Key Strengthening Menggunakan KD5

Eko Budhi Susanto[1]

Departemen Teknik Informatika
Institut Teknologi Bandung
Jl. Ganesha 10 Bandung 40132
E-mail : *if14075@students.if.itb.ac.id*[1]

Abstraksi

Kunci, atau biasa kita sebut *key*, dalam konteks kriptografi adalah suatu informasi yang mengendalikan jalannya sebuah algoritma kriptografi. Dalam mendesain suatu sistem keamanan digital, harus diasumsikan bahwa para penyerang telah mengetahui algoritma yang digunakan. Dan sesuai dengan prinsip **Kerckhoff**, hanya kerahasiaan kunci yang menjamin keamanan. Kunci lebih mudah dirahasiakan karena berisi lebih sedikit informasi, memiliki banyak kemungkinan, dan mudah diganti. Jadi, keamanan sebuah enkripsi tergantung pada kerahasiaan kunci yang digunakan. Karena itu, untuk menjamin keamanan sebuah enkripsi diperlukan kunci yang sulit untuk ditebak, atau biasa dinamakan kunci kuat (*strong key*). *Strong key* bisa didapat dengan beberapa cara, antara lain, bisa dengan ditentukan sendiri, sesuai syarat sebuah *strong key*, atau bisa juga dengan *key generator*, dan bisa juga dengan melakukan Penguatan kunci (*key strengthening*) pada kunci yang lemah (*weak key*). Dalam makalah ini akan dibahas hasil eksperimen penguatan kunci dengan cara membuat *Message Digest* dari sebuah kunci lemah masukan menggunakan fungsi yang diberi nama KD5 (*key strengthening with ascii converted MD5 hash*). Hasil akhir dari fungsi ini berupa 16 karakter Ascii.

***Kata kunci** : Strong Key,Key Generator, Key Strengthening, Weak Key, KD5, MD5, ASCII.*

1. Latar Belakang Masalah

Kunci, atau biasa kita sebut *key*, dalam konteks kriptografi adalah suatu informasi yang mengendalikan jalannya sebuah algoritma kriptografi. Dalam proses enkripsi, kunci memberikan cara khusus bagi sebuah algoritma untuk mentransformasikan sebuah *plaintext* menjadi sebuah *ciphertext*, ataupun sebaliknya. Kunci digunakan dalam berbagai algoritma kriptografi, dari mulai algoritma kriptografi kunci-simetri sampai Nirsimetri.

Dalam sebuah sistem keamanan digital, algoritma enkripsi dan dekripsi bukan menjadi sesuatu yang rahasia, jadi pihak penyerang yang ingin menyusup atau mengambil pesan rahasia selalu diasumsikan mengetahui algoritma yang digunakan. Maka, untuk menjamin keamanan, kunci lah yang harus dirahasiakan. Kunci lebih mudah dirahasiakan dibanding algoritma yang digunakan, karena berisi lebih sedikit informasi, memiliki banyak kemungkinan, dan mudah diganti. Maka dari itu, keamanan sebuah sistem enkripsi amat bergantung pada kerahasiaan kunci yang digunakan.

Maka dari itu, diperlukan kunci yang kuat (*Strong key*) untuk menjamin keamanan sebuah enkripsi. Syarat kunci yang kuat, seperti syarat sebuah kata sandi yang kuat, harus terdiri dari :

- Gabungan karakter *Uppercase* dan *Lowercase*
- Angka
- Karakter spesial

Dalam makalah ini akan dibahas hasil dari sebuah eksperimen penguatan kunci yang menghasilkan sebuah fungsi yang diberi nama KD5. Fungsi ini dapat menghasilkan *strong key* dari sebuah *weak key*. KD5 membangkitkan nomor *hash* dari *weak key* yang di input menggunakan MD5, lalu mempartisi hasil *hash*-nya yang berukuran 128 bit, atau 32 karakter hexadesimal, menjadi 16 buah bilangan hexadesimal yang masing-masing terdiri dari 2 karakter (minimum 00, maksimum FF), dan mengubahnya menjadi 16 karakter Ascii.

2. *Key Strengthening*

Dalam kriptografi, *Key Strengthening* atau biasa juga disebut dengan *Key Stretching* merujuk pada sebuah teknik yang digunakan untuk memperkuat sebuah kunci lemah.

