapat diberikan untuk pengembangan lebih lanjut dari program aplikasi kriptografi ini adalah: 1. Untuk memberikan jaminan keamanan data yang lebih baik, dapat digunakan kunci dengan panjang bit yang lebih besar.

Daftar Pustaka

Koblitz, N., *"A Course in Number Theory and Cryptography",* Springer-Verlag, New York, 1997.

Rijmen, Vincent, *"The Block Cipher SQUARE",*http://www.esat.k

uleuven.ac.be/~rijmen/ square/,14 Desember 1997.

Schneider, Bruce, *"Applied Cryptography: Protocols, Algorithms and Source Code in C",* 2nd edition, John Wiley & Sons Inc., Toronto, 1996.

Stallings, Williams, *"Cryptography and Network Security: Principles and Practices",* 3rd edition, Prentice Hall Inc, Upper Saddle River, 2003.