Bystar Block Cipher

Tugas 2 IF4020 Kriptografi

Bintang Fajarianto - 13519138
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: 13519138@std.stei.itb.ac.id

***Abstrak*—Pada era modern ini, dikenal algoritma cipher blok, yaitu algoritma enkripsi yang diterapkan pada setiap blok bit berukuran sama pada *plain text*. Makalah ini membahas mengenai rancangan sebuah cipher blok baru bernama Bystar. Bystar merupakan salah satu algoritma cipher blok yang mengimplementasikan jaringan feistel dengan jumlah putaran sebanyak 16 kali dan memiliki 128-bit kunci yang beroperasi pada 128-bit blok data. Berdasarkan hasil eksperimen yang telah dilakukan, algoritma ini memiliki efek longsoran dibuktikan dengan perbedaan *cipher text* terhadap perubahan 1-bit kunci ataupun *plain text* serta ketahanan terhadap analisis frekuensi yang baik.**

Kata kunci—kriptografi; cipher blok; bystar; confussion; diffusion; jaringan feitsel;

I. PENDAHULUAN

Perkembangan teknologi digital yang pesat pada masa ini terutama dalam hal penyebaran dan pengolahan informasi membuat keamanan informasi tersebut menjadi aspek yang sangat penting. Terbukanya jalan menuju kemudahan dalam bertukar informasi diiringi dengan terbukanya jalan baru untuk menyadap informasi-informasi yang bersifat privasi. Oleh karena itu, diperlukan suatu mekanisme agar informasi yang dipertukarkan secara digital terjamin keamanannya.

Kriptografi merupakan teknik yang digunakan untuk berkomunikasi secara aman di lingkungan yang terdapat pihak ketiga[1]. Kriptografi juga merupakan bidang ilmu yang mempelajari proses enkripsi suatu pesan dalam berbagai media informasi, seperti teks, suara, gambar, serta video.

Konsep pengamanan data dengan kriptografi telah ada sejak ribuan tahun yang lalu. Caesar cipher adalah salah satu algoritma sederhana dari kriptografi klasik yang terkenal. Kriptografi terus berkembang dan algoritma yang dihasilkan semakin kompleks sehingga sulit untuk dipecahkan kuncinya. Saat ini, algoritma Rijndael telah ditetapkan sebagai standar algoritma kriptografi modern atau yang dikenal sebagai AES.

Pada makalah ini, telah dirancang sebuah algoritma kriptografi simetri cipher blok yang baru dengan menerapkan prinsip *confusion* dan *diffusion* oleh Shannon. Algoritma ini menerapkan jaringan feistel dengan jumlah putaran sebanyak 16 kali serta menggunakan 128-bit kunci yang beroperasi pada 128-bit blok data.

II. LANDASAN TEORI

A. Cipher Blok

Cipher blok adalah suatu algoritma untuk melakukan enkripsi dan dekripsi. Dalam prosesnya, *plain text* dibagi menjadi blok-blok bit yang berukuran sama dan proses enkripsi dilakukan pada setiap blok tersebut. Cipher blok menggunakan *symmetric key* atau kunci simetris yang berarti kunci untuk melakukan proses enkripsi maupun dekripsi merupakan kunci yang sama. Saat ini, terdapat beragam jenis algoritma cipher blok yang ada, seperti GOST, DEST, AES, dan lain-lain.

B. Confusion dan Diffusion

Prinsip *confusion* dan *diffusion* pertama kali diperkenalkan oleh Claude Shannon pada tahun 1945. Keduanya saling berkaitan serta digunakan untuk mempersulit dipecahkannya sebuah *cipher text* dengan serangan berbasis statistik.

1) Confusion

Prinsip *confusion* bertujuan untuk menyembunyikan hubungan antara huruf pada *plain text* terhadap huruf pada *cipher text* yang berkoresponden. Prinsip ini mengatakan bahwa setiap bit pada *cipher text* harus bergantung pada beberapa bagian dari kunci yang digunakan. Dengan demikian, perubahan satu bit pada kunci akan menyebabkan perubahan pada sebagian atau seluruh bit pada *cipher text*. Salah satu teknik yang dapat digunakan untuk merealisasikan prinsip ini adalah teknik subtitusi.

