ADA Journal of Information System Research
Volume 1, No 2, February 2024 Page: 59−69
ISSN 3025-9568 (media online)
DOI: doi.org/10.64366/adajisr.v1i2.36

Implementasi Algoritma Sprizt Untuk Mengamankan Database Aplikasi Ujian Semester Berbasis Komputer

Magresi Manalu

Fakultas Ilmu Komputer dan Teknologi Informasi, Program Studi Teknik Informatika, Universitas Budi Darma, Medan, Indonesia
Email: magresimanalu@gmail.com

Abstrak−Maraknya sekolah melakukan ujian berbasis komputer, maka banyak aplikasi-aplikasi ujian berbasis *web* yang bermunculan dan dapat digunakan secara gratis, terutama pada SMK Alwashliyah Pasar Senen 2 Medan. Akan tetapi kelemahan pada ujian berbasis komputer adalah tidak adanya pengamanan soal yang disimpan kedalam *database*, tentu soal merupakan suatu data yang rahasia dan harus aman. Oleh sebab itu butuhkanya teknik pengamanan *database* soal untuk menjaga kerahasian soal baik terjadi serangan atau tidak. Teknik yang digunakan adalah teknik enkripsi kriptografi. Penelitian ini melakukan enkripsi *database* soal ujian berbasis komputer dengan algoritma *Spritz*. Dalam pengamanan *database* soal ujian dengan teknik kriptografi, soal yang berada pada *database* tidak dapat dimengerti artinya, sehingga apabila terjadi serangan, maka soal tidak dapat dibaca dan dimengerti oleh pihak penyerangan sehingga meminalisir tingkat kecurangan saat ujian. Tahapan proses enkripsi dilakukan saat admin menginput sebuah soal pada *interface web* ujian admin. Sedangkan tahap dekripsi dilakukan ketika siswa/i melaksanaankan ujian berbasis komputer. Berdasarkan proses enkripsi manual pada sampel *database* dengan *plaintext* "Penemu", didapati nilai *karakter* baru yang dinamakan dengan *ciphertext* yaitu rz¸ñjJ.

Kata Kunci: Ujian; Berbasis Komputer; Kriptografi; Spritz

Abstract−With the rise of schools conducting computer-based exams, many web-based exam applications have sprung up and can be used free of charge, especially at SMK Alwashliyah Pasar Senen 2 Medan. However, the weakness of the computer-based exam is that there is no security for questions that are stored in the database, of course the questions are confidential data and must be safe. Therefore we need a database security technique to keep the question secret, whether an attack occurs or not. The technique used is cryptographic encryption techniques. This study encrypts the database of computer-based exam questions with the Spritz algorithm. In securing the database of exam questions using cryptographic techniques, the meaning of the questions in the database cannot be understood, so that in the event of an attack, the questions cannot be read and understood by the attacking party, thereby eliminating the level of cheating during the exam. The encryption process is carried out when the admin inputs a question on the admin exam web interface. Meanwhile, the decryption stage is carried out when students carry out a computer-based exam. Based on the manual encryption process on the sample database with the plaintext "Inventor", a new character value is obtained which is called the ciphertext, namely rz¸ñjJ.

Keywords: Exams; Computer Based; Cryptography; Spritz

1. PENDAHULUAN

Berdasarkan perkembangan dunia digital saat ini, segala sesuatu pekerjaan pada bidang tertentu mulai dilakukan dengan bantuan komputer, tak terkecuali pada bidang pendidikan disekolah. Salah satu kegiatan disekolah yang digantikan oleh komputer adalah ujian sekolah berbasis komputerisasi. Beberapa sekolah yang berkembang saat ini menerapkan ujian akhir semester menggunakan komputer dan server sebagai media untuk meletakan soal serta mengerjakan soal ujian dengan bantuan aplikasi ujian yang telah tersedia. Salah satu sekolah tersebut adalah SMK Swasta Alwashliyah Pasar Senen 2 Medan.

Berbagai macam cara yang digunakan untuk melakukan ujian sekolah berbasis komputer diantaranya dapat dilaksanakan secara *online* dan dapat pula secara *offline* (*localhost*). Pada saat ini di SMK Swasta Alwashliyah Pasar Senen 2 Medan dilakukan ujian akhir semester berbasis *online* dengan mode ujian pilihan berganda. Aplikasi yang digunakan untuk menampilkan soal ujian berbasis *web* dan soal ujian disimpan ke dalam *database*. Ujian *online* berbasis *web* dimulai dengan peserta masuk/*login* ke aplikasi ujian *online* kemudian mengerjakan semua soal yang diujiankan dan pengumpulan jawaban ujian dilakukan dengan peserta mensubmit semua jawaban yang telah dipilih ke dalam *server*. Kemudian *server* akan memeriksa hasil ujian siswa dan mendapatkan *score* berupa hasil jawaban yang benar atau jumlah jawaban yang salah.

