 doi.org/10.56706/ik.v19i1.114

Pemulihan Kunci pada *Simplified Data Encryption Standard* (S-DES) Melalui Serangan Aljabar: Studi Kasus

Fadila Paradise[1,*)], Santi Indarjani[2)]

1) Badan Siber dan Sandi Negara, fadila.paradise@bssn.go.id

2) Rekayasa Kriptografi, Politeknik Siber dan Sandi Negara, santi.indarjani@poltekssn.ac.id

Riwayat Artikel

Dikirim 12 November 2024
Diterima 19 Maret 2025
Diterbitkan 25 April 2025

Kata kunci:

serangan aljabar
S-DES
persamaan polinomial
algoritma XL

Abstrak

Serangan aljabar merupakan salah satu metode kriptanalisis yang dilakukan dengan mengubah sistem kriptografi menjadi ekspresi aljabar. Pada penelitian ini dilakukan serangan aljabar terhadap algortima Simplified Data Encryption Standard (S-DES) sebagai media pembelajaran. Serangan dilakukan dengan asumsi penyerang memiliki sepasang teks terang dan teks sandi yang saling berkorespondensi dan digunakan untuk mencari kunci input. Serangan dilakukan melalui dua tahap, yaitu membentuk sistem persamaan polinomial yang merepresentasikan bit-bit teks sandi pada S-DES, dan melakukan pencarian solusi terhadap sistem persamaan polinomial menggunakan algoritma Extended Linearization (XL). Tahap kedua dalam penelitian ini menghasilkan tiga solusi yang merepresentasikan tiga kemungkinan nilai kunci. Ketiga nilai kunci ini kemudian digunakan untuk melakukan dekripsi sehingga nilai kunci yang benar dapat ditentukan berdasarkan kesesuaian teks terang yang dihasilkan. Serangan aljabar pada penelitian ini membuat percobaan pencarian kunci yang semula sebanyak 2^{10} percobaan (*6*) menjadi tiga percobaan. Penelitian ini bisa menjadi referensi penerapan serangan aljabar pada algoritma sejenis.

Keywords:

algebraic attack
S-DES
polynomial equation
XL algorithm

Abstract

Algebraic attack is one of cryptanalysis methods that performed by converting a cryptographic system into an algebraic expression. In this research, the algebraic attack is implemented on Simplified Data Encryption Standard (S-DES) algorithm as a learning medium. The attack is carried out by assuming the attacker has a corresponding plaintext and ciphertext and used to find the input key. The attack is performed in two stages, forming a system of polynomial equations that represents ciphertext bits in S-DES, and searching for the solution of a system of polynomial equations using Extended Linearization (XL) algorithm. The second stage of this research produces three solutions that represent three possible key values. These three key values are then used to perform decryption so that the correct key value can be determined based on the suitability of the resulting plaintext. The algebraic attack in this research reduces the key search experiment as many as 2^{10} trials (brute force attack) into three trials. This research can be used as reference for the application of algebraic attacks on similar algorithms.

1. PENDAHULUAN

Serangan aljabar adalah teknik kriptanalisis yang menggunakan pendekatan linier dengan mengubah sistem kriptografi menjadi persamaan linier. Secara umum, ada dua langkah dalam

melakukan serangan aljabar, pertama menentukan sistem persamaan polinomial sebagai representasi aljabar dari sistem kripto (algoritma, teks terang, teks sandi, atau kunci), dan kedua menyelesaikan sistem persamaan yang telah diperoleh [1].

Serangan aljabar telah diimplementasikan dalam beberapa algoritma, seperti NTRU [2], DES [3], dan Simplified AES [4]. Dalam paper ini, serangan aljabar diimplementasikan dalam *Simplified* DES (S-DES), algoritma DES yang disederhanakan, yang dibuat oleh Profesor Edward Schaefer dari Universitas Santa Clara pada tahun 1996 dengan tujuan pendidikan [5]. Algoritma S-DES dipilih karena strukturnya yang sederhana, tetapi memiliki beberapa jenis fungsi (linier dan non-linier) sehingga dapat dijadikan media pembelajaran yang sederhana dan mendalam.

Serangan aljabar pada algoritma S-DES diawali dengan mengubah algoritma enkripsi/dekripsi dan algoritma penjadwalan kunci ke dalam ekspresi aljabar. Selanjutnya dilakukan simulasi algoritma penjadwalan kunci dan enkripsi dalam bentuk aljabar, dengan input nilai teks terang dan teks sandi yang diketahui, serta kunci berupa *unknown variable*. Simulasi ini menghasilkan delapan persamaan polinomial yang merepresentasikan delapan bit teks sandi.

Pencarian solusi terhadap delapan persamaan polinomial dilakukan menggunakan algoritma *Extended Linearization* (XL). Pada dasarnya algoritma XL terdiri dari tiga tahap, yaitu memperluas sistem persamaan, melinierisasi sistem persamaan, dan menyelesaikan sistem persamaan dengan pendekatan aljabar linier [6]. Algoritma XL efektif digunakan pada penelitian ini karena jumlah *unknown variable* yang dihasilkan jauh lebih banyak dari persamaan polinomial yang dihasilkan, sehingga sistem persamaan polinomial perlu diperluas.

Paper ini disusun dalam lima bagian yaitu pendahuluan, kerangka teoritis, penentuan sistem persamaan polinomial yang merupakan representasi aljabar dari algoritma S-DES, pemulihan kunci rahasia melalui pencarian solusi persamaan polinomial, dan simpulan. Hasil dari studi ini dapat memberikan gambaran utuh tentang penerapan serangan aljabar pada algoritma S-DES.

2. KERANGKA TEORITIS

2.1. Serangan Aljabar

Serangan aljabar merupakan serangan *known-plaintext* yang dilakukan dengan mengubah algoritma menjadi ekspresi aljabar. Serangan aljabar pertama kali diperkenalkan oleh Kipnis dan Shamir (1998) dalam sebuah makalah berjudul "*Cryptanalysis of The HFE Public Key Cryptosystem by Relinearization*". Kipnis dan Shamir menggunakan algoritma baru bernama "Relinierisasi" untuk memecahkan sistem persamaan non-linier dalam lapangan hingga.

Prinsip utama serangan aljabar adalah mengubah masalah dalam menyerang sistem kriptografi (seperti menemukan kunci rahasia) menjadi memecahkan sistem persamaan polinomial. Pada dasarnya, ada dua tahap yang dilakukan dalam serangan aljabar [1]:

1) Menentukan sistem persamaan polinomial dari algoritma target.
2) Mencari solusi dari sistem persamaan polinomial untuk memulihkan kunci input rahasia.