2) Diffusion

Prinsip *diffusion* bertujuan untuk menyebarkan pengaruh dari bit dalam *plain text* sebanyak mungkin pada bit dalam *cipher text*. Prinsip ini mengatakan bahwa pengaruh dari perubahan satu bit pada plain text harus terdifusi ke sebanyak mungkin bit pada *cipher text*. Dengan demikian, perubahan yang drastis akan terjadi pada *cipher text* apabila satu atau lebih karakter dalam *plain text* berubah. Salah satu teknik yang dapat digunakan untuk merealisasikan prinsip ini adalah teknik permutasi.

C. Mode Operasi

Dalam melakukan proses enkripsi dan dekripsi pada sebuah cipher blok, terdapat keterkaitan antar blok yang didefinisikan

dalam beberapa mode. Pada algoritma Bystar, terdapat tiga dari lima mode cipher blok yang dapat dipilih dalam melakukan proses enkripsi. Berikut ini adalah mode-mode yang diimplementasikan dalam algoritma Bystar.

1) ECB

Pada mode ECB, sebuah blok pada *plain text* yang akan dienkripsi bersifat independen terhadap blok yang lain. Hasil enkripsi dari suatu blok tidak akan memengaruhi hasil enkripsi blok berikutnya sehingga blok *plain text* yang sama akan dienkripsi menjadi *cipher text* yang sama pula.

2) CBC

Pada mode CBC, sebuah blok pada *plain text* memiliki ketergantungan terhadap blok yang lain. Hasil enkripsi dari suatu blok akan digunakan pada proses enkripsi blok berikutnya. Untuk blok pertama, diperlukan sebuah umpan untuk melakukan enkripsi yang dapat disebut sebagai *initialization vector*.

3) Counter

Pada mode Counter, tidak ada dependensi antar blok seperti pada mode CBC. Mode ini menggunakan sebuah nilai *counter* berupa blok bit berukuran sama dengan blok *plain text* yang akan terus bertambah setiap melakukan enkripsi pada suatu blok. Nilai *counter* ini diinisialisasi secara acak dan dienkripsi, kemudian hasil enkripsi tersebut akan dilakukan operasi XOR dengan blok *plain text* untuk menghasilkan blok *cipher text*.

D. *Jaringan Feistel*

Jaringan Feistel adalah sebuah struktur simetris yang digunakan dalam menyusun suatu cipher blok. Struktur jaringan ini bersifat *reversible*, yang berarti hanya diperlukan satu algoritma untuk melakukan enkripsi dan dekripsi. Dengan adanya sifat ini, perancang cipher blok akan lebih mudah dalam membangun cipher blok yang dirancang dengan tidak membuat algoritma baru untuk melakukan dekripsi.

Pada Gambar 1, terdapat sebuah fungsi F yang sama untuk melakukan enkripsi dan dekripsi. Fungsi ini dapat dibuat serumit mungkin karena bersifat independen terhadap jumlah iterasi yang dilakukan.

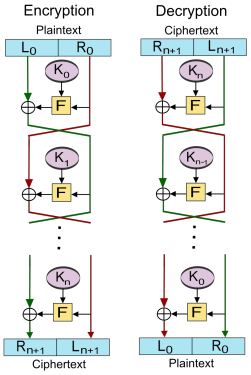

Fig. 1. Jaringan Feistel

III. Rancangan Block Cipher Baru

A. *Ikhtisar Algoritma*

Algoritma Bystar adalah algoritma cipher blok yang mampu mengenkripsi pesan per blok berukuran 128-bit dengan panjang kunci 128-bit. Algoritma ini menggunakan jaringan feitsel dengan jumlah putaran sebanyak 16 kali dalam proses enkripsi maupun dekripsi.

B. *Pembangkitan Kotak Permutasi*

Kotak permutasi digunakan untuk melakukan transposisi pada setiap blok *plain text*. Pada algoritma Bystar, kotak permutasi merupakan sebuah larik berukuran 128 elemen yang terbentuk dari gabungan matriks IP dan matriks IP^{-1} pada algoritma DES. Elemen dari matriks IP yang berukuran 64 elemen disusun berselang-seling dengan elemen dari matriks IP^{-1} yang juga berukuran 64 elemen sehingga terbentuk sebuah larik baru berukuran 128 elemen.