Kelemahan dari sistem ujian yang dilaksanakan pada SMK Swasta Alwashliyah Pasar Senen 2 Medan yang menggunakan aplikasi *website* adalah tidak adanya pengamanan enkripsi *record database* terutama pada *record* soal ujian dan jawaban. Hal ini merupakan sebuah celah yang dapat dimanfaat oleh pihak-pihak yang ingin membocorkan soal ujian serta jawaban. *Database* soal ujian merupakan salah satu aspek rahasia yang harus diamankan dengan proses enkripsi mengingat tidak ada sistem komputer yang sepenuhnya aman dari serangan orang yang tidak bertanggung jawab. Sehingga berdasarkan lemahnya tidak keamanan aplikasi ujian yang dimiliki oleh SMK Swasta Alwashliyah Pasar Senen 2 Medan maka dibutuhkan sebuah teknik keamanan kriptografi yang dapat meminalisir penyerangan *record database* yang dapat dibaca oleh manusia.

Kriptografi adalah sebuah teknik yang menjaga kerahasian data dan informasi agar tidak disalah gunakan oleh pihak yang tidak sah. Metode kriptografi melakukan pengacakan terhadap data yang diamankan sehingga data tersebut sulit dibaca dan tidak menutup kemungkinan tidak bisa dibaca sama sekali[1]–[5]. Salah satu algoritma kriptografi yang dapat digunakan untuk mengamankan *database* yaitu algoritma kriptografi *Spritz.* Algoritma *spritz*

ADA Journal of Information System Research
Volume 1, No 2, February 2024 Page: 59−69
ISSN 3025-9568 (media online)
DOI: doi.org/10.64366/adajisr.v1i2.36

merupakan varian dari algoritma RC4 yang dilakukan oleh Ron Rivest dan Jacob Schultz pada tahun 2014. Algoritma ini menghasilkan *sponge-base construction* dalam menghasilkan kunci dalam proses enkripsi dan dekripsi. Algoritma *Spritz* memeliki beberapa prosedur utama yang terdiri dari 3 proses yaitu *Key Scheduling Algorithm* (KSA), *Prseudo-Random Generation Algorithm* (PGRA) dan proses enkripsi dan dekripsi. Penambahan elemen w yang relatif prima dengan nilai N pada proses *Pseudo-Random Generation Algorithm* adalah perbedaan algoritma ini dengan RC4 sehingga algoritma Spritz memiliki tingkat enkripsi yang lebih baik dari RC4 dalam proses PGRA[6]–[10]. Hal tersebut menjadi alasan algoritma *Spritz* digunakan dalam pengamanan *database* ujian pada penelitian ini.

Penelitian sebelumnya telah dilakukan oleh Deni Anggara pada tahun 2020, dia menggunakan algoritma *Spritz* dalam mengamankan file suara dengan melakukan enkripsi, mereka melakukan proses pengubahan file suara menjadi format terenkripsi menghasilkan file baru yang disebut chiper. Durasi file suara chiper ini umumnya lebih pendek dibandingkan dengan file suara aslinya. Contoh kasus yang dibahas pada penelitian tersebur yaitu file suara listening dengan durasi awal 15.36 detik setelah dienkripsi menjadi 15.00 detik. Proses enkripsi ini membantu dalam mengamankan file suara, baik yang dibuat maupun disimpan dalam format wav dan mp3[11].

Raka Febrianto dan Sejati Waluyo juga melakukan penelitian pada tahun 2023. Mereka melakukan enkripsi terhadap database dengan menggunakan algoritma AES 256. Database yang diamankan mengenai data penilaian karyawan. Program Pengamanan Database Penilaian Pegawai Pada KJPP NDR menggunakan Algoritma AES 256 terbukti efektif dalam meningkatkan keamanan data. Pengembangan program ini secara berkelanjutan diharapkan dapat meningkatkan kemampuannya dalam melindungi data dan mencegah pencurian data[12].