Mengingat penggunaan kunci dalam sebuah sistem itu banyak sekali, dan biasanya yang sering digunakan adalah kunci lemah, maka teknik ini menjadi sesuatu yang penting dalam dunia kriptografi. Dalam kehidupan sehari-hari, contoh kunci lemah adalah password komputer yang mudah diingat oleh ingatan manusia. Kunci lemah biasanya dapat dipecahkan dengan mudah menggunakan kombinasi serangan *Brute Force* dan *Dictionary Attack*.

Teknik *Key Strengthening* biasanya bekerja sebagai berikut : Menggunakan sebuah algoritma *Key Strengthening* dengan input kunci lemah, dan menghasilkan sebuah kunci kuat. Untuk memcahkan sebuah kunci kuat dibutuhkan waktu kalkulasi yang

tidak sedikit dan membutuhkan prosesor super canggih.

Ada beberapa cara untuk melakukan *Key Strengthening*, antara lain :

- *Key Strengthening* menggunakan fungsi hash
- *Key Strengthening* menggunakan metode *block cipher*
- Dan lain-lain

Dalam makalah ini akan dibahas hasil eksperimen pembuatan fungsi *Key Strengthening* baru yang memanfaatkan fungsi hash MD5. Untuk lebih jelasnya akan dibahas pada bagian selanjutnya.

3. Penjelasan Algoritma MD5

MD5 adalah fungsi satu arah yang dibuat oleh **Ronald Rivest** pada tahun 1991. MD5 menerima masukan berupa pesan dengan ukuran sembarang dan menghasilkan *message digest* yang panjangnya 128 bit.

Langkah-langkah pembuatan *message digest* secara garis besar adalah sebagai berikut :

- Penambahan nilai bit-bit pengganjal (*padding bits*)
- Penambahan nilai panjang pesan semula.
- Inisialisasi penyangga (*buffer*) MD
- Pengolahan pesan dalam blok berukuran 512 bit

Pertama kali akan dilakukan penambahan bit pengganjal. Bit-bit pengganjal ditambahkan pada pesan sehingga panjangnya dapat dibagi dengan 512. Angka 512 ini muncul karena MD5 mengolah pesan dalam blok-blok yang berkuran 512 bit. Sebuah bit 1 akan disambungkan di akhir pesan yang akan ditambahkan dengan 0 sebanyak yang dibutuhkan agar panjang pesan adalah kurang 64 dari suatu kelipatan 512. Sisa dari 64 tersebut akan diisi dengan sebuah nilai integer yang merepresentasikan panjang asli pesan tersebut.

Algoritma utama MD5 akan beroperasi dengan menggunakan state 128 bit yang dibagi ke dalam empat 32-bit penyangga *(buffer).* Dinotasikan sebagai penyangga A, B, C dan D, yang akan diinisialisasi dengan konstanta tertentu. Untuk versi MD5 yang umum buffer tersebut diinisialisasi sebagai berikut :

```
A = 01234567
B = 89ABCDEF
C = FEDCBA98
D = 76543210
```

Pesan kemudian akan dibagi menjadi sejumlah blok 512 bit. Algoritma kemudian akan beroperasi di tiap 512 bit blok dari pesan, tiap blok akan memodifikasi penyangga. Pemrosesan untuk tiap blok pesan terdiri dari 4 tingkat yang mirip yang dinamakan *round*. Tiap *round* terdiri dari 16 operasi yang mirip berdasarkan sebuah fungsi non linear F, *modular addition*, dan *left rotation*. Fungsi dibawah mengilustrasikan ssatu operasi didalam sebuah *round.*

Terdapat 4 kemungkinan fungsi F, yang masing-masing akan digunakan di tiap *round* :

$$F(X,Y,Z) = (X \wedge Y) \vee (\neg X \wedge Z)$$
$$G(X,Y,Z) = (X \wedge Z) \vee (Y \wedge \neg Z)$$
$$H(X,Y,Z) = X \oplus Y \oplus Z$$
$$I(X,Y,Z) = Y \oplus (X \wedge \neg Z)$$

$\wedge$, $\vee$, $\oplus$, $\neg$ menyatakan masing-masing operator AND, OR, XOR, dan NOT.

Yang patut menjadi perhatian utama di algoritma MD5 dalam masalah *birthday attack* adalah pesan dengan panjang sembarang akan selalu menghasilkan pesan ringkas dengan panjang 128, sehingga jumlah kemungkinan keluaran yang harus dihasilkan adalah 2^{128} kemungkinan pesan ringkas unik.

4. Penjelasan KD5

KD5 (*key strengthening with ascii converted MD5 hash*) adalah sebuah fungsi penguatan kunci (*key strengthening*) yang dikembangkan dari algoritma MD5, dengan memanfaatkan teknik manipulasi String.