Kotak permutasi terdiri dari elemen-elemen yang menyatakan indeks untuk melakukan transposisi terhadap setiap bit dalam blok *plain text*. Proses transposisi dilakukan dengan menyusun ulang urutan bit pada *plain text* sesuai dengan nilai indeks yang terdapat dalam kotak permutasi. Selain itu, inverse dari kotak permutasi juga perlu dibangkitkan untuk melakukan pembalikan transposisi yang telah dilakukan. Berikut ini adalah gambar dari kotak permutasi dan invers dari kotak permutasi yang digunakan dalam algoritma Bystar.

58	104	50	72	42	112	34	80	26	120	18	88	10	128	2	96
60	103	52	71	44	111	36	79	28	119	20	87	12	127	4	95
62	102	54	70	46	110	38	78	30	118	22	86	14	126	6	94
64	101	56	69	48	109	40	77	32	117	24	85	16	125	8	93
57	100	49	68	41	108	33	76	25	116	17	84	9	124	1	92
59	99	51	67	43	107	35	75	27	115	19	83	11	123	3	91
61	98	53	66	45	106	37	74	29	114	21	82	13	122	5	90
63	97	55	65	47	105	39	73	31	113	23	81	15	121	7	89

Fig. 2. Kotak Permutasi

79	15	95	31	111	47	127	63	77	13	93	29	109	45	125	61
75	11	91	27	107	43	123	59	73	9	89	25	105	41	121	57
71	7	87	23	103	39	119	55	69	5	85	21	101	37	117	53
67	3	83	19	99	35	115	51	65	1	81	17	97	33	113	49
116	100	84	68	52	36	20	4	120	104	88	72	56	40	24	8
124	108	92	76	60	44	28	12	128	112	96	80	64	48	32	16
114	98	82	66	50	34	18	2	118	102	86	70	54	38	22	6
122	106	90	74	58	42	26	10	126	110	94	78	62	46	30	14

Fig. 3. Invers Kotak Permutasi

C. *Pembangkitan Kotak-S*

Kotak-S digunakan untuk melakukan subtitusi dari sejumlah m_1 bit menjadi m_2 bit dengan ketentuan $m_2 > m_1$.

Pada algoritma Bystar, hanya terdapat satu kotak-S yang merupakan sebuah larik berukuran 256 elemen. Kotak-S ini terbentuk dari bilangan terurut 255 hingga 0 yang diacak menggunakan *random seed* khusus, yaitu 230501 yang merupakan tanggal ulang tahun pembuat algoritma.

Dalam penggunaannya, mula-mula setiap 8-bit dalam satu blok *plain text* akan diinterpretasi sebagai suatu bilangan. Kemudian, bilangan tersebut akan digunakan sebagai indeks dari larik kotak-S sehingga bilangan tersebut akan disubtitusi menjadi bilangan yang lain. Berikut ini adalah gambar dari kotak-S yang digunakan dalam algoritma Bystar.

83	100	232	159	92	91	22	147	255	116	162	180	52	240	58	207
121	3	156	35	67	184	114	55	10	6	75	117	106	72	130	85
46	250	119	32	40	186	115	178	27	182	222	199	175	170	54	57
63	15	163	234	20	2	249	43	242	172	128	21	47	224	196	209
188	228	236	0	42	131	192	11	201	93	167	108	203	204	65	235
137	148	60	253	231	221	98	118	223	68	238	50	213	185	110	154
227	95	165	31	153	246	76	160	8	30	189	140	107	56	244	84
136	123	145	24	229	177	139	127	239	39	157	215	79	86	105	216
200	120	96	158	198	111	149	126	146	104	61	112	132	33	66	101
179	45	81	252	220	176	48	29	78	211	13	73	28	171	193	208
74	187	183	169	202	77	210	53	103	247	80	82	38	197	205	9
161	69	1	90	51	89	34	248	230	37	16	64	71	18	174	44
164	173	218	195	144	166	17	190	41	26	150	102	70	23	237	7
88	87	124	254	152	191	62	133	217	168	99	151	125	194	109	206
226	233	59	155	12	135	25	36	181	214	212	19	49	134	141	245
4	14	113	241	5	225	138	122	251	94	219	143	243	142	129	97

Fig. 4. Kotak-S

D. Rancangan Algoritma Pembangkitan Kunci

Pada algoritma Bystar, terdapat 16 putaran dalam melakukan enkripsi dan dekripsi sehingga dibutuhkan 16 kunci yang berbeda dalam setiap putarannya, atau dapat disebut sebagai kunci internal. Kunci-kunci internal dibangkitkan dari kunci eksternal berukuran 128-bit.