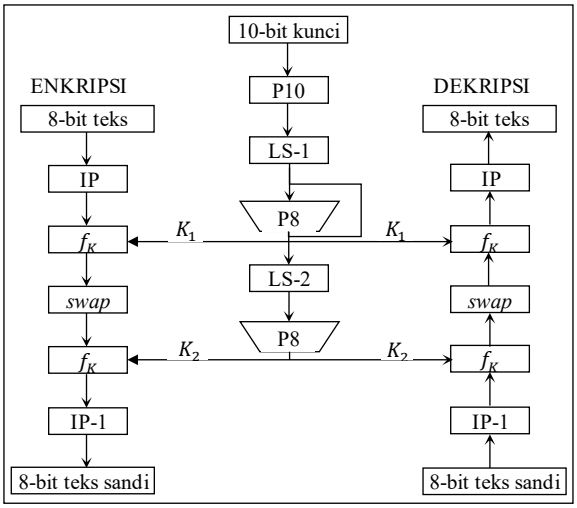
Serangan aljabar merupakan serangan berbasis teks terang yang diketahui (*known plaintext attack*) dengan asumsi penyerang memiliki sejumlah pasangan teks terang dan teks sandi yang berkorespondensi (dienkripsi menggunakan kunci rahasia yang sama). Dalam setiap proses pencarian kunci, digunakan satu pasang teks terang dan teks sandi yang berkorespondensi [1].

2.2. Algoritma Simplified DES (S-DES)

Simplified Data Encryption Standard (S-DES) adalah penyederhanaan dari DES, algoritma kriptografi yang ditetapkan sebagai standar dalam Standar Pemrosesan Informasi Federal (FIPS) untuk melindungi data rahasia [7]. S-DES diciptakan oleh Profesor Edward Schaefer pada tahun 1996 dengan tujuan pendidikan. S-DES memiliki karakter dan struktur yang mirip dengan DES dengan parameter yang lebih kecil [5]. Algoritma enkripsi S-DES terdiri dari dua putaran dengan ukuran blok data 8-bit dan sebuah kunci input berukuran 10-bit. Dua kunci putaran (masing-masing berukuran 8-bit) dibentuk menggunakan algoritma penjadwalan kunci. Skema dari algoritma S-DES dapat dilihat pada Gambar 1.

Sesuai algoritma DES, pada S-DES digunakan fungsi f yang terdiri dari fungsi *expand*, fungsi XOR, fungsi *s-box*, dan fungsi permutasi. Teks terang 8-bit akan dipermutasi dengan fungsi

permutasi IP, kemudian hasilnya dibagi dua bagian berukuran 4-bit. Blok bagian kanan yang masuk fungsi f akan diekspansi menjadi 8-bit untuk di-XOR dengan kunci putaran berukuran 8-bit. Selanjutnya setiap 4-bit akan masuk ke *s-box* berukuran 4×2 yang berbeda dinotasikan S0 dan S1. *Output* dari kedua *s-box* digabung membentuk string berukuran 4-bit yang akan dipermutasi oleh fungsi permutasi P4.



Gambar 1. Skema Algoritma S-DES

Output dari fungsi f di-XOR dengan setengah bagian kiri dari teks terang awal. Hasilnya akan ditukar posisinya (*swap*) untuk diproses pada putaran kedua dengan cara yang sama kecuali fungsi *swap* tidak digunakan lagi. *Output* dari fungsi f pada putaran kedua akan dipermutasi menggunakan fungsi IP-1 untuk menghasilkan teks sandi.

2.3. Sistem Persamaan Polinomial

Sebuah polinomial dalam x merupakan penjumlahan dari sejumlah hingga suku ax^n, dengan a adalah bilangan riil, n merupakan bilangan bulat non-negatif. Polinomial yang terdiri dari satu suku disebut monomial, terdiri dari dua suku berbeda disebut binomial, sedangkan tiga suku berbeda disebut trinomial. Persamaan polinomial dapat berbentuk persamaan linier, persamaan kuadrat, persamaan kubik, atau lainnya. Selanjutnya, sekumpulan persamaan polinomial yang terbatas disebut sistem persamaan polinomial [8].

2.4. Algoritma Extended Linearization (XL)

Algoritma XL dibuat pada tahun 2000 oleh Nicolas Curtois, Alexander Klimov, Jacques Patarin, dan Adi Shamir. Ide dasar algoritma XL adalah untuk membuat ekstensi sistem persamaan polinomial dengan melakukan perkalian dalam antara setiap persamaan polinomial dengan sejumlah monomial yang ada [9]. Berikut adalah langkah-langkah untuk algoritma XL [6]:

1) Menentukan jumlah monomial yang ada dan proyeksi jumlah persamaan polinomial yang dibutuhkan untuk mencari solusi persamaan.
2) Mengalikan seluruh persamaan polinomial yang diperoleh dari tahap pertama dengan monomial yang ditentukan, misalkan monomial dengan derajat kurang dari dua.
3) Mengubah sistem persamaan polinomial yang dibentuk menjadi sistem persamaan linier (linierisasi).
4) Menemukan solusi untuk sistem persamaan linier menggunakan substitusi dan eliminasi Gauss.

3. MENENTUKAN PERSAMAAN POLINOMIAL

Simulasi serangan pada penelitian ini menggunakan teks terang acak 01101101. Teks terang dienkripsi menggunakan kunci rahasia 1010001010 sehingga menghasilkan sepasang teks terang dan teks sandi yang berkorespondensi seperti terlihat pada Tabel 1.

Tabel 1. Skenario Simulasi

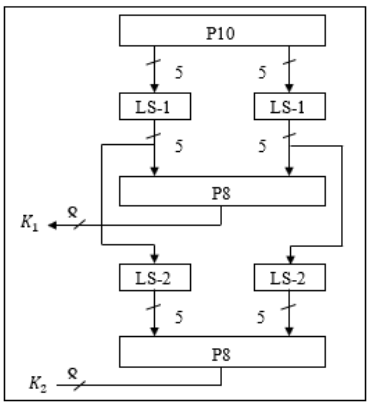
Kunci = 1010001010	
Teks Terang	Teks Sandi
01101101	01000110

Langkah awal adalah menentukan persamaan polinomal dari fungsi *s-box* yang digunakan, serta persamaan aljabar dari kunci putaran sesuai algoritma penjadwalan kunci yang digunakan. Selanjutnya, dengan menggunakan persamaan polinomial dari *s-box* dan persamaan dari kunci putaran, akan ditentukan persamaan aljabar dari teks sandi dengan menerapkan proses penyandian menggunakan teks terang yang dipilih di atas. Sesuai jumlah ukuran dari teks sandi yaitu 8-bit maka akan diperoleh delapan buah persamaan polinomial yang masing-masing merupakan representasi aljabar dari setiap bit teks sandi. Langkah-langkah yang dilakukan dalam pencarian sistem persamaan polinomial adalah sebagai berikut.

3.1. Menentukan Persamaan Polinomial Kunci Putaran 1 dan Kunci Putaran 2 Dinotasikan $K1$ dan $K2$

Misalkan digunakan sebuah kunci rahasia terdiri dari 10-bit yang dinotasikan dengan $K = k_1 \dots k_{10}$. Skema algoritma penjadwalan kunci (*Key Scheduling Algorithm* disingkat KSA) pada S-DES dapat dilihat pada Gambar 2.

Pada putaran 1, kunci input 10-bit akan masuk ke fungsi permutasi P10, kemudian selanjutnya dibagi dua dan masing-masing masuk ke fungsi LS-1 yaitu fungsi pergeseran 1-bit ke kiri. Misalkan 10010 digeser 1-bit ke kiri menjadi 00101. Hasil proses pergeseran dipermutasi menjadi 8-bit dengan fungsi permutasi P8, yang akan digunakan sebagai kunci putaran 1 yaitu K_1 .