Dilihat dari jenisnya, KD5 adalah jenis metode *key strengthening* yang memanfaatkan fungsi hash. KD5 dikembangkan atas dasar kebutuhan akan kunci atau *password* yang aman, dalam artian sulit untuk dipecahkan oleh manusia, dengan bantuan komputer sekalipun.

Dalam prakteknya, KD5 mengubah kunci lemah menjadi kunci yang kuat, mengandung kombinasi karakter *Uppercase*, *Lowercase*, angka, dan karakter spesial.

Skema dasar KD5 adalah seperti berikut :

Kunci kuat = KD5(kunci lemah)

Dalam mengubah kunci kuat menjadi kunci lemah, KD5 memiliki langkah-langkah sebagai berikut :

- Menerima input berupa sebuah kunci lemah berukuran bebas
- Melakukan Generasi *HashNumber* kunci lemah menggunakan fungsi MD5

- Melakukan looping, mengambil per dua karakter String output MD5 yang memiliki panjang 32 karakter. Jadi loop terjadi sebanyak 16 kali.
- Dua karakter yang diambil, direpresentasikan sebagai sebuah bilangan heksadesimal dua digit. (Minimum 0x00, maksimum 0xFF)
- Karakter-karakter tersebut dikonversi menjadi karakter ASCII yang dapat di cetak (*Ascii Printable Character*), yaitu antara karakter ke-32 sampai karakter ke-126.
  Fungsi konversi :

  PrintableAsciiNum = (HexNum%95)+32

  Keterangan :
  - PrintableAsciiNum : bilangan antara 32 sampai 126, output fungsi
  - HexNum : bilangan antara 0 (0x00) sampai 255 (0xFF)
  - 95 : Jumlah karakter Ascii yang *printable*
  - 32 : Karakter pertama yang *printable* dalam tabel Ascii
- Hasil konversi berupa 16 buah bilangan interger yang masing-masing merepresentasikan satu buah karakter Ascii yang *printable* lalu diubah menjadi String sepanjang 16 karakter Ascii yang terdiri dari karakter-karakter yang direpresentasikan oleh bilangan-bilangan tersebut.

Fungsi KD5:

```
KD5(String WeakKey)
{
   String StrongKey = “”;
   String Hash = MD5Function(WeakKey);
   For(int i=0; i<Hash.length; i+=2)
   {
      Hex=Hash.SubString(i,2);
      //fungsi konversi
      int AsciiNum = ((int)Hex%95)+32;
      Output += AsciiNum.toString();
   }
   Return StrongKey;
}
```

Contoh penggunaan fungsi adalah sebagai berikut :

- Input : informatika
- Hasil Hash MD5 :
  `EFE3F2F24F93AAD0E5ED80AD2036D284`
  Hasil fungsi tersebut dipecah menjadi 16 *substring* :
  `%EF %E3 %F2 %F2 %4F %93 %AA %D0`
  `%E5 %ED %80 %AD %20 %36 %D2 %84`
- Hasil Fungsi konversi :
  `81 69 84 84 111 84 107 50`
  `71 79 65 110 64 86 52 69`
- Lalu, merujuk pada tabel Ascii, maka dari angka-angka tersebut terbentuklah susunan karakter seperti ini :
  Output : **QETToTk2GOAn@V4E**
- Tabel karakter Ascii selengkapnya dapat dilihat pada Lampiran 1.

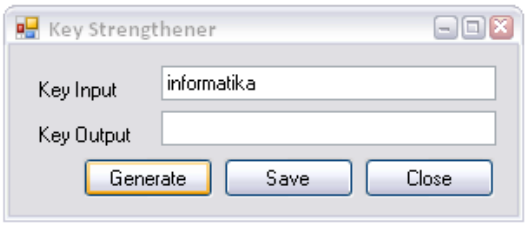
Seperti dapat dilihat, hasil dari fungsi KD5 biasanya merupakan gabungan antara huruf kecil, huruf besar, angka dan karakter spesial. Gabungan karakter-karakter tersebut, seperti dikatakan sebelumnya, merupakan ciri sebuah kunci kuat.

5. Penjelasan Aplikasi KD5

Aplikasi KD5 yang dibuat, yaitu **Key Strengthener**, berfungsi sebagai aplikasi yang dapat digunakan untuk berbagai keperluan yang berhubungan dengan kunci atau kata sandi. Aplikasi ini memiliki antarmuka yang sederhana, namun tepat guna, tujuannya agar mudah digunakan dan praktis.