Dalam proses pembangkitan kunci internal, mula-mula kunci eksternal akan dipermutasi dengan sebuah tabel permutasi yang terbentuk dari bilangan terurut 1 hingga 128 yang diacak menggunakan *random seed* khusus, yaitu 230501 yang merupakan tanggal ulang tahun pembuat algoritma. Pada setiap iterasi, tabel permutasi ini akan diacak dengan *random seed* spesifik, yaitu bilangan iterasi saat ini sehingga proses permutasi pada setiap iterasi akan menggunakan tabel permutasi yang berbeda. Berikut ini adalah gambar dari tabel permutasi awal yang digunakan dalam algoritma Bystar dalam pembangkitan kunci-kunci internal.

28	43	107	9	11	102	127	82	76	50	35	98	103	89	29	92
34	23	17	125	63	5	83	120	78	119	2	87	37	69	109	26
93	91	36	122	18	73	48	14	114	101	27	39	75	54	49	108
100	47	65	128	71	30	24	86	32	60	40	115	118	68	55	4
13	113	70	94	41	88	46	42	117	97	56	45	112	90	105	95
77	106	10	121	84	85	96	1	52	33	62	20	44	79	53	66
31	74	25	15	12	99	51	116	111	38	21	22	110	104	61	58
6	124	72	8	123	16	59	67	3	81	19	126	7	57	64	80

Fig. 5. Tabel Permutasi Awal

Pada tahap berikutnya, hasil proses permutasi yang telah dilakukan akan dibagi menjadi empat bagian sama besar, masing-masing berukuran 32-bit. Keempat bagian tersebut akan dilakukan operasi XOR dengan ketentuan bagian pertama dengan bagian ketiga serta bagian kedua dengan bagian keempat. Hasil operasi XOR bagian ganjil akan digeser kekiri sejauh bilangan iterasi saat ini dan hasil operasi XOR bagian genap akan digeser ke kanan sejauh bilangan iterasi saat ini. Selanjutnya, kedua bagian yang telah digeser digabungkan kembali menjadi suatu blok yang utuh. Berikut ini adalah skema pembangkitan kunci internal pada algoritma Bystar.

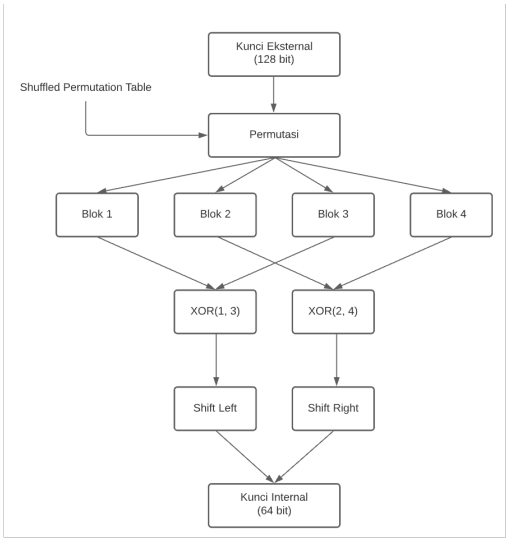

Fig. 6. Skema Algoritma Pembangkitan Kunci Internal

Hasil dari proses pembangkitan kunci internal ini adalah 16 buah kunci yang masing-masing berukuran 64 bit. Perubahan ukuran kunci terhadap kunci eksternal ini terjadi saat empat bagian hasil proses permutasi dilakukan operasi XOR menjadi dua bagian saja. Kunci-kunci ini akan digunakan dalam setiap putaran dalam proses enkripsi dan dekripsi.

E. *Rancangan Algoritma Enkripsi dan Dekripsi*

1) (Optional) Konversi Heksadesimal Menjadi Teks

Algoritma Bystar menyediakan opsi jika dalam proses dekripsi, *cipher text* yang diterima berada dalam bentuk heksadesimal.

