VIGENERE BIT SHIFT : SEBUAH PARADIGMA BARU

Hably Robbi Wafiyya – NIM : 13507128
Program Studi Teknik Informatika
Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
e-mail : if17128@students.if.itb.ac.id

ABSTRAKSI :

Algoritma enkripsi modern yang digunakan saat ini pada umumnya menyandarkan kekuatan mereka pada kesulitan melakukan exhaustive search untuk menemukan kunci yang dicari. Tetapi ternyata banyak cara yang dapat digunakan untuk mengurangi waktu pencarian secara drastic. Dari waktu milyaran tahun menjadi hanya beberapa bulan. Tak ada algoritma modern yang mampu bertahan lebih dari setengah abad, tetapi algoritma vigenere cipher berhasil bertahan tak dapat dipecahkan selama 2 abad. Makalah ini membahas tentang bagaimana menerapkan vigenere cipher dengan prinsip-prinsip algoritma enkripsi modern. Vigenere bit shift dilakukan dalam mode bit, dan disubtitusikan secara homofonik untuk mengelabui analisis statistic terhadap cipherteks.

Kata Kunci : Vigenere, bit, subtitusi homofinik, analisis frekuensi, metode kasiski, brute force, algorotma enkripsi modern

1. PENDAHULUAN

Kriptografi merupakan suatu ilmu dan seni yang bertujuan untuk menjaga kerahasiaan pengiriman pesan. Pesan yang akan dikirim (plainteks) akan diubah menjadi pesan sandi (cipherteks) yang tidak bermakna sebelum dikirimkan ke alamat tujuan. Fungsinya adalah untuk mencegah akses orang yang tidak berhak terhadap pesan tersebut. Proses itu dinamakan enkripsi dan dekripsi. Metode penyandian sangatlah beragam, tetapi secara garis besar algoritma pengenkripsian dibagi menjadi 2, yaitu Algoritma enkripsi klasik dan modern.

Contoh algoritma enkripsi klasik antara lain caesar cipher, vigenere cipher, playfair cipher. Pada dasarnya algoritma enkripsi klasik menggunakan ciri-ciri yang sama, yaitu :

1. Alphanumerik
   Maksudnya adalah algoritma enkripsi klasik hanya mengolah pesan berupa huruf atau angka. Tanda baca seperti titik, koma, atau titik dua pada plainteks tidak dienkripsikan dan tetap seperti apa adanya pada cipherteks
2. Subtitusi
   Teknik subtitusi, yaitu menggantikan suatu huruf menjadi huruf lain, dengan menggunakan suatu pola (seperti vigenere) ataupun acak.
3. Transposisi
   Teknik transposisi, yaitu mengubah posisi karakter dengan pola tertentu. Transposisi yang umu dilakukan adalah transposisi matriks, yaitu mengubah kolom menjadi baris, dan sebaliknya
4. Permutasi
   Mirip dengan transposisi yang intinya adalah mengacak posisi karakter, tetapi permutasi lebih terlihat “acak” walaupun sebenarnya tetap menggunakan pola tertentu. Permutasi dapat dilakukan dengan tabel permutasi atau aturan lain

Algoritma kriptografi klasik memiliki kelemahan fatal, yaitu mudahnya diterka dengan menggunakan metode analisis frekuensi, dikarenakan sifatnya yang alfanumerik dan banyaknya perulangan kata yang dipakai. Vigenere yang pada awalnya dinyatakan “unbreakable” pun berhasil ditumbangkan dengan

menggunakan metode kasiski yang juga memanfatkan analisis frekuensi.