Gambar 2. Skema Penjadwalan Kunci pada S-DES

Selanjutnya masing-masing keluaran dari fungsi LS-1 dimasukkan ke fungsi LS-2 yaitu pergeseran ke kiri sebanyak 2-bit. Misalkan string 00101 hasil dari LS-1 digeser 2-bit ke kiri menjadi 10100. Hasil dari LS-2 kemudian masuk ke fungsi permutasi P8 menghasilkan 8-bit sebagai kunci putaran 2 yaitu K_2. Tabel permutasi yang digunakan adalah tabel P8 dan P10 secara berurutan seperti terlihat pada Gambar 3. Pada proses pembentukan kunci putaran, K_1 dan K_2 dibangkitkan berdasarkan kunci input $K = k_1 \dots k_{10}$ seperti pada Gambar 4, dengan rincian $K_1 = k_1k_7k_9k_4k_8k_3k_{10}k_6$ dan $K_2 = k_8k_3k_6k_5k_{10}k_2k_9k_1$.

P8							
6	3	7	4	8	5	10	9

P10									
3	5	2	7	4	10	1	9	8	6

Gambar 3. Fungsi Permutasi pada KSA S-DES

Input	Proses Pembangkitan Kunci Putaran		Output
Kunci	$k_1k_2k_3k_4k_5k_6k_7k_8k_9k_{10}$		
P10	$k_3k_5k_2k_7k_4k_{10}k_1k_9k_8k_6$		
Partisi 2	$k_3k_5k_2k_7k_4$	$k_{10}k_1k_9k_8k_6$	
LS-1	$k_5k_2k_7k_4k_3$	$k_1k_9k_8k_6k_{10}$	
P8	$k_1k_7k_9k_4k_8k_3k_{10}k_6$		K_1
LS-2	$k_7k_4k_3k_5k_2$	$k_8k_6k_{10}k_1k_9$	
P8	$k_8k_3k_6k_5k_{10}k_2k_9k_1$		K_2

Gambar 4. Proses Penjadwalan Kunci

3.2. Menentukan Persamaan Polinomial Dari S-Box S-DES

Fungsi *s-box* diklaim sebagai fungsi non-linier yang bersifat acak sehingga dapat meningkatkan kompleksitas algoritma enkripsi. Langkah berikutnya adalah menentukan representasi aljabar dari masing-masing *s-box*. Pada S-DES ada dua *s-box* yang digunakan dinotasikan S0 dan S1 secara berurutan seperti terlihat di Gambar 5. *S-box* di S-DES memiliki 4-bit input dan 2-bit *output*, dengan ketentuan 2-bit terluar menunjukkan baris dan dua bit di dalam menunjukkan kolom dari *s-box*. Misalkan input 1101 pada *s-box* S0 akan menghasilkan *output* 3 yaitu S0(1101) = S0(3,2) = 3 = 11 (lihat yang ditandai pada *s-box* S0).

S0				
	0	1	2	3
0	1	0	3	2
1	3	2	1	0
2	0	2	1	3
3	3	1	3	2

S1				
	0	1	2	3
0	0	1	2	3
1	2	0	1	3
2	3	0	1	0
3	2	1	0	3

Gambar 5. Tabel *S-Box* S0 dan S1 pada S-DES

Selanjutnya dibuat tabel kebenaran dari setiap kemungkinan input dari *s-box* seperti terlihat pada Tabel 2. Misalkan input *s-box* dinotasikan sebagai $X = x_1, x_2, x_3, x_4$ dan $Y = y_1, y_2$ adalah *output* dari *s-box* S0. (Cat. Nilai Y pada tabel masih merupakan representasi dari 2-bit *output S-box*).

Tabel 2. Tabel Kebenaran S-Box S0

x_1	x_2	x_3	x_4	x_1x_2	…	$x_1x_2x_3$	…	$x_1x_2x_3x_4$	1	Y
0	0	0	0	0	…	0	…	0	1	1
0	0	0	1	0	…	0	…	0	1	3
0	0	1	0	0	…	0	…	0	1	0
0	0	1	1	0	…	0	…	0	1	2
0	1	0	0	0	…	0	…	0	1	3
0	1	0	1	0	…	0	…	0	1	1
0	1	1	0	0	…	0	…	0	1	2
0	1	1	1	0	…	0	…	0	1	0
1	0	0	0	0	…	0	…	0	1	0
1	0	0	1	0	…	0	…	0	1	3
1	0	1	0	0	…	0	…	0	1	2
1	0	1	1	0	…	0	…	0	1	1
1	1	0	0	1	…	0	…	0	1	1
1	1	0	1	1	…	0	…	0	1	3
1	1	1	0	1	…	1	…	0	1	3
1	1	1	1	1	…	1	…	1	1	2

Berdasarkan Tabel 2, selanjutnya ditentukan kombinasi input pada tabel kebenaran yang menghasilkan *output* yang sesuai dengan bit pertama dan bit kedua pada *s-box* S0, dan berlaku untuk

setiap kemungkinan input. Dari hasil analisis, untuk persamaan polinomial yang hasilnya konsisten sama dengan nilai bit pertama dari *output s-box* S0 dapat dilihat pada Tabel 3. Kolom bernilai 1 disediakan untuk memastikan nilai 1 untuk nilai input yang bernilai nol. Contoh pada baris pertama nilai *output* dari *s-box* adalah 1 atau 01, dengan bit pertama 0 dan bit kedua 1. Diketahui pada baris pertama semua nilai input adalah nol sehingga perlu adanya sebuah suku atau monomial konstan yang bernilai 1 yang akan ditambahkan pada persamaan polinomial untuk menjamin hasilnya adalah sama dengan 1 untuk bit kedua pada baris pertama.

Hal yang sama dilakukan pada *s-box* S1 untuk bit output pertama dan juga bit *output* yang kedua. Tabel 3 menunjukkan tabel kebenaran dari kombinasi monomial yang hasil XORnya sesuai dengan bit pertama dari *output s-box* S0 untuk setiap kemungkinan input. Pada Tabel 3 tidak dibutuhkan kolom bit 1 karena bit *output* pertama pada baris pertama *s-box* S0 bernilai nol. Sementara untuk tabel kebenaran bit *ouput* kedua membutuhkan kolom bit 1 karena pada baris pertama nilai bit kedua bernilai 1 pada *s-box* S1.