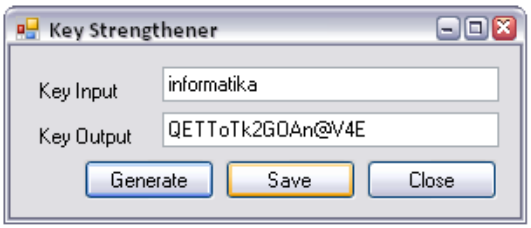
Aplikasi ini dibangun menggunakan Visual Studio .Net 2005, dengan bahasa pemrograman C#, dan berjalan pada sistem operasi MS Windows XP maupun MS Windows Vista dengan *dotnet frameworks* 2.0.

Ketika pertama kali kita klik pada ikon aplikasi, maka akan keluar tampilan seperti ini :



Gambar 1 : Antarmuka *Key Strengthener*

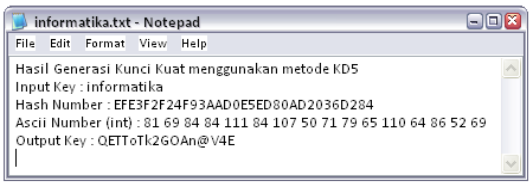
Untuk mencobanya, kita tinggal meng-input sebuah kunci lemah, misalnya dalam contoh ini, **informatika**. Setelah itu, tekan tombol *Generate*, dan akan terlihat hasilnya seperti dibawah ini :



Gambar 2 : *Key Strengthener* dengan hasil generasi kunci

Lalu, untuk menyimpan kunci kuat yang sudah dihasilkan dalam file txt, tinggal menekan tombol *save*, lalu akan terbuka dialog *save file*. Pilih destinasi penyimpanan, ketikkan nama file, lalu

tekan *save*. Isi file txt yang dihasilkan akan seperti ini :



Gambar 3 : File Output hasil *save* dari aplikasi *Key Strengthener*

6. Studi Kunci Hasil Generasi KD5

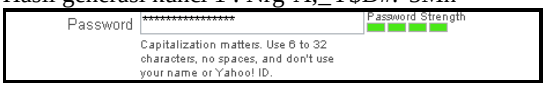
Untuk mengetahui keberhasilan fungsi yang dibuat, maka hasil generasi kunci akan diujikan di beberapa situs internet yang memerlukan *password* sebagai alat autentikasi usernya.

Uji pertama dilakukan pada situs **yahoo** (http://mail.yahoo.com), pengetesan dilakukan pada textbox password saat pendaftaran *account* baru.

Kunci lemah 1 : e

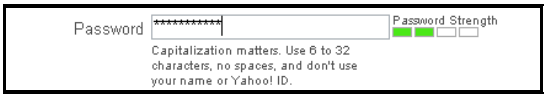
Gambar 4 : Kunci lemah "e" pada yahoo

Hasil generasi kunci 1 : Nrg-A;_Y$D#."5Mn



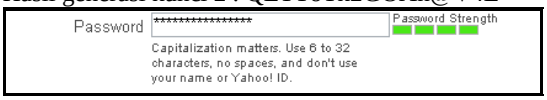
Gambar 5 : Kunci hasil generasi kunci "e" pada yahoo

Kunci lemah 2 : informatika



Gambar 6 : Kunci lemah "informatika" pada yahoo

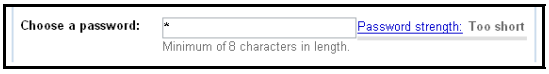
Hasil generasi kunci 2 : QETToTk2GOAn@V4E



Gambar 7 : Kunci hasil generasi kunci "informatika" pada yahoo

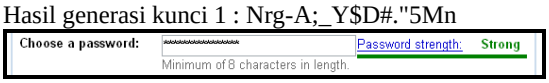
Uji kedua dilakukan pada penyedia layanan email, GMail milik **google** (http://mail.google.com)**.** Pengetesan dilakukan pada textbox password saat pendaftaran *account* baru.

Kunci lemah 1 : e



Gambar 8 : Kunci lemah "e" pada google

Hasil generasi kunci 1 : Nrg-A;_Y$D#."5Mn



Gambar 9 : Kunci hasil generasi kunci "e" pada google

Kunci lemah 2 : informatika

Gambar 10 : Kunci lemah "informatika" pada google

Hasil generasi kunci 2 : QETToTk2GOAn@V4E

Gambar 11 : Kunci hasil generasi kunci "informatika" pada google

Uji ketiga dilakukan pada sebuah fungsi pengetes kekuatan kunci yang terdapat pada formulir pendaftaran forum ***justpressplay*** (http://www.justpressplay.net/forum/).