2. VIGENERE CIPHER

Algoritma vigenere ditemukan oleh seorang diplomat Perancis, Blaise de Vigenere pada abad 16. Algoritma ini merupakan salah satu algoritma cipher abjad-majemuk manual terbaik, dan bahkan sempat dinyatakan “unbreakable” selama 2 abad. Vigenere cipher hampir sama dengan caesar cipher. Perbedaannya adalah, caesar cipher hanya menggunakan kunci sebuah huruf, sedangkan vigenere menggunakan kunci berupa string yang akan secara periodik berulang. Cara kerja vigenere cipher adalah sebagai berikut :

1. Nyatakan setiap huruf menjadi sebuah bilangan integer (a=0, b=1, dst).
2. Definisikan Kunci (Key). Jika panjang kunci kurang dari panjang plianteks, maka dilakukan padding (penggenapan)
3. Enkripsikan plainteks dengan menggunakan persamaan berikut :
   $C = (P + K) \bmod 26$
4. Dan dekripsikan dengan menggunakan persamaan
   $P = (C – K) \bmod 26$

Contoh penggunaan vigenere cipher :

Plainteks : THIS PLAINTEKS
Kunci : sony sonysonys
Cipherteks : LVVQ HZNGFHRVL

2.1 Analisis Vigenere Cipher

Kekuatan dari vigenere cipher adalah sulitnya menebak plainteks hanya dengan mengetahui cipherteks. Hal itu dikarenakan sebuah huruf A pada plainteks dapat menjadi huruf E,Z, atau huruf apapun tergantung huruf pada kunci yang bersesuaian.

Namun kekurangannya adalah jika Plainteks dan ciphertext diketahui, maka kita dapat mendapatkan kunci dengan mudah, dan menggunakan kunci itu untuk mendekripsikan cipherteks lain yang belum diketahui plainteksnya. Hal ini dikenal dengan istilah “known-plaintext attack”.

Bahkan, pada tahun 1854, Friedrich Kasiski menemukan sebuah metode untuk menebak plainteks hanya dengan cipherteksnya. Ia memanfaatkan fakta bahwa sebuah plainteks dalam suatu bahasa mengandung beberapa perulangan pasangan huruf atau tripel huruf. Seperti kata ”THE” dalam bahasa inggris yang merupakan kata yang paling sering muncul. Cara kerja metode karsiki adalah sebagai berikut :

1. Cari semua kriptogram berulang, kemudian hitung jarak antara kriptogram yang berulang tersebut
2. Cari faktor pembagi dari jarak tersebut, itu adalah kemungkinan panjang kunci. Kemudian cari irisan antara faktor pembagi jarak kriptogram berulang A, dengan B, dst. Maka sangat mungkin itulah panjang kunci
3. Setelah diketahui panjang kunci, maka bagilah cipherteks menjadi sebanyak panjang kunci. Setiap sub cipherteks selalu dienkripsi dengan huruf yang sama, sehingga hal itu akan membuat kriptanalis memiliki n buah caesar cipher.
4. Gunakan analisis frekuensi untuk menentukan kunci yang sesuai

2.2 Teknik Analisis Frekuensi

Dalam menganalisis sebuah cipherteks, kriptanalis menggunakan berbagai metode, salah satunya adalah analisis frekuensi. Yaitu dengan memanfaatkan ketidakmerataannya frekuensi kemunculan suatu huruf. Dalam bahasa inggris contohnya, huruf E sangat sering muncul, sedangkan huruf Z jarang muncul. Setelah melakukan penelitian terhadap jurnal, media cetak, dan berbagai sumber lain, dibuatlah sebuah tabel frekuensi kemunculan.

Huruf	%	Huruf	%
A	8,2	N	6,7
B	1,5	O	7,5
C	2,8	P	1,9
D	4,2	Q	0,1
E	12,7	R	6,0
F	2,2	S	6,3
G	2,0	T	9,0
H	6,1	U	2,8
I	7,0	V	1,0
J	0,1	W	2,4
K	0,8	X	2,0
L	4,0	Y	0,1
M	2,4	Z	0,1

Tabel di atas merupakan tabel frekuensi kemunculan seriap huruf dalam bahasa inggris. Dapat terlihat bahwa E merupakan huruf yang paling sering muncul dan sangat dominan. Frekuensi kemunculannya 12,7% sedangkan huruf T dengan frekuensi terbesar kedua hanya 9%. Hal ini membuat sebuah huruf yang paling sering muncul pada cipherteks, sangatlah mungkin berkorespondensi dengan huruf E. Setelah itu, kriptanalis dapat mengetahui kunci, dan akhirnya membongkar kunci secara keseluruhan.