Adiguna Ahlul Bai'at dan beberapa rekan penelitiannya melakukan sebuah penelitian pada tahun 2023, mereka melakukan pengemenan terhadap sebuah database transaksi. Penelitian ini menunjukkan bahwa metode kriptografi dengan algoritma Rivest Code 4 (RC4) efektif untuk mengamankan data transaksi pada GDS. Kunci yang dapat diacak berdasarkan panjangnya berperan penting dalam menjaga keamanan data asli (plainteks) dan data yang disandikan (ciphertext). Proses dekripsi dengan kunci yang sama dapat mengembalikan ciphertext menjadi plaintext seperti semula. Pengujian software CrackStation pada 6 ciphertext menunjukkan bahwa 100% ciphertext tidak dapat dipecahkan, membuktikan keamanan metode ini. Pengujian performa menunjukkan bahwa panjang kunci 5, 20, dan 90 karakter memengaruhi kecepatan enkripsi dan dekripsi, tetapi selisih waktunya sangat kecil. Hal ini menunjukkan bahwa perbedaan panjang kunci tidak signifikan terhadap performa keseluruhan fungsi enkripsi dan dekripsi[13].

2. METODOLOGI PENELITIAN

2.1 Kriptografi

Kriptografi bagaikan benteng digital yang melindungi informasi berharga dari pihak tak berwenang. Kata "kriptografi" berasal dari bahasa Yunani, "kryptos" yang berarti tersembunyi dan "graphia" yang berarti menulis. Inti kriptografi adalah mengubah informasi menjadi bentuk tersembunyi, aman dari pembacaan dan pemahaman pihak yang tidak memiliki akses. Di era digital ini, kriptografi menjadi alat vital untuk menjaga keamanan informasi. Seiring perkembangan teknologi, kriptografi terus berinovasi untuk menangkal berbagai ancaman keamanan baru. Penerapan Kriptografi dalam Berbagai Bidang diantaranya Keamanan Siber yaitu Melindungi data dan sistem komputer dari serangan siber, Keamanan Komunikasi yaitu Mengamankan komunikasi online, seperti email dan pesan instan. Otentikasi yaitu Memverifikasi identitas pengguna dan perangkat, Dan Perlindungan Hak Cipta seperti Melindungi konten digital dari pembajakan. Teknik-Teknik Kriptografi terdiri dari empat bagian yaitu Enkripsi, Mengubah informasi menjadi bentuk tersembunyi. Dekripsi yaitu Mengubah informasi terenkripsi kembali ke bentuk aslinya. Hashing yaitu Menghitung nilai unik dari informasi untuk memverifikasi integritasnya dan Tanda Tangan Digital untuk Memverifikasi keaslian informasi dan mencegah penyangkalan[14]–[20].

2.2 Algoritma Spritz

Algoritma *Spritz* adalah sebuah algoritma yang di perbaharui dari algoritma RC4 yang dilakukan oleh *Ron Rivest* dan *Jacob Schultz* pada 2014. Penamambahan elemen A yang relatif prima terhadap nilai N menjadi perbedaan antara algoritma *Spritz* dan RC4. Proses enkripsi algoritma *Spritz* adalah varian dari enkripsi algoritma RC4, dimana data atau pesan yang dienkripsi akan diproses dengan konsep *stream cipher* yaitu enkripsi satu persatu. Selain *stream cipher*, algoritma *Spritz* juga bias digunakan sebagai fungsi *hash* dan *Code Authentication Code* (MAC) dengan menggunakan *spons* berfungsi dalam mengamankan data[6]–[10], [21]. Algoritma *Spritz* memiliki prosedur utama yang terdiri dari tiga proses, yaitu :

1. *Key Scheduling Algorithm* (KSA)
   Key Scheduling Algorithm atau algoritma penjadwalan kunci digunakan untuk membuat tabel S-Box (*array* S) dan permutasi tabel dalam *array* S. Panjang *array* yang diperlukan adalah 256 yang mulai dari indeks 0 hingga 255. Tujuan dari KSA adalah memiliki proses *array* permutasi dengan nilai sebanyak 256 kali yang diinisialisasi dengan variabel i dan j dengan tipe integer. Rumus KSA adalah :
   untuk i = 0 sampai N – 1
   S[i] = i

ADA Journal of Information System Research
Volume 1, No 2, February 2024 Page: 59−69
ISSN 3025-9568 (media online)
DOI: doi.org/10.64366/adajisr.v1i2.36

selanjutnya i
i, j = 0
untuk i = 0 sampai N – 1
j = (j + S[i] + K[i mod Key.length]) mod N
tukar (S[i], S[j])
j = j
selanjutnya i
di mana N adalah ukuran array yang akan dimutasi, mis. 0 - 255.