2) Konversi Teks Menjadi Representasi Biner

Pada proses enkripsi ataupun dekripsi, *plain text* ataupun *cipher text* akan dikonversi menjadi representasi biner berbasis 8. Representasi biner akan dibagi menjadi beberapa blok berukuran 128-bit untuk dieksekusi secara sekuensial tiap blok.

3) Permutasi Awal

Pada algoritma Bystar, representasi biner pada proses enkripsi ataupun dekripsi akan dipermutasi dengan kotak permutasi pada Gambar 2.

4) Pemisahan Representasi Biner Menjadi Dua Bagian

Setelah dilakukan permutasi awal, representasi biner akan dibagi menjadi dua bagian (kiri dan kanan), masing-masing berukuran 64-bit. Salah satu bagian akan dioperasikan ke dalam fungsi putaran, sedangkan bagian yang lain akan dioperasikan pada iterasi setelahnya.

5) Fungsi Putaran

Pada algoritma Bystar, terdapat 16 putaran yang dilakukan dalam proses enkripsi maupun proses dekripsi. Algoritma ini menerapkan jaringan feistel yang didalamnya terdapat fungsi putaran yang cukup kompleks.

Pada awalnya, representasi biner berukuran 64-bit akan dibagi menjadi dua bagian berukuran sama. Bagian kiri akan digeser ke kiri dan bagian kanan akan digeser ke kanan sejauh iterasi modulo panjang bagian representasi biner tersebut.

Pada tahap selanjutnya, kedua bagian yang telah digeser digabungkan kembali menjadi representasi biner berukuran 64-bit. Representasi biner dikonversi menjadi larik bilangan bulat untuk dilakukan operasi XOR dengan kunci internal. Kemudian setelahnya akan dilakukan operasi subtitusi dengan kotak-S pada Gambar 4.

Pada tahap selanjutnya, larik bilangan bulat dikonversi kembali mepresentasi biner berukuran 64-bit untuk dibagi menjadi dua bagian berukuran sama. Bagian kiri akan digeser ke kanan dan bagian kanan akan digeser ke kiri sejauh iterasi modulo panjang bagian representasi biner tersebut.

Berikut ini adalah skema fungsi putaran untuk memahami proses yang terjadi dalam fungsi putaran pada algoritma Bystar.

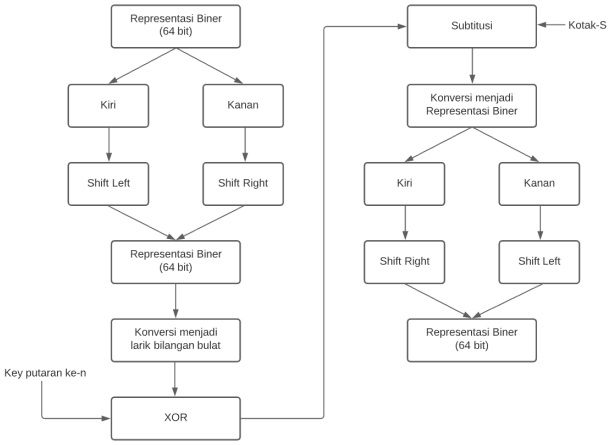

Fig. 7. Skema Fungsi Putaran pada Iterasi ke-n

6) Penggabungan Dua Bagian Representasi Biner

Setelah dilakukan operasi dalam 16 putaran, dua bagian (kiri dan kanan) representasi biner digabungkan kembali untuk dilakukan proses permutasi.

7) Permutasi Akhir

Pada algoritma Bystar, representasi biner yang dihasilkan setelah 16 putaran pada proses enkripsi ataupun dekripsi akan dipermutasi dengan invers kotak permutasi pada Gambar 3.

8) Konversi Representasi Biner Menjadi Teks

Pada proses enkripsi ataupun dekripsi, representasi biner berbasis 8 akan dikonversi kembali menjadi *plain text* ataupun *cipher text*.

9) (Optional) Konversi Teks Menjadi Heksadesimal

Algoritma Bystar juga menyediakan opsi jika dalam proses enkripsi, *cipher text* yang diharapkan berada dalam bentuk heksadesimal. Opsi ini dibutuhkan untuk keperluan analisis efek longsoran (*avalanche effect*).