Teknik analisis frekuensi adlah teknik yang cukup powerful untuk membongkar algoritma enkripsi klasik, dikarenakan sifatnya yang alphanumerik. Namun tidak berdaya menghadapi algoritma enkripsi modern yang berjalan dalam mode bit

3. ALGORITMA ENKRIPSI MODERN

Penggunaan kriptografi mengalami perkembangan sejalan dengan perkembangan teknologi. Penggunaan komputer digital mengubah paradigma algoritma enkripsi klasik menjadi modern. Enkripsi tidak lagi dilakukan dalam mode karakter, tetapi dalam mode bit biner 0 atau 1.

Jika pada algoritma enkripsi klasik, proses pengenkripsian difokuskan agar cipherteks terlihat acak dan tak bermakna. Dalam algoritma enkripsi modern, kekuatan algoritma lebih ditekankan pada kemustahilan menganalisis cipherteks menjadi plainteks tanpa mengetahui kunci (bandingkan dengan semua algoritma enkripsi klasik yang lemah terhadap teknik analisis frekuensi yang memungkinkan menebak plainteks hanya dengan mengetahui kunci). Kemudian membuat kunci tersebut tidak dapat diterka dan harus dicari dengan metode brute force. Lalu membuat fungsi brute force menjadi tidak magkus dikarenakan panjangnya kunci.

Ringkasnya, algorirtma enkripsi modern memiliki ciri khas sebagai berikut :

1. Berjalan dalam mode bit biner 0 atau 1
2. Enkripsi sedemikian rumit sehingga tidak mungkin menebak plainteks hanya dengan cipherteks
3. Kunci dibuat panjang, agar metode brute force menjadi infeasible (tidak layak digunakan) untuk menganalisis cipherteks

Algritma enkripsi DES, AES, atau RSA menyandarkan kekuatan algoritma mereka pada sulitnya seseorang menebak kunci. Dan tidak mungkinnya mengetrahui plainteks hanya dengan mengetahui cipherteks.

Algoritma DES misalnya, menggunakan jaringan feistel dan feedback sehingga sangatlah rumit. Dan menggunakan kunci sepanjang 56 bit. Artinya, untuk menemukan kunci yang tepat secara brute force, dibutuhkan 2^{56} kali percobaan, atau setara dengan 7,2 x 10^{16} percobaan. Jika dalam satu detik dapat dikerjakan 1 juta percobaan, maka dibutuhkan 1142 tahun untuk menemukan kunci yang benar.

Secara teoritis, perhitungan di atas sangatlah tepat dan logis. Tapi kenyataan berkata lain, pada tahun 1998 (26 tahun setelah DES ditemukan), Electronic Frontier Foundation membuat hardware khusus yang mampu menemukan kunci DES secara exhaustive search hanya dalam waktu 5 hari.

Contoh lain yang lebih ekstrim adalah algoritma kunci-publik RSA terbaru yang menggunakan kunci sepanjang 512 bit dan operasi pemangkatan (operasi pangkat memiliki kompleksitas algoritma yang tinggi) pun berhasil dipecahkan walau secara teritis membutuhkan waktu 10^{25} tahun untuk menemukan kunci yang tepat.

3.1 Kelemahan Algoritma Enkripsi Modern

Walaupun secara teoritis, algoritma enkripsi modern tidak mungkin dapat dipecahkan, ternyata hingga saat ini tidak ada satupun algoritma pengenkripsian yang 100% aman. Mengapa demikian ? Hal itu disebabkan adanya kelemahan fatal pada algoritma enkripsi modern, yaitu menitikberatkan kekuatannya pada kesulitannya mendapatkan kunci yang tepat. Kekuatan algoritma modern adalah “kesulitannya” memecahkan kunci, tetapi bukan “kemustahilannya”. Hal tersebut berarti, algoritma tersebut selalu dapat dipecahkan jika kita dapatberhasil melakukan exhaustive search pada kunci, dan ini adalah sebuah keniscayaan.