2. *Pseudo-Random Generation Algoritma* (PRGA)
Pseudo-Random Generation Algoritma (PRGA) atau pengacakan dilakukan untuk mendapatkan kunci baru dengan sejumlah elemen - elemen polos. Nilai W adalah variabel baru yang ditambahkan ke algoritma *Spritz* sesuai dengan algoritma RC4. Nilai dari variabel i, j, k dan z dimulai dari 0 dan akan berubah sesuai hasil pada setiap iterasi. Proses ini melibatkan nilai-nilai *array* S yang telah diijinkan dalam proses KSA. Rumus PRGA adalah:
untuk i = 0 sampai *plain.length*
i = (i + w) mod N
j = (k + S [j + S [i]]) mod N
k = (i + k + S [j]) mod N
tukar S [i], S [j]
z = (S [j + S [i + S [z + k]]]) mod N
output z
selanjutnya i
di mana w adalah nilai integer yang relatif prima dengan N dan nilai i, j, k, z dimulai dari 0.
3. Enkripsi dan Dekripsi.
Proses enkripsi dan dekripsi dilakukan dengan cara XOR-biner setiap *output* z dengan masing-masing elemen polos dalam aliran.
Rumus enkripsi :

$$C_i = P_i \text{ XOR } z_i \quad (1)$$

Rumus dekripsi :

$$P_i = C_i \text{ XOR } z_i \quad (2)$$

Dimana :
P_i = elemen sederhana
C_i = elemen sandi
zi = elemen kunci (hasil proses PRGA)

3. HASIL DAN PEMBAHASAN

Algoritma yang digunakan dalam pembahasan ini adalah sebuah algoritma kriptografi simetri yaitu algoritma *Spritz*. Pengamanan *database* soal menggunakan algoritma *Spritz* dilakukan dengan 2 proses, yaitu proses enkripsi untuk mengamankan *database* dengan isi *record* soal didalam menu *adminweb* dan proses dekripsi *database* soal didalam menu *user* ketika memulai ujian.

Proses enkripsi untuk mendapatkan *ciphertext* algoritma *Spritz* pada tahap awal adalah pengacakan tabel *Key Scheduling Algorithm* (KSA) sebanyak 256 kali dengan nilai desimal kunci menggunakan rumus *Key Scheduling Algorithm* (KSA), dilanjutkan dengan tahap *Prseudo-Random Generation Algorithm* (PGRA) untuk mendapatkan nilai z akhir yang akan di XOR dengan nilai dari *plaintext* pada tahap enkripsi untuk mendapatkan *ciphertext*. Sedangkan proses dekripsi dilakukan dengan cara yang sama untuk mendapatkan *plaintext*. Operasi dari enkripsi dan dekripsi algoritma *Spritz* dilakukan dalam aplikasi berbasis *web*.

3.1 Penerapan Algoritma Spritz

Penerapan metode adalah proses hitungan manual untuk mengenkripsi suatu *database* soal menggunakan algoritma *Spritz*. *Database* tabel berisi soal ujian yang akan dienkripsi hanya berupa sampel untuk keperluan hitungan manual. Berikut diuraikan penerapan algoritma *Spritz* dalam mengenkripsi *database* tabel soal ujian yang berisi karakter huruf dan angka sebagai berikut:

Tabel 1. Sampel Tabel Soal Ujian

id_soal	Soal
1	Penemu..

ADA Journal of Information System Research
Volume 1, No 2, February 2024 Page: 59−69
ISSN 3025-9568 (media online)
DOI: doi.org/10.64366/adajisr.v1i2.36

Berdasarkan pada tabel 1 didapati beberapa karakter huruf dan angka yang akan di enkripsi menggunakan algoritma *Spritz*. Pada proses hitungan manual ini, untuk mempercepat proses hitungan diambil sampel karakter "Penemu" pada tabel database 4.1 di atas. Kunci yang digunakan untuk proses enkripsi adalah karakter "2020". Tahap awal proses enkripsi adalah pengacakan tabel *Key Scheduling Algorithm* (KSA) sebanyak 256 kali dengan nilai desimal kunci menggunakan rumus *Key Scheduling Algorithm* (KSA), dilanjutkan dengan tahap *Prseudo-Random Generation Algorithm* (PGRA) untuk mendapatkan nilai z akhir yang akan di XOR dengan nilai dari *plaintext* pada tahap enkripsi untuk mendapatkan *ciphertext.*