Tabel 3. Tabel Kebenaran Bit *Output* Pertama S0

Bit *output* pertama S0	x_2	x_4	x_1x_2	x_1x_3	$x_1x_2x_3x_4$	Hasil
0	0	0	0	0	0	**0**
0	0	0	0	0	0	**0**
1	1	0	0	0	0	**1**
1	1	0	0	0	0	**1**
1	0	1	0	0	0	**1**
1	0	1	0	0	0	**1**
0	1	1	0	0	0	**0**
0	1	1	0	0	0	**0**
0	0	0	0	0	0	**0**
1	0	0	0	1	0	**1**
0	1	0	1	0	0	**0**
1	1	0	1	1	0	**1**
1	0	1	0	0	0	**1**
0	0	1	0	1	0	**0**
0	1	1	1	0	0	**0**
1	1	1	1	1	1	**1**

Berdasarkan Tabel 3 terbukti kombinasi linier dari suku-suku $x_2 + x_4 + x_1x_2 + x_1x_3 + x_1x_2x_3x_4$ menghasilkan nilai yang sama dengan bit *output* pertama pada S0, sehingga persamaan (1) merupakan representasi aljabar dari bit *output* pertama S0 dinotasikan dengan S_0^1. Menggunakan proses yang sama selanjutnya ditentukan representasi aljabar dari setiap bit *output* lainnya pada *s-box* S0 dan S1 yang dinotasikan S_0^2, S_1^1, S_1^2. Dengan demikian diperoleh empat persamaan polinomial yang mewakili 2-bit *output* dari *s-box* 0 dan *s-box* 1 sebagai berikut:

$$S_0^1 = x_2 + x_4 + x_1x_2 + x_1x_3 + x_1x_2x_3x_4 \quad (1)$$
$$S_0^2 = x_1 + x_3 + x_1x_2 + x_1x_3 + \cdots + x_1x_2x_3x_4 + 1 \quad (2)$$
$$S_1^1 = x_1 + x_2 + x_4 + x_1x_3 + x_1x_4 + \cdots + x_1x_2x_3x_4 \quad (3)$$
$$S_1^2 = x_1 + x_3 + x_1x_4 + x_2x_4 + x_3x_4 + \cdots + x_1x_3x_4 \quad (4)$$

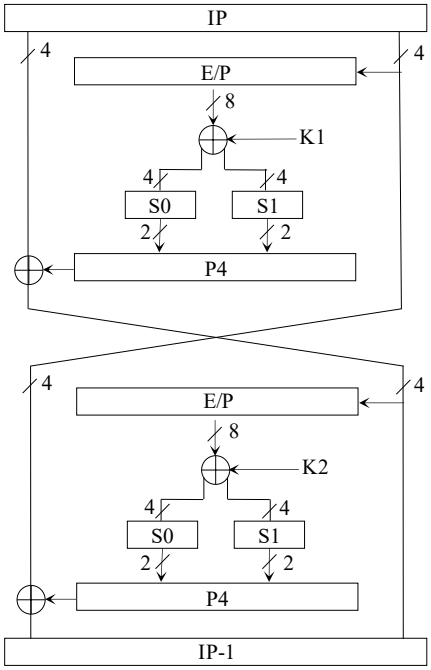
Keempat persamaan polinomial di atas akan digunakan untuk menentukan persamaan polinomial yang merupakan representasi dari hasil enkripsi algoritma S-DES (atau bit teks sandi) berdasarkan persamaan dari kunci putaran yang telah diperoleh sebelumnya.

3.3. Menentukan Persamaan Polinomial dari Teks Sandi

Ilustrasi serangan menggunakan pasangan teks terang dan teks sandi yang berkorespondensi $(P, C) = (01101101, 01000110)$ dengan asumsi kunci enkripsi tidak diketahui. Penyerang menggunakan pasangan teks terang dan teks sandi yang diketahui di atas untuk menentukan persamaan polinomial dari setiap bit teks sandi. Teks terang yang diketahui dienkripsi menggunakan kunci putaran yang diperoleh sebelumnya sesuai dengan skema enkripsi pada

 doi.org/10.56706/ik.v19i1.114

Gambar 6. Beberapa tabel permutasi yang digunakan pada proses enkripsi S-DES dapat dilihat pada Gambar 7.



Gambar 6. Skema Enkripsi S-DES

Sesuai skema enkripsi, teks terang P = 01101101 dipermutasi dengan fungsi IP kemudian dibagi dua masing-masing berukuran 4-bit. Setengah bagian bit kanan masuk ke fungsi f dan diproses menggunakan kunci putaran K_1 pada putaran ke-1 dan kunci putaran K_2 pada putaran ke-2. Input teks terang 01101101 dipermutasi dengan IP menghasilkan 11100110, kemudian dibagi menjadi dua bagian $(L_0, R_0) = (1110, 0110)$. Nilai R_0 masuk ke dalam fungsi f untuk diekspansi menjadi 00111100 menggunakan tabel E/P agar setara dengan ukuran kunci putaran K_1.

IP							
2	6	3	1	4	8	5	7

IP^{-1}							
4	1	3	5	7	2	8	6

E/P							
4	1	2	3	2	3	4	1

P4			
2	4	3	1

Gambar 7. Tabel Permutasi pada Enkripsi/Dekripsi S-DES

Selanjutnya hasil ekspansi di XOR dengan K_1 menghasilkan 8-bit *output* yang direpresentasikan oleh 8-digit karakter $k_1k_7(k_9+1)(k_4+1)\ (k_8+1)\ (k_3+1)k_{10}k_6$. Empat karakter pertama $k_1k_7(k_9+1)(k_4+1)$ akan masuk ke *s-box* S0 dan sisanya $(k_8+1)(k_3+1)k_{10}k_6$ masuk ke *s-box* S1.

Secara lengkap, proses enkripsi pada putaran ke-1 dapat diilustrasikan seperti terlihat pada Gambar 8.

Input	01101101	
IP	11100110	
Split	L_0 =1110	R_0 = 0110
E/P		00111100
XOR K_1		$k_1k_7(k_9+1)(k_4+1)\ (k_8+1)(k_3+1)k_{10}k_6$
S-Box		
		S0 S1
		$(S_0^1, S_0^2, S_1^1, S_1^2)$
P4		$(S_0^2, S_1^2, S_1^1, S_0^1)$
XOR L_0	$(S_0^2+1, S_1^2+1, S_1^1+1, S_0^1)$	
swap	$L_1 = 0110$	$R_1 = (S_0^2+1, S_1^2+1, S_1^1+1, S_0^1)$

Gambar 8. Proses Enkripsi Putaran Ke-1

Digit-digit karakter yang masuk ke dalam *s-box* merepresentasikan bit-bit input yang dinotasikan dengan string $x_1x_2x_3x_4$, sehingga untuk 4-digit yang masuk ke *s-box* S0 adalah $k_1k_7(k_9+1)(k_4+1)$ dengan:

$$x_1 = k_1 \quad (5)$$
$$x_2 = k_7 \quad (6)$$
$$x_3 = (k_9+1) \quad (7)$$
$$x_4 = (k_4+1) \quad (8)$$

Selanjutnya karakter input tersebut dimasukan ke dalam persamaan *s-box* (1) dan (2) pada *s-box* S0 sehingga diperoleh persamaan *output s-box* S0 sebagai berikut:

$$S_0^1 = k_7 + (k_4+1) + \cdots + k_1k_7(k_9+1)(k_4+1) \quad (9)$$
$$S_0^2 = k_1 + (k_9+1) + \cdots + k_1k_7(k_9+1)(k_4+1) + 1 \quad (10)$$

Persamaan (9) dan (10) merepresentasikan 2-bit *output* pada *s-box* S0. Menggunakan cara yang sama juga ditentukan dua persamaan bit *output* pada *s-box* S1 berdasarkan 4-karakter input $(k_8+1)(k_3+1)k_{10}k_6$ sebagai berikut:

$$S_1^1 = (k_8+1) + \cdots + (k_8+1)(k_3+1)k_{10}k_6 \quad (11)$$
$$S_1^2 = (k_8+1) + k_{10} + \cdots + (k_8+1)(k_3+1)k_{10}k_6 \quad (12)$$

Keempat persamaan (5,6,7,8) digabung yang merepresentasikan 4-bit *output s-box* dinotasikan $(S_0^1, S_0^2, S_1^1, S_1^2)$ dan selanjutnya dipermutasi dengan tabel P4 menghasilkan $(S_0^2, S_1^2, S_1^1, S_0^1)$.