IV. Eksperimen dan Pembahasan Hasil

Rancangan algoritma Bystar diimplementasikan menggunakan Bahasa pemrograman Python 3. Program ini menerima masukan *plain text*, *cipher text*, dan *key* dalam bentuk string serta mampu menghitung waktu eksekusi proses enkripsi dan dekripsi file dengan panjang yang beragam. Program ini mampu menampilkan hasil enkripsi dalam bentuk string unicode serta string heksadesimal untuk mempermudah proses analisis cipher yang dirancang.

A. Analisis Waktu Proses Enkripsi dan Dekripsi

Eksperimen ini menggunakan empat file yang memiliki ukuran berbeda-beda. File ini memiliki kategori *small*, *medium*, *large*, dan *very large* yang secara terurut berukuran 256 bytes, 1024 bytes, 4096 bytes, dan 16384 bytes. Hasil uji coba dapat dilihat pada tabel berikut.

TABLE I. HASIL UJI COBA WAKTU PROSES ENKRIPSI DAN DEKRIPSI ALGORITMA BYSTAR MODE ECB

Key	Performansi (Mode ECB)		
	Kategori (ukuran file)	**Waktu Enkripsi (s)**	**Waktu Dekripsi (s)**
128 bit	*Small* (256 bytes)	0.01147	0.01153
128 bit	*Medium* (1024 bytes)	0.04606	0.04611
128 bit	*Large* (4096 bytes)	0.18338	0.18311
128 bit	*Very Large* (16384 bytes)	0.72343	0.71796

[a] 128-bit *key* yang digunakan pada setiap percobaan sama, yaitu "Bystar138Cipherz"

TABLE II. HASIL UJI COBA WAKTU PROSES ENKRIPSI DAN DEKRIPSI ALGORITMA BYSTAR MODE CBC

Key	Performansi (Mode CBC)		
	Kategori (ukuran file)	**Waktu Enkripsi (s)**	**Waktu Dekripsi (s)**
128 bit	*Small* (256 bytes)	0.01217	0.01200
128 bit	*Medium* (1024 bytes)	0.04879	0.04898
128 bit	*Large* (4096 bytes)	0.19421	0.19538
128 bit	*Very Large* (16384 bytes)	0.76504	0.76692

[a] 128-bit *key* yang digunakan pada setiap percobaan sama, yaitu "Bystar138Cipherz"

TABLE III. HASIL UJI COBA WAKTU PROSES ENKRIPSI DAN DEKRIPSI ALGORITMA BYSTAR MODE COUNTER

Key	Performansi (Mode Counter)		
	Kategori (ukuran file)	**Waktu Enkripsi (s)**	**Waktu Dekripsi (s)**
128 bit	*Small* (256 bytes)	0.01245	0.01248
128 bit	*Medium* (1024 bytes)	0.05061	0.05069
128 bit	*Large* (4096 bytes)	0.20309	0.20160
128 bit	*Very Large* (16384 bytes)	0.79280	0.79130

[a] 128-bit *key* yang digunakan pada setiap percobaan sama, yaitu "Bystar138Cipherz"

B. *Analisis Ruang Kunci*

Algoritma Bystar dengan kunci sepanjang 128 bit memiliki ruang kunci sebesar 2^{128} atau 3,4 x 10^{38}. Dengan ruang pencarian sebesar itu, jika algoritma ini diserang menggunakan serangan *brute force* dengan kecepatan 1 juta serangan per detik, maka akan dibutuhkan waktu selama 3,4 x 10^{32} detik atau setara 10,7 x 10^{24} tahun.

C. *Analisis Efek Longsoran (Avalanche Effect)*

Eksperimen ini bertujuan untuk membandingkan perbedaan hasil enkripsi menggunakan masukan *plain text* yang mirip dengan *key* yang sama serta menggunakan masukan *plain text* yang sama dengan *key* yang mirip. Proses enkripsi akan menggunakan mode ECB, yaitu proses enkripsi untuk setiap blok dilakukan secara independent.