1. Tahap *Key Scheduling Algorithm (KSA)*
   Langkah *key scheduling* dimulai dengan menginisialisasikan *state* awal berupa larik yang terdiri dari 256 elemen. Larik *state* awal dapat dilihat pada tabel 2 berikut ini :

Tabel 2. Larik *State* Awal

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

Selanjutnya akan dilakukan perhitungan untuk nilai *j*. Untuk melakukan perhitungan nilai *j* maka kunci yang digunakan harus diubah ke dalam kode ASCII. Kode ASCII dapat dilihat pada bab sebelumnya. Kemudian tentukan nilai desimal kunci "2020" berdasarkan tabel ASCII. Nilai desimal kunci yaitu :

Tabel 3. Nilai Desimal Kunci

Index	0	1	2	3
Kunci	2	0	2	0
Desimal	50	48	50	48

Kemudian dilakukan perhitungan nilai *j* yang pertama dengan nilai awal $i = 0$ dan $j = 0$ sebagai berikut :

$j = (j + S[i] + key[i \bmod keylength]) \bmod 256$

$j = (0 + S[0] + key[0 \bmod 4] \bmod 256$

$j = (0 + 0 + 50) \bmod 256$

j = 50 mod 256

j = 50

Nilai $S[i] = S[0]$ *swap* dengan nilai $S[j] = S[50]$ maka,

Nilai $S[0] = 50$ dan nilai $S[50] = 0$

Selanjutnya perhitungan kembali untuk nilai *j* yang kedua dengan nilai $i = 1$ dan $j = 50$ sebagai berikut :

$j = (j + S[i] + key[i \bmod keylength]) \bmod 256$

$j = (50 + S[1] + key[1 \bmod 4] \bmod 256$

$j = (50 + 1 + 48) \bmod 256$

$j = 99 \bmod 256$

$j = 99$

Nilai $S[i] = S[1]$ *swap* dengan nilai $S[j] = S[99]$ maka,

Nilai $S[1] = 99$ dan nilai $S[99] = 1$

Selanjutnya perhitungan kembali untuk nilai *j* yang ketiga dengan nilai $i = 2$ dan $j = 99$ sebagai berikut :

$j = (j + S[i] + key[i \bmod keylength]) \bmod 256$

ADA Journal of Information System Research
Volume 1, No 2, February 2024 Page: 59−69
ISSN 3025-9568 (media online)
DOI: doi.org/10.64366/adajisr.v1i2.36

j = (99 + S[2] + *key*[2 mod 4] mod 256

j = (99 + 2 + 50) mod 256

j = 151 mod 256

j = 151

Nilai $S[i]$ = S[2] *swap* dengan nilai $S[j]$ = S[151] maka,

Nilai S[2] = 151 dan nilai S[151] = 2

Selanjutnya perhitungan kembali untuk nilai j yang keempat dengan nilai i = 3 dan j = 151 sebagai berikut :

j = (j + $S[i]$ + *key*[i mod *keylength*]) mod 256

j = (151 + S[3] + *key*[3 mod 4] mod 256

j = (151 + 3 + 48) mod 256

j = 202 mod 256

j = 202

Nilai $S[i]$ = S[3] *swap* dengan nilai $S[j]$ = S[202] maka,

Nilai S[3] = 202 dan nilai S[210] = 3

Langkah ini diulangi hingga i mencapai nilai 255 dan akan mengalami perubahan susunan nilai sesuai dengan nilai yang di-*swap* (ditukarkan) dan hasil dari tahap *key scheduling* dapat dilihat pada tabel 3 dimana nilai i berada pada baris yang berwarna biru.

Tabel 3. Hasil *Key Scheduling Algorithm(KSA)*

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

2. Tahap *Pseudo-Random Generation Algorithm(PRGA)*
Setelah melakukan tahap *Key Scheduling Algorithm(KSA)* maka akan dilakukan proses *Pseudo-Random Generation Algorithm(PRGA)* sebanyak panjang nilai *karakter plaintext*. Adapun panjang nilai *karakter* sampel *plaintext* sebagai berikut:

Tabel 4. Sampel Karakter *Database* yang dienkripsi

Plaintext	Sampel Karakter Database					
	P	e	n	e	m	u
Desimal	80	101	110	101	109	117
Biner	01010000	01100101	01101110	01100101	01101101	01110101