Namun, secara teritis, metode exhaustive search membutuhkan waktu yang sangat lama utnuk menemukan kunci, dan bahkan mencapai 10^{25} tahun (yang mungkin saat itu dunia sudah kiamat). Tapi ternyata, perhitungan tersebut dilakukan secara sederhana dan tanpa mempertimbangkan berbagai faktor yang dapat mengurangi waktu exhaustive search menjadi jauh lebih kecil.

Faktor-faktor tersebut antara lain :

1. Perkembangan Teknologi
2. Metode Heuristic
3. Birthday Paradox
4. Zombie Komputer

3.1.1 Perkembangan teknologi

Perkembangan kecepatan pengolahan bit pada processor berkembang sangat pesat. Hukum Moore (walau sekarang ini keakuratannya mulai dipertanyakan) menyatakan bahwa jumlah transistor pada chip akan menjadi dua kali lipatnya setiap 18 bulan. Sampai saat ini, hokum tersebut selalu benar, walau tak ada yang dapat memprediksi kapan hokum moore akan berhenti. Hal itu membuat setiap 1,5 tahun processor memiliki kecepatan proses dua kali lipatnya, dan setelah 3 tahun kecepatannya akan menjadi 4 x lipatnya. Hal itu berarti perhitungan tentang kekuatan algoritma dengan brute force yang dihitung menggunakan processor terkcepat di saatnya dapat menjadi tak valid 1,5 tahun kemudian. Sehinnga kita tak dapat mempercayai begitu saja klaim kekuatan suatu algoritma enkripsi.

Bukan hanya itu, kecepatan pemrosesan bit dapat ditingkatkan lagi dengan merancang hardware khusus yang hanya digunakan untuk melakukan exhaustive search.

3.1.2 Metode Heuristic

Metode Heuristic adalah metode yang digunakan untuk mempersingkat kompleksitas algoritma. Jika untuk memecahkan suatu permasalahan digunakan algoritma biasa dengan kompleksitas O^2, maka dengan metode heuristic, kompleksitasnya dapat diperkecil menjadi O.

Salah satu penggunaan metode heuristic adalah dalam masalah pemfaktoran. Masalah pemfaktoran digunakan dalam pembangkitan kunci publik-privat dalam RSA. Masalah pemfaktoran adalah sebagai berikut :

Diketahui sebuah bilangan n. tentukan semua pasangan bilangan (a,b) sehingga

n = a x b

a dan b merupakan faktor prima dari n

Misal n adalah bilangan 20 digit. Jika menggunakan algoritma exhaustive biasa, maka dibutuhkan percobaan sebanyak 10^{20} untuk menemukan kunci tersebut. Sedangkan metode heuristic memanfaatkan persamaan matematika

Jika n = a x b, maka salah satu faktor pembaginya selalu kurang dari atau sama dengan akar n. Sehingga cukup dicari nilai faktor pembagi yang lebih kecil, yang range nya hanya sebesar akar n. Sehingga exhaustive search yang dilakukan hanya perlu melakukan 10^{10} percobaan. Lihatlah bahwa dengan fungsi heuristic sederhana, waktu pencarian dapat dikurangi hingga 1/10.000.000.000

Dengan penurunan rumus yang lebih rumit lagi, dapat diperoleh fungsi heuristic yang lebih baik lagi, dan waktu yang dibutuhkan menjadi jauh lebih kecil lagi.

3.1.3 Birthday Paradox

Berapa jumlah sampel yang dibutuhkan untuk PASTI mendapatkan dua orang dengan tanggal ulang tahun yang sama? Secara logika, jelas kita dapat mengatakan 366 sampel (asumsi bukan tahun kabisat).

Namun, berapa jumlah sampel yang dibutuhkan untuk mendapatkan peluang 50% mendapatkan dua orang dengan ulang tahun yang sama ? Dengan logika sederhana dan tanpa perhitungan hati-hati, kita mungkin berpikir butuh 183 sampel (separuh dari 366). Tapi apakah benar seperti itu?