TABLE IV. ANALISIS EFEK LONGSORAN – *PLAIN TEXT*

Plain Text	***Key***	***Cipher text***
Bintang-13519138	Bystar138Cipherz	C29CC28844C2B8C294C28F48C2A94C186BC2A7780B2AC3B2
*intang-13519138		3946354CC28F380F60C2A1C2ADC293C390C2B86AC2A30C
Bint*ng-13519138		0303C2BAC285440E051133C3B6C28544C292607758
Bintang-*3519138		C2A6C2887EC2B937C29FC3B9C2A84E18C389C2A74B0B19C3A2
Bintang-1351*138		1FC2995EC2B9C2BFC28E51C2B87F19C38AC2A6601AC299C3A3

[a] Perubahan pada *plain text* ditandai dengan *highlight*.

TABLE V. ANALISIS EFEK LONGSORAN – *KEY*

Plain Text	***Key***	***Cipher text***
Bintang-13519138	Bystar138Cipherz	C29CC28844C2B8C294C28F48C2A94C186BC2A7780B2AC3B2
	*ystar138Cipherz	1CC3BDC3B30FC38F404BC2BC3FC39DC29D6E2332C3B032
	Byst*r138Cipherz	C2B9270B79C29C5036C297C3A62EC2BDC28F4DC28B5A40
	Bystar13*Cipherz	C2A7C2965C444FC2ACC2BD627A3A5E06C3A7C2907C1A
	Bystar138Cip*erz	C29C2200C2BAC290C3894C2D0C5A2FC3AF38432AC3B0

[a] Perubahan pada *key* ditandai dengan *highlight*.

Hasil eksperimen tersebut menunjukkan bahwa perubahan satu karakter pada *plain text* ataupun *key* mampu membuat hasil enkripsi yang berbeda dengan perubahan karakter yang acak sehingga sulit untuk diprediksi. Hal ini menyebabkan proses kriptanalisis dengan statistik sulit untuk dilakukan.

V. KESIMPULAN DAN SARAN

Algoritma Bystar dapat digunakan sebagai cipher blok untuk mengenkripsi dan mendekripsi sebuah pesan per blok berukuran 128-bit dengan panjang kunci 128-bit. Algoritma ini telah menerapkan prinsip-prinsip cipher blok yang baik, yaitu prinsip *confusion* dengan melakukan subtitusi, prinsip *diffusion* dengan melakukan permutasi, serta jaringan feistel beserta fungsi putaran yang kompleks.

Kelebihan dari algoritma ini diantaranya adalah kunci yang digunakan cukup fleksibel karena dalam algoritma ini diterapkan validasi kunci jika ukuran yang tidak berjumlah 128-bit. Jumlah putaran yang dilakukan dalam proses enkripsi ataupun dekripsi juga cukup banyak, yaitu sebanyak 16 kali dengan fungsi putaran yang kompleks sehingga dapat dikatakan aman. Berdasarkan hasil eksperimen, algoritma Bystar juga memiliki tingkat keamanan yang baik. Hasil analisis efek longsoran menunjukkan hasil yang baik sehingga proses kriptanalisis untuk memecahkan algoritma ini sulit dilakukan. Selain itu, algoritma ini memiliki ruang kunci yang besar sehingga membutuhkan waktu yang sangat lama jika diserang dengan algoritma *brute force*.

Kekurangan dari algoritma ini diantaranya adalah proses eksekusi yang kurang efisien. Dalam kode yang digunakan, masih banyak proses konversi dari atau menjadi representasi bit sehingga proses eksekusi menjadi lebih lambat. Selain itu, alangkah baiknya algoritma ini dapat diimplementasikan dengan bahasa pemrograman yang lebih baik dan lebih cepat dalam proses penanganan di level bit, seperti bahasa C++.

REFERENCES

[1] Rivest, Ronald L. (1990). "Cryptography". In J. Van Leeuwen (ed.). Handbook of Theoretical Computer Science. 1. Elsevier.

[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/13-Block-Cipher-Bagian1-2023.pdf (diakses pada 5 Maret 2023)

[3] https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/14-Prancangan-block-cipher-2023.pdf (diakses pada 5 Maret 2023)

[4] https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2022-2023/15-Beberapa-block-cipher-bagian1-2023.pdf (diakses pada 5 Maret 2023)