Hasil permutasi P4 kemudian di-XOR dengan L_0 =1110 menghasilkan $(S_0^2+1, S_1^2+1, S_1^1+1, S_0^1)$ dan selanjutnya di-*swap* dengan $R_0 = 0110$ menghasilkan $L_1 = 0110$ dan $R_1 = (S_0^2+1, S_1^2+1, S_1^1+1, S_0^1)$. Kedua nilai tersebut menjadi input pada putaran ke-2 dengan R_1 input pada fungsi f yang akan diproses idengan proses yang sama. *Output* dari fungsi f kemudian disandingkan dengan R_1 dan dipermutasi dengan fungsi IP^{-1}. Pada putaran ke-2 tidak dilakukan fungsi *swap*. *Output* dari permutasi IP^{-1} menjadi teks sandi.

Seperti tersebut di atas, hasil putaran ke-1 diperoleh empat buah persamaan polinomial yang menjadi input pada putaran ke-2. Hasil dari putaran ke-2 akan menghasilkan delapan persamaan polinomial yang mewaikili 8-bit teks sandi sebagai berikut:

$$c_1 = k_3 + k_4 + \cdots + k_1k_3k_4k_5k_7k_8k_9k_{10} + 1 \quad (13)$$
$$c_2 = k_3 + k_8 + k_{10} + \cdots + k_1k_3k_4k_5k_7k_8k_9k_{10} + 1 \quad (14)$$
$$c_3 = k_1 + k_2 + k_3 + \cdots + k_1k_2k_3k_4k_6k_7k_8k_9k_{10} \quad (15)$$
$$c_4 = k_1 + k_9 + k_1k_4 + k_1k_7 + \cdots + k_1k_4k_7k_9 + 1 \quad (16)$$

$$c_5 = k_3 + k_8 + k_3k_{10} + k_6k_8 + \cdots + k_3k_6k_8k_{10} + 1 \quad (17)$$
$$c_6 = k_2 + k_3 + k_9 + \cdots + k_1k_3k_4k_6k_7k_8k_9k_{10} \quad (18)$$
$$c_7 = k_1 + k_4 + k_7 + k_1k_9 + \cdots + k_1k_4k_7k_9 + 1 \quad (19)$$
$$c_8 = k_6 + k_8 + k_{10} + k_8k_{10} + \cdots + k_3k_6k_8k_{10} \quad (20)$$

Ke-8 persamaan polinomial representasi teks sandi yang diperoleh disandingkan dengan 8-bit teks sandi 01000110 yang diketahui sehingga menghasilkan sistem persamaan sebagai berikut:

$$c_1 = k_3 + k_4 + \cdots + k_1k_3k_4k_5k_7k_8k_9k_{10} = 1 \quad (21)$$
$$c_2 = k_3 + k_8 + k_{10} + \cdots + k_1k_3k_4k_5k_7k_8k_9k_{10} = 0 \quad (22)$$
$$c_3 = k_1 + k_2 + k_3 + \cdots + k_1k_2k_3k_4k_6k_7k_8k_9k_{10} = 0 \quad (23)$$
$$c_4 = k_1 + k_9 + k_1k_4 + k_1k_7 + \cdots + k_1k_4k_7k_9 = 1 \quad (24)$$
$$c_5 = k_3 + k_8 + k_3k_{10} + k_6k_8 + \cdots + k_3k_6k_8k_{10} = 1 \quad (25)$$
$$c_6 = k_2 + k_3 + k_9 + \cdots + k_1k_3k_4k_6k_7k_8k_9k_{10} = 1 \quad (26)$$
$$c_7 = k_1 + k_4 + k_7 + k_1k_9 + \cdots + k_1k_4k_7k_9 = 0 \quad (27)$$
$$c_8 = k_6 + k_8 + k_{10} + k_8k_{10} + \cdots + k_3k_6k_8k_{10} = 0 \quad (28)$$

Selanjutnya ke-8 persamaan di atas dicari solusinya untuk memulihkan kunci rahasianya. Dalam proses pencarian solusi dapat digunakan beberapa pendekatan diantaranya eliminasi Gauss [10], algoritma *extended linearization* (XL) [9], atau basis Grobner [11]. Dalam implementasinya dilakukan linierisasi agar dimungkinan ditentukan solusinya. Pada penelitian ini digunakan pendekatan dengan algoritma XL. Hal ini dilakukan karena jumlah persamaan yang diperoleh hanya delapan sementara jumlah *unknown variable* sebanyak 344 sehingga perlu dilakukan perluasan jumlah persamaan yang dijelaskan pada bagian 4.

4. MENCARI SOLUSI SISTEM PERSAMAAN POLINOMIAL

4.1. Memperluas Sistem Persamaan Polinomial Dengan Algoritma XL

Perluasan sistem persamaan polinomial dilakukan untuk memenuhi jumlah persamaan polinomial yang ada sehingga dapat ditemukan solusi. Tahapan penerapan algoritma XL adalah sebagai berikut:

1) Membuat daftar semua monomial. Pada tahap ini semua monomial yang terkandung dalam delapan persamaan polinomial dari bit teks sandi diidentifikasi dan dibuat daftarnya. Daftar jumlah monomial dari setiap derajat dapat dilihat pada Tabel 4.
2) Menentukan jumlah *unknown variable* yang harus ditentukan dalam pencarian solusi persamaan. Dari delapan persamaan yang diperoleh, terdapat monomial sebanyak 405 dengan derajat tertinggi adalah sembilan.

Tabel 4. Daftar Jumlah Monomial

Derajat	Monomial	Jumlah
1	$k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8, k_9, k_{10}$	10
2	$k_1k_3, k_1k_4, k_1k_6, k_1k_7, k_1k_8, k_1k_9, \ldots, k_9k_{10}$	21
3	$k_1k_2k_7, k_1k_3k_7, k_1k_3k_8, k_1k_3k_9, \ldots, k_8k_9k_{10}$	65
4	$k_1k_2k_4k_7, k_1k_2k_4k_8, k_1k_2k_4k_9, \ldots, k_7k_8k_9k_{10}$	92
5	$k_1k_2k_3k_4k_6, k_1k_2k_3k_4k_{10}, \ldots, k_6k_7k_8k_9k_{10}$	95
6	$k_1k_2k_3k_4k_6k_7, \ldots, k_5k_6k_7k_8k_9k_{10}$	72
7	$k_1k_2k_3k_4k_6k_7k_8, \ldots, k_3k_5k_6k_7k_8k_9k_{10}$	37
8	$k_1k_2k_3k_4k_6k_7k_8k_9, \ldots, k_1k_4k_5k_6k_7k_8k_9k_{10}$	11
9	$k_1k_2k_3k_4k_6k_7k_8k_9k_{10}, k_1k_3k_4k_5k_6k_7k_8k_9k_{10}$	2
	Total	405

3) Menentukan jumlah *unknown variable* yang harus ditentukan dalam pencarian solusi persamaan. Dari 405 monomial yang terdapat pada ke-8 persamaan polinomial, terdatap 344 variabel yang tidak diketahui dengan derajat tertinggi adalah sembilan.