Kunci lemah 1 : e

Gambar 12 : Kunci lemah "e" pada *justpressplay*

Hasil generasi kunci kuat 1 : Nrg-A;_Y$D#."5Mn

Gambar 13 : Kunci hasil generasi kunci "e" pada *justpressplay*

Kunci lemah 2 : informatika

Gambar 14 : Kunci lemah "informatika" pada *justpressplay*

Hasil generasi kunci 2 : QETToTk2GOAn@V4E

Gambar 15 : Kunci hasil generasi kunci "informatika" pada *justpressplay*

Situs ketiga ini memiliki fungsi pengetes kunci yang bagus. Kunci hasil generasi aplikasi KD5 yang biasanya selalu menghasilkan nilai maksimum pada kedua tes sebelumnya, disini ternyata tidak selalu menghasilkan nilai maksimum. Walaupun begitu, hasil tes tetap selalu menghasilkan kunci yang tergolong kuat.

Kesimpulan dari tes yang dilakukan, fungsi KD5 hampir selalu dapat menghasilkan kunci yang kuat dari kunci lemah. Kekuatan kunci lemah tidak mempengaruhi kekuatan kunci kuat (seperti terlihat pada uji ketiga, kunci *very weak* ternyata menghasilkan kunci *very strong*, sedangkan kunci *weak* hanya menghasilkan kunci *strong*).

7. Kelebihan dan Kekurangan KD5

Kelebihan yang ditawarkan KD5, antara lain :

- Penggunaannya yang praktis.
- Dapat menghasilkan kunci yang kuat sehingga sulit untuk ditebak, bahkan dengan menggunakan gabungan *Dictionary Attack* dan *Bruteforce Attack*.
- Dapat digunakan untuk berbagai jenis keperluan kunci atau *password* dalam dunia maya.

Namun dibalik itu KD5 masih memiliki beberapa kekurangan, antara lain :

- Panjang kunci yang dihasilkan selalu 16 karakter.
- Dalam keadaan tertentu yang sangat jarang, kunci yang dihasilkan bisa juga kunci lemah.
- Kunci kuat yang dihasilkan tidak dapat dibalikkan ke kunci aslinya (input)
- Jika aplikasi KD5 diketahui oleh musuh (penyerang), mungkin saja kunci dapat ditebak. (Menggunakan gabungan *Dictionary Attack*, *Bruteforce Attack*, dan aplikasi KD5)

8. Kesimpulan

Kunci merupakan hal yang sangat penting dalam dunia kriptografi. Kerahasiaan kunci sangat berpengaruh pada kerahasiaan isi pesan yang di-*encrypt*, pada isi *account* sebuah login, dll. Karena hal itu, kekuatan kunci menjadi sangatlah penting.

KD5 adalah sebuah fungsi yang dihasilkan dari sebuah eksperimen mengenai penguatan kunci. Fungsi ini memanfaatkan fungsi hash MD5 dan manipulasi string untuk membangun sebuah kunci kuat dari input kunci lemah, dengan kata lain berfungsi untuk memperkuat kunci.

Dengan adanya fungsi KD5 ini, diharapkan pengguna kunci, untuk keperluan apapun, bisa memperkuat kunci favoritnya (misal: nama, tanggal lahir, dsb) agar kerahasiaan miliknya yang dilindungi oleh kunci menjadi lebih aman dari serangan.

9. Referensi

- Munir, Rinaldi, *Kriptografi*, Institut Teknologi Bandung, 2006.
- Kunci *(kriptografi)* id.wikipedia.org/wiki/kunci_(kriptografi)
- *Key Strengthening* en.wikipedia.org/wiki/Key_stengthening
- *Password Strengthening* www.authenticationworld.com/Password-Authentication/PasswordStrengthening.html
- *Hexadecimal To Ascii* http://www.paulschou.com/tools/xlate/
- *Ascii Table* http://www.asciitable.com/
- *How To Create Strong Password* http://www.microsoft.com/protect/yourself/password/create.mspx
- *printable ascii char* http://www.robelle.com/smugbook/ascii.html

10. Lampiran

Lampiran I, Tabel Ascii
Karakter printable adalah 95 karakter, mulai karakter
ke-33 (#32) sampai ke-127 (#126).

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com