Berdasarkan teori peluang, kejadian diambilnya orang pertama, kedua, dan seterusnya merupakan kejadian “saling bebas” yang berarti pengambilan sampel pertama tidak mempengaruhi peluang pengambilan sampel selanjutnya.

Untuk dua kejadian saling bebas, berlaku rumus

$$P(A \cap B) = P(A)P(B)$$

365n(365 – n)!

Dan dengan prinsip komplemen

$$P(x) = 1 - \neg P(x)$$

Kita dapat menghitung

$$\neg P(x) = \frac{365}{365} x \frac{364}{365} x \ldots x \frac{365 - (n-1)}{365}$$

$$\neg P(x) = \frac{365!}{365^n (365 - n)!}$$

Kemudian kita menghitung jika kita ingin memiliki peluang 50% untuk mendapatkan dua orang dengan ulang tahun yang sama, maka

$$\frac{1}{2} = \frac{365!}{365^n (365 - n)!}$$

Dan nilai n yang diperoleh adalah 23. Sangat jauh dari 366 atau 183. Hal itu berarti, kita hanya membutuhkan 6% dari total sampel untuk mendapatkan peluang 50% menemukan dua orang dengan ulang tahun yang sama.

Teori inilah yang digunakan dalam pencarian kunci dengan exhaustive search. Kriptanalis tidak perlu memiliki peluang 100% untuk menemukan kunci, ia hanya butuh peluang 50% untuk menemukan kunci yang artinya jika ia beruntung, ia dapat mendapatkan kunci pada pencarian awal.

3.1.4 Zombie Komputer

Perkembangan internet yang sangat pesat memberikan keuntungan kriptanalis dalam menganalisis kunci, dengan memanfaatkan zombie komputer. Zombie computer adalah computer yang telah disisipi software/spyware/Trojan yang membuat seorang hacker mampu mengendalikan computer tersebut tanpa diketahui pemilik computer itu sendiri.

Seorang hacker dapat menyebarkankan software zombienya melalui spamming email, file mp3, link situs, gambar, dan media lain yang tidak mencurigakan. Software itu akan menginstall dirinya sendiri seperti virus, dan siap dipanggil oleh si hacker sewaktu-waktu.

Zombie computer sangatlah banyak di dunia ini, dan computer anda saat ini mungkin telah terinfeksi zombie software tanpa diketahui. Dan ini dapat dibuktikan dengan banyaknya kasus cyber crime dengan metode *Denial of Service* (DoS) yang memerintahkan semua computer zombie untuk mengakses sebuah situs ecara serentak sehingga menyebabkan server situs tersebut menjadi down.

Idenya adalah, kriptanalis membagi proses exhaustive search menjadi beberapa bagian (dapat menjadi sangat banyak sekali, bahkan lebih dari 1 juta), dan memerintahkan zombie computer memrosesnya dan mengirimkan hasilnya pada kriptanalis. Dengan demikian, seolah-olah kriptanalis menggunakan lebih dari 1 juta processor secara bersamaan. Dapat dibayangkan bagaimana dahsyat efeknya dalam menghemat waktu pencarian kunci dengan exhaustive search, dan bahwa kalkulasi waktu secara sederhana tidak berjalan disini.

4. VIGENERE BIT SHIFT

Jika kita bandingkan algoritma enkripsi klasik dan modern, sebenarnya algoritma enkripsi klasik lebih "baik" karena algoritma enkripsi modern mengandalkan kekuatannya pada infeasibility bexhaustive search yang ternyata tidak sekuat yang dibayangkan. Dengan menggunakan 4 metode yang telah dibahas pada subbab sbelumnya, seorang kriptanalis mampu memecahkan sebuah kunci 512 bit dalam hitungan bulan. Sebuah fakta yang sangat mengejutkan, bahkan system keamanan Playstation3 yang dinyatakan unhackable dapat dihack oleh George Hotz, seorang pemuda 17 tahun, hanya dalam waktu 5 minggu.