4) Menentukan target perluasan persamaan polinomial sesuai kebutuhan untuk mencari solusi persamaan. Diketahui jumlah persamaan hanya delapan, sementara jumlah variabel yang tidak diketahui ada 344. Oleh karena itu perlu dilakukan perluasan jumlah persamaan menggunakan algoritam XL. Pada kasus ini ditentukan $d = 3$, dengan kata lain setiap persamaan dikalikan dengan setiap monomial dengan derajat ≤ 3. Jumlah monomial yang berderajat ≤ 3 berjumlah 96 monomial. Dari hasil perluasan diperoleh 8 × 96 monomial menghasilkan 768 persamaan baru, Sehingga total persamaan yang akan digunakan adalah 768 + 8 = 776 persamaan. Contoh perluasan dengan monomial k_1 setelah dikalikan dengan ke-8 persamaan awal diperoleh persamaan baru:

$$c_1 = k_1 + k_1k_3 + \cdots + k_1k_3k_4k_5k_7k_8k_9k_{10} = 0 \quad (29)$$
$$c_2 = k_1k_3 + k_1k_8 + \cdots + k_1k_3k_4k_5k_7k_8k_9k_{10} = 0 \quad (30)$$
$$c_3 = k_1 + k_1k_2 + \cdots + k_1k_2k_3k_4k_6k_7k_8k_9k_{10} = 0 \quad (31)$$
$$c_4 = k_1k_4 + k_1k_7 + k_1k_9 + \cdots + k_1k_4k_7k_9 = 0 \quad (32)$$
$$c_5 = k_1 + k_1k_3 + k_1k_8 + \cdots + k_1k_3k_6k_8k_{10} = 0 \quad (33)$$
$$c_6 = k_1 + k_1k_2 + \cdots + k_1k_3k_4k_6k_7k_8k_9k_{10} = 0 \quad (34)$$
$$c_7 = k_1 + k_1k_4 + k_1k_7 + \cdots + k_1k_4k_7k_9 = 0 \quad (35)$$
$$c_8 = k_1k_6 + k_1k_8 + \cdots + k_1k_3k_6k_8k_{10} = 0 \quad (36)$$

5) Mentransformasi persamaan polinomial menjadi persamaan linier. Tahap ini dilakukan dengan cara mengubah semua monomial dengan derajat ≥ 1 menjadi monomial berderajat 1 sebagai variabel baru. Diketahui variabel kunci rahasia terdiri dari 10 karakter $k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8, k_9, k_{10}$, sehingga dibuat daftar semua kemungkinan monomial berderajat satu yang merepresentasikan semua kemungkinan variasi kombinasi ke-10 nilai kunci rahasia tersebut. Tabel 5 menunjukkan daftar perubahan setiap monomial berderajat *n* (untuk *n* = 1,2, …, 10) menjadi monomial berderajat satu sebagai variabel baru.

Tabel 5. Daftar Variabel Baru (Sebagian)

Sebelum	Sesudah
k_1	$M1$
k_2	$M2$
k_3	$M3$
⋮	⋮
k_{10}	M_{10}
k_1k_2	M_{11}
k_1k_3	M_{12}
⋮	⋮
$k_1k_2k_3k_4k_5k_6k_7k_8k_9$	M_{1014}
⋮	⋮
$k_1k_2k_3k_4k_6k_7k_8k_9k_{10}$	M_{1018}
⋮	⋮
$k_2k_3k_4k_5k_6k_7k_8k_9k_{10}$	M_{1022}
$k_1k_2k_3k_4k_5k_6k_7k_8k_9k_{10}$	M_{1023}

6) Berdasarkan Tabel 5, selanjutnya seluruh persamaan polinomial baru yang diperoleh dari hasil perluasan pada tahap 4), ditransformasi menjadi persamaan linier. Sebagai contoh persamaan awal c_3 pada persamaan (31) ditransformasi menjadi variabel baru c'_3 yang berderajat 1 sebagai berikut.

c_3	$k_1 + k_2 + k_3 + \cdots + k_1k_2k_3k_4k_6k_7k_8k_9k_{10} = 0$
c'_3	$M_1 + M_2 + M_3 + \cdots + M_{1018} = 0$

Gambar 9. Transformasi Persamaan Polinomial ke Persamaan Linier

Setelah seluruh persamaan ditransformasi menjadi persamaan linier maka diperoleh sistem persamaan linier untuk dicari solusinya.

4.2. Memecahkan Sistem Persamaan Linier

Setelah seluruh 776 persamaan polinomial telah ditransformasi menjadi persamaan linier, dilakukan pencarian solusinya menggunakan sistem eliminasi Gauss. Pencarian solusi dapat dilakukan dengan dua pendekatan. Pendekatan pertama melalui komputasi matriks menggunakan seluruh persamaan yang ada. Teknik ini membutuhkan sumber daya komputasi yang relatif lebih besar. Pendekatan kedua dilakukan bertahap dengan membentuk subset persamaan menjadi suatu sistem persamaan linier baru untuk memulihkan sebagian solusi. Berdasarkan hasil yang diperoleh dipilih lagi beberapa persamaan linier untuk menentukan nilai yang lain. Berikut adalah tahapan dalam menyelesaikan sistem persamaan linier dengan pendekatan kedua:

1) Dipilih beberapa persamaan untuk membentuk sistem persamaan linier baru. Sistem persamaan linier baru dibentuk dengan menganalisis persamaan mana yang dapat digunakan untuk menemukan nilai kunci. Tahap ini dilakukan untuk menyederhanakan proses pemecahan sistem persamaan dengan membentuk satu atau beberapa sistem persamaan baru dengan ukuran yang lebih kecil. Pada tahap ini dipilih 11 persamaan yang hanya mengandung variabel k_1, k_4, k_7, k_9 sebagai sistem persamaan I (terdiri dari persamaan (37) s.d. persamaan (47)), kemudian dikonversi ke dalam persamaan linier sesuai daftar variabel pada Tabel 5.

$$M1 + M9 + M13 + M16 + M18 + M87 + M235 = 1 \quad (37)$$
$$M13 + M16 + M87 + M235 = 0 \quad (38)$$
$$M2 + M11 + M26 + M57 + M60 + \cdots + M417 = 0 \quad (39)$$
$$M3 + M12 + M33 + M64 + M67 + \cdots + M452 = 0 \quad (40)$$
$$M4 + M39 + M73 + M75 = 0 \quad (41)$$
$$M5 + M14 + M44 + M71 + M78 + \cdots + M482 = 0 \quad (42)$$
$$M6 + M15 + M48 + M72 + M82 + \cdots + M488 = 0 \quad (43)$$
$$M7 + M51 + M73 + M235 = 0 \quad (44)$$
$$M8 + M17 + M53 + M74 + M86 + \cdots + M493 = 0 \quad (45)$$
$$M75 + M235 = 0 \quad (46)$$
$$M10 + M19 + M55 + M76 + M88 + \cdots + M495 = 0 \quad (47)$$

2) Dilakukan eliminasi Gauss pada sistem persamaan I yang telah ditentukan untuk menemukan solusi dengan mengubah setiap sistem persamaan linier menjadi matriks eselon baris. Disarankan untuk membentuk beberapa matriks eselon baris yang berisi variabel kunci yang berbeda. Tabel 6 menunjukkan contoh matriks eselon baris dari sistem persamaan I.

Tabel 6. Matriks Eselon Baris dari Sistem Persamaan 1

	$M1$	$M2$	$M3$	$M4$	$M5$	$M6$	⋯	$M495$	1
1	1	0	0	0	0	0	⋯	0	1
2	0	1	0	0	0	0	⋯	0	0
3	0	0	1	0	0	0	⋯	0	0
4	0	0	0	1	0	0	⋯	0	0
5	0	0	0	0	1	0	⋯	0	0
6	0	0	0	0	0	1	⋯	0	0
7	0	0	0	0	0	0	⋯	0	0
8	0	0	0	0	0	0	⋯	0	0
9	0	0	0	0	0	0	⋯	1	0
10	0	0	0	0	0	0	⋯	0	0
11	0	0	0	0	0	0	⋯	0	0

3) Dilakukan analisis matriks eselon baris. Analisis persamaan dilakukan dengan mengamati matriks eselon baris yang terbentuk. Misalkan, baris pertama dalam baris matriks eselon mewakili persamaan (48).

$M1 + M9 + M18 = 1$ (48)

Ubah monomial kembali ke bentuk aslinya.

$k_1 + k_9 + k_1k_9 = 1$ (49)

Memfaktorkan persamaan untuk mendapatkan solusi.

$(k_1 + 1)(k_9 + 1) = 0$ (50)

Dapat dilihat bahwa ada tiga kemungkinan solusi, yaitu:

- $k_1 = 0, k_9 = 1$ (51)
- $k_1 = 1, k_9 = 0$ (52)
- $k_1 = 1, k_9 = 1$ (53)

Selanjutnya analisis matriks eselon baris lainnya untuk menemukan nilai kunci lainnya. Misalkannya, baris ke-10 dalam baris matriks eselon mewakili persamaan (54).

$M13 + M16 + M87 + M235 = 0$ (54)

Ubah monomial kembali ke bentuk aslinya.

$k_1k_4 + k_1k_7 + k_1k_7k_9 + k_1k_4k_7k_9 = 0$ (55)

Substitusikan kemungkinan nilai kunci $k_1 = 0, k_9 = 1$ pada persamaan.

$0.k_4 + 0.k_7 + 0.k_7.1 + 0.k_4k_7.1 = 0$ (56)
$0 + 0 + 0 + 0 = 0$ (?) (57)

Substitusikan kemungkinan nilai kunci $k_1 = 1, k_9 = 0$ pada persamaan.

$1.k_4 + 1.k_7 + 1.k_7.0 + 1.k_4k_7.0 = 0$ (58)
$k_4 + k_7 = 0$ (59)
$k_4 = k_7$ (60)

Substitusikan kemungkinan nilai kunci $k_1 = 1, k_9 = 1$ pada persamaan.

$1.k_4 + 1.k_7 + 1.k_7.1 + 1.k_4k_7.1 = 0$ (61)
$k_4 + k_7 + k_7 + k_4k_7 = 0$ (62)
$k_4 + k_4k_7 = 0$ (63)
$k_4(1 + k_7) = 0$ (64)

Dari hasil substitusi di atas, diperoleh $\boldsymbol{k_1 = 1}$ sebagai nilai yang benar, selain itu didapat pula lima kemungkinan nilai kunci sebagai berikut:

- $k_4 = 0, k_7 = 0, k_9 = 0$ (65)
- $k_4 = 1, k_7 = 1, k_9 = 0$ (66)
- $k_4 = 0, k_7 = 0, k_9 = 1$ (67)
- $k_4 = 0, k_7 = 1, k_9 = 1$ (68)
- $k_4 = 1, k_7 = 1, k_9 = 1$ (69)

Selanjutnya kelima kemungkinan nilai kunci disubstitusi ke dalam persamaan bit *ciphertext* ke-7,

yang mengandung nilai k_1,k_4,k_7, dan k_9 untuk mencari nilai kunci yang benar.

$k_1 + k_4 + k_7 + k_1k_9 + \cdots + k_1k_4k_7k_9 = 0$ (70)

Substitusikan kemungkinan nilai kunci $k_1 = 1, k_4 = 0, k_7 = 0, k_9 = 0$ pada persamaan (70).

$1 + 0 + 0 + 1.0 + 1.0.0 + 1.0.0 + 1.0.0.0 = 0$ (71)
$1 + 0 + 0 + 0 + 0 + 0 + 0 = 0$ (×) (72)

Substitusikan kemungkinan nilai kunci $k_1 = 1, k_4 = 1,\ k_7 = 0, k_9 = 0$ pada persamaan (70).

$1 + 1 + 1 + 1.0 + 1.1.1 + 1.1.0 + 1.1.1.0 = 0$ (73)
$1 + 1 + 1 + 0 + 1 + 0 + 0 = 0$ (✓) (74)

Substitusikan kemungkinan nilai kunci $k_1 = 1, k_4 = 0,\ k_7 = 0, k_9 = 1$ pada persamaan (70).

$1 + 0 + 0 + 1.1 + 1.0.0 + 1.0.1 + 1.0.0.1 = 0$ (75)
$1 + 0 + 0 + 1 + 0 + 0 + 0 = 0$ (✓) (76)

Substitusikan kemungkinan nilai kunci $k_1 = 1, k_4 = 0,\ k_7 = 1, k_9 = 1$ pada persamaan (70).

$1 + 0 + 1 + 1.1 + 1.0.1 + 1.1.1 + 1.0.1.1 = 0$ (77)
$1 + 0 + 1 + 1 + 0 + 1 + 0 = 0$ (✓) (78)

Substitusikan kemungkinan nilai kunci $k_1 = 1, k_4 = 1,\ k_7 = 1, k_9 = 1$ pada persamaan (70).

$1 + 1 + 1 + 1.1 + 1.1.1 + 1.1.1 + 1.1.1.1 = 0$ (79)
$1 + 1 + 1 + 1 + 1 + 1 + 1 = 0$ (×) (80)

Didapat tiga kemungkinan solusi, yaitu:

- $k_4 = 1, k_7 = 0, k_9 = 0$ (81)
- $k_4 = 0, k_7 = 0, k_9 = 1$ (82)
- $k_4 = 0, k_7 = 1, k_9 = 1$ (83)

Pada tahap analisis menggunakan sistem persamaan I, didapat $\boldsymbol{k_1 = 1}$, sedangkan k_4, k_7, k_9 belum dapat ditentukan. Diperlukan sistem persamaan baru untuk mencari nilai $k_2, k_3, k_4k_5, k_6, k_7, k_8, k_9k_{10}$.