Dalam makalah ini, dicetuskan ide algoritma enkripsi baru yang semoga mampu memberi paradigm baru dengan mencoba menerapkan prinsip-prinsip algoritma enkripsi klasik pada algoritma enkripsi modern.

4.1 Konsep

Konsep dari pembuatan algoritma ini adalah penerapan algoritma vigenere cipher dalam algoritma enkripsi modern. Realisasinya berupa penggunaan mode bit dalam proses vigenere. Jika vigenere klasik

menggunakan 26 karakter, atau 36 alphanumerik, maka Vigenere Bit Shift menggunakan 256 ASCII symbol.

Dengan demikian, Proses Enkripsi dan Dekripsi akan dilakukan dalam modulo 256. Tapi hal ini saja tidaklah cukup, karena teknik analisis frekuensi masih dapat diterapkan untuk menganalisis cipherteks.

Salah satu solusinya adalah dengan menggunakan subtitusi homofonik. Dalam table ASCII yang terlampir, kita dapat melihat bahwa hanya sebagian simbol ASCII yang umum dipakai. Yaitu ASCII 33-146 (alphanumeric dan tanda baca) dan beberapa tambahan lain yang merupakan simbol matematika.

Symbol ASCII lainnya sangat jarang digunakan, sehingga kita dapat menggunakannya sebagai domain dari subtitusi homofonik. Subtitusi homofonik adalah menyubtitusikan sebuah symbol menjadi satu atau lebih symbol lain, sehingga diharapkan kemunculan rata-rata semua symbol adalah sama, yang akan membuat teknik analisis frekuensi menjadi tak berguna.

Maka kita perlu membuat table frekuensi kemunculan dengan memperhitungkan tanda baca, spasi, huruf, angka, dan symbol lain. Misal dalam bahasa inggris didapatkan bahwa karakter spasi memiliki frekuensi paling tinggi, yaitu sebesar 15%, kemudian disusul karakter E sebesar 7%, dst. Maka kita memetakan byte spasi menjadi 15 byte berbeda. Misal spasi akan disubtitusikan dengan byte 4, 23, 27, 101, 115, 156, 178, 190, 192, 201, 203, 220, 240, 250, 254. Dan E menjadi 7 byte berbeda, dst. Untuk byte yang hanya memiliki frekuensi di bawah 1 %, maka hanya dipetakan menjadi 1 byte.

Proses dekripsinya dengan mensubtitusikan ke 15 byte yang berkorespondensi dengan spasi menjadi karakter spasi, dst. Hal ini sangatlah mungkin karena tersedia 256 kemungkinan karakter ASCII, sedangkan yang dibutuhkan hanya 150an lebih karakter sebagai kodomainnya.

Dengan demikian, seandainya dilakukan analisis frekuensi, maka akan didapatkan frekuensi kemunculan yang merata.

Selain itu, kunci yang digunakan haruslah panjang, Kunci dapat berupa suatu file yang mengandung ribuan byte. Sehingga algoritma ini akan menyerupai one time pad yang dinyatakan “unbreakable”.

5. KESIMPULAN

Algoritma enkripsi modern yang menyandarkan kekuatannya pada kesulitan melakukan exhaustive search buaknlah algoritma eknkripsi terbaik. Sejarah telah membuktikan, tak ada satupun algoritma enkripsi modern yang mampu bertahan lebih dari 1 abad, tetapi algoritma vigenere mampu bertahan selama 2 abad hingg akhirnya kasiski menemukan cara membongkar algoritma tersebut.

Konsep vigenere bit shift yang disajikan dalam makalah ini mungkin sangat sederhana dan diragukan kekuatannya, tetapi yang ingin ditekankan dalam makalah ini adalah ide akan sebuah paradigm kriptografi baru yang menggabungkan algoritma enkripsi klasik yang penuh intrik, dengan algoritma enkripsi modern yang berjalan dalam mode bit.