4) Mengulangi langkah 1-3 menggunakan sistem persamaan baru. Pada tahap ini dipilih 164 persamaan yang mengandung variabel $k_3, k_4, k_6, k_7, k_8, k_9, k_{10}$ sebagai sistem persamaan II sebagai berikut:

$M1 + M9 + M13 + M16 + M18 + M87 + M235 = 1$ (84)
$M13 + M16 + M87 + M235 = 0$ (85)
$M2 + M11 + M26 + M57 + M60 + \cdots + M417 = 0$ (86)
$M3 + M12 + M33 + M64 + M67 + \cdots + M452 = 0$ (87)
$M4 + M39 + M73 + M75 = 0$ (88)
$M5 + M14 + M44 + M71 + M78 + \cdots + M482 = 0$ (89)
$M6 + M15 + M48 + M72 + M82 + \cdots 1 + M488 = 0$ (90)
$M7 + M51 + M73 + M235 = 0$ (91)
$M8 + M17 + M53 + M74 + M86 + \cdots + M493 = 0$ (92)
$M75 + M235 = 0$ (93)

$$\vdots$$
$$M55 + M171 + M350 = 0 \quad (94)$$

Selanjutnya sistem persamaan II diubah menjadi matrik eselon baris seperti ditunjukkan pada Tabel 7. Setelah dilakukan analisis matriks eselon baris (seperti langkah 3) terhadap matriks pada Tabel 7, didapatkan nilai $\boldsymbol{k_3 = 1, k_6 = 0, k_8 = 0, k_{10} = 0}$.

Tabel 7. Matriks Eselon Baris dari Sistem Persamaan 2

No.	$M1$	$M2$	$M3$	$M4$	$M5$	$M6$	⋯	$M840$	1
1	1	0	0	0	0	0	⋯	0	0
2	0	1	0	0	0	0	⋯	0	0
3	0	0	1	0	0	0	⋯	0	1
4	0	0	0	1	0	0	⋯	0	0
5	0	0	0	0	1	0	⋯	0	0
6	0	0	0	0	0	1	⋯	0	0
7	0	0	0	0	0	0	⋯	0	0
8	0	0	0	0	0	0	⋯	0	0
9	0	0	0	0	0	0	⋯	1	1
10	0	0	0	0	0	0	⋯	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋱	⋮	⋮
164	0	0	0	0	0	0	⋯	0	0

5) Pada langkah 1-4, nilai kunci $k_1, k_3, k_4, k_6, k_8, k_9, k_{10}$ berhasil ditemukan, sedangkan nilai k_2, k_4, k_5, k_7, k_9 belum dapat ditemukan. Variabel k_2 dan k_5 berada di dalam persamaan yang cukup panjang sehingga tidak efektif jika dilakukan eliminasi Gauss. Oleh karena itu dilakukan substitusi pada persamaan yang mengandung variabel k_2 atau k_5 menggunakan 3 kemungkinan nilai kunci $k_1k_3k_4k_6k_7k_8k_9k_{10} = \{11100000, 11000010, 11001010\}$. Dari hasil substitusi, didapat $\boldsymbol{k_2 = 0}$ dan $\boldsymbol{k_5 = 0}$.

Setelah semua langkah dilakukan, kunci input rahasia $K = k_1k_2k_3k_4k_5k_6k_7k_8k_9k_{10}$ dapat direkonstruksi menjadi tiga kemungkinan nilai kunci sebagai berikut:

- $K = 1011000000$ (95)
- $K = 1010000010$ (96)
- $K = 1010001010$ (97)

Dari hasil rekonstruksi masih diperoleh tiga kemungkinan kunci, dengan nilai berbeda berada di posisi k_4, k_7, k_9. Langkah selanjutnya yang dapat dilakukan untuk menentukan mana kunci yang benar adalah dilakukan *brute force attack* dengan menggunakan ketiga kandidat kunci input yang diperoleh pada persamaan (95) sampai (97) berdasarkan teks sandi yang diperoleh. Dari hasil *brute force* dengan kedua kandidat kunci tersebut diperoleh teks terang seperti terlihat pada Tabel 8.

Tabel 8. Tabel Hasil Serangan *Brute Force*

Teks Sandi		
01000110		
No	Kandidat Kunci	Teks Terang
1	1011000000	11000101
2	1010000010	01101101
3	1010001010	01101101

Berdasarkan Tabel 8, terlihat bahwa terdapat dua kunci yang dapat digunakan untuk melakukan pemulihan teks terang dengan benar. Dengan menerapkan serangan aljabar ini maka serangan *brute force* untuk menentukan kunci yang benar hanya memerlukan tiga percobaan dari total *brute force* 2^{10} percobaan jika tanpa serangan aljabar.

5. KESIMPULAN

Berdasarkan penelitian yang telah dilakukan, dengan menggunakan satu pasang teks acak dan teks sandi yang berkorespondensi, dua kemungkinan kunci input rahasia berhasil diperoleh. Algoritma XL sebagai metode pencarian solusi sistem persamaan dapat digunakan untuk mencari solusi sistem persamaan polinomial yang diperoleh dari teks sandi pada S-DES secara efektif. Oleh karena itu, dapat dibuktikan bahwa S-DES rentan terhadap serangan aljabar. Melalui simulasi serangan ini dapat diperoleh gambaran utuh penerapan serangan aljabar pada S-DES sebagai praktek terbaik untuk pembelajaran penerapan serangan aljabar pada algoritma *block cipher* dengan desain sejenis.

REFERENSI

[1] M. Voros, *Algebraic Cryptanalysis on Stream Cipher*. Universitas Comminius, Departemen Ilmu Komputer. Bratislava.

[2] F. Paradise & K. Sugeng, *Algebraic Cryptanalysis on NTRU-HPS and NTRU-HRSS*. BAREKENG: J. Math. & App., vol. 17, no. 4, pp. 2187-2196, Dec, 2023.

[3] NT Courtois & GV Bard, *Algebraic Attack of the Data Encryption Standard*. Prosiding Konferensi Internasional IMA ke-11 tentang Kriptografi dan Pengkodean (IMACC'07). Catatan Kuliah dalam Ilmu Komputer, Vol. 4887. Springer, hlm. 152–169, 2008.

[4] Simmons, Sean. *Algebraic Cryptanalysis on Simplified AES*. Cryptologia 33, 305 – 314, 2009.

[5] E. Schaefer, *A Simplified Data Encryption Standard Algorithm*. Kriptologia 96, 1996.

[6] GV Bard, *Algebraic Cryptanalysis*. Amerika Serikat: Springer Science + Media Bisnis, hlm. 213–215, 2009.

[7] FIPS 46-3, *FIPS 46-3, Data Encryption Standard (DES)*. 2005.

[8] H. Anton, *Elementary Linear Algebra (Tenth Edition)*. John Wiley & Sons, Inc., hlm. 3–4, 2010.

[9] NT Curtois, dkk., *Efficient Algorithms for Solving Overdefined Systems of Multivariate Polinomial Equations*. Prosiding Konferensi Internasional tentang Teori dan Penerapan Teknik Kriptografi (EUROCRYPT'00). Catatan Kuliah dalam Ilmu Komputer, Vol. 1807. Springer, hlm. 392-407, 2000.

[10] Higham, Nicholas J., *Gaussian Elimination*. Wiley Interdisciplinary Reviews: Computational Statistics 3, 2011.

[11] G. Bourgeois & J. Faugère, *Algebraic Attack on NTRU Using Witt Vectors and Grobner Bases*. J. Math. Crypt. 3. 205-214. 10.1515/JMC.2009.011, 2009.