6. DAFTAR PUSTAKA

[1] Munir, Ir. Rinaldi. *Diktat kuliah Kriptografi*. 2006. Bandung.

[2] Lau, Tak Wing Edward. *Super Heuristic and Their Applications to Combinatorial Problems*. 1999. Asian Journal of Control

[3] Amanda, Magdalena Marlin. *Study Birthday Attack*. 2009. Bandung

[4] http://johnbokma.com/mexit/2006/01/16/zombie-comment-spam-referer-spam.html

[5] http://computer.howstuffworks.com/zombie-computer.htm

[6] http://nexus404.com/Blog/2010/01/27/unhackable-playstation-3-hacked-by-george-hotz-ps3-cracked-by-iphone-jailbraker-supposedly-plays-older-ps2-games-or-ps3-pirated-games/

LAMPIRAN

Lower Table

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00		32	20		64	40	@	96	60	`
1	01	☺	33	21	!	65	41	A	97	61	a
2	02	☻	34	22	"	66	42	B	98	62	b
3	03	♥	35	23	#	67	43	C	99	63	c
4	04	♦	36	24	$	68	44	D	100	64	d
5	05	♣	37	25	%	69	45	E	101	65	e
6	06	♠	38	26	&	70	46	F	102	66	f
7	07	•	39	27	'	71	47	G	103	67	g
8	08	◘	40	28	(	72	48	H	104	68	h
9	09	○	41	29	)	73	49	I	105	69	i
10	0A	◙	42	2A	*	74	4A	J	106	6A	j
11	0B	♂	43	2B	+	75	4B	K	107	6B	k
12	0C	♀	44	2C	,	76	4C	L	108	6C	l
13	0D	♪	45	2D	-	77	4D	K	109	6D	m
14	0E	♫	46	2E	.	78	4E	N	110	6E	n
15	0F	☼	47	2F	/	79	4F	O	111	6F	o
16	10	►	48	30	0	80	50	P	112	70	p
17	11	◄	49	31	1	81	51	Q	113	71	q
18	12	↕	50	32	2	82	52	R	114	72	r
19	13	‼	51	33	3	83	53	S	115	73	s
20	14	¶	52	34	4	84	54	T	116	74	t
21	15	§	53	35	5	85	55	U	117	75	u
22	16	▬	54	36	6	86	56	V	118	76	v
23	17	↨	55	37	7	87	57	W	119	77	w
24	18	↑	56	38	8	88	58	X	120	78	x
25	19	↓	57	39	9	89	59	Y	121	79	y
26	1A	→	58	3A	:	90	5A	Z	122	7A	z
27	1B	←	59	3B	;	91	5B	[	123	7B	{
28	1C	∟	60	3C	<	92	5C	\	124	7C	|
29	1D	↔	61	3D	=	93	5D	]	125	7D	}
30	1E	▲	62	3E	>	94	5E	^	126	7E	~
31	1F	▼	63	3F	?	95	5F	_	127	7F	⌂

Upper Table

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	└	224	E0	α
129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
131	83	â	163	A3	ú	195	C3	├	227	E3	π
132	84	ä	164	A4	ñ	196	C4	─	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	═	237	ED	φ
142	8E	Ä	174	AE	«	206	CE	╬	238	EE	ε
143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
144	90	É	176	B0	░	208	D0	╨	240	F0	≡
145	91	æ	177	B1	▒	209	D1	╤	241	F1	±
146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
147	93	ô	179	B3	│	211	D3	╙	243	F3	≤
148	94	ö	180	B4	┤	212	D4	╘	244	F4	⌠
149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	°
153	99	Ö	185	B9	╣	217	D9	┘	249	F9	∙
154	9A	Ü	186	BA	║	218	DA	┌	250	FA	·
155	9B	¢	187	BB	╗	219	DB	█	251	FB	√
156	9C	£	188	BC	╝	220	DC	▄	252	FC	ⁿ
157	9D	¥	189	BD	╜	221	DD	▌	253	FD	²
158	9E	₧	190	BE	╛	222	DE	▐	254	FE	■
159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	