Pada tahap awal inisialisasikan nilai i = *0* , j = 0, k= 0, z = 0, dan w adalah bilangan acak yang merupakan bilangan GCD atau relatif prima dengan panjang S yaitu 256. Pada contoh ini bilangan relatif prima w telah ditetapkan yaitu= 221. Selanjutnya lakukan proses seperti berikut:

$i = i + w$ = 0+221 mod 256 = 221

$j = k + S[j + S[i]]$ = 0 + S[0 + S[221]] mod 256

ADA Journal of Information System Research
Volume 1, No 2, February 2024 Page: 59−69
ISSN 3025-9568 (media online)
DOI: doi.org/10.64366/adajisr.v1i2.36

= 0 + *S*[0 + 76] mod 256
= *S*[76] = 187

k = *i* + *k* + *S*[*j*] = 221 + 0 + *S*[187] mod 256

= 221 + 0 + 169 mod 256

= 134

S[*i*] , *S*[*j*] = *S*[*i*] = *S*[221] = 76, *S*[*j*] = *S*[187] = 169

Swap S[*i*] *with S*[*j*] = *S*[*i*] = *S*[221] = 169, *S*[*j*] = *S*[187] = 76

z = *S*[*j* + *S*[*i* + *S*[*z* + *k*]]] = S[187 + S[221 + S[0 + 134]]] mod 256

= S[187 + S[221 + 240]] mod 256

= S[187 + S[205] mod 256

= S[187 + 97] mod 256

= S[28] mod 256

z = 34

Tahap kedua inisialisasikan nilai *i* = 221, *j* = 187, k = 134, z = 34, dan w adalah bilangan acak yang merupakan bilangan GCD atau relatif prima dengan panjang S yaitu 256. Pada contoh ini bilangan relatif prima w telah ditetapkan yaitu= 221. Selanjutnya lakukan proses seperti berikut:

i = *i* + *w* = 221+221 mod 256 = 186

j = *k* + *S*[*j* + *S*[*i*]] = 134 + *S*[187 + *S*[186]] mod 256

= 134 + *S*[187 + 178] mod 256
= 134 + *S*[109] = 250

= 134 + 52

= 186

k = *i* + *k* + *S*[*j*] = 186 + 134 + *S*[186] mod 256

= 64 + 178 mod 256

= 242

S[*i*] , *S*[*j*] = *S*[*i*] = *S*[186] = 178, *S*[*j*] = *S*[186] = 178

SwapS[*i*] *withS*[*j*] = *S*[*i*] = *S*[186] = 178, *S*[*j*] = *S*[186] = 178

z = *S*[*j* + *S*[*i* + *S*[*z* + *k*]]] = S[186 + S[186 + S[34 + 242]]] mod 256

= S[186 + S[186 + S[20]]] mod 256

= S[186 + S[186 + 75] mod 256

= S[186 + S[5] mod 256

= S[186 + 195] mod 256

= S[125] mod 256

z = 31

Tahap ketiga inisialisasikan nilai *i* = 186, *j* = 186, k = 242 , z = 31, dan w adalah bilangan acak yang merupakan bilangan GCD atau relatif prima dengan panjang S yaitu 256. Pada contoh ini bilangan relatif prima w telah ditetapkan yaitu= 221. Selanjutnya lakukan proses seperti berikut:

i = *i* + *w* = 186+221 mod 256 = 151

j = *k* + *S*[*j* + *S*[*i*]] = 242 + *S*[186 + *S*[152]] mod 256

= 242 + *S*[186 + 89] mod 256
= 242 + *S*[19] mod 256

= 242 + 40 mod 256

= 26

ADA Journal of Information System Research
Volume 1, No 2, February 2024 Page: 59−69
ISSN 3025-9568 (media online)
DOI: doi.org/10.64366/adajisr.v1i2.36

$k = i + k + S[j]$ = 151 + 242 + S[26] mod 256

= 137 + 71 mod 256

= 208

$S[i]$, $S[j]$ = $S[i]$ = S[151] = 89, $S[j]$ = S[26] = 71

SwapS[i] *withS*[j] = $S[i]$ = S[151] = 71, $S[j]$ = S[26] = 89

$z = S[j + S[i + S[z + k]]]$ = S[26 + S[151 + S[31 + 208]]] mod 256

= S[26 + S[151 + S[239]]] mod 256

= S[26 + S[151 + 215] mod 256

= S[26 + S[110] mod 256

= S[26 + 17] mod 256

= S[43] mod 256

z = 214

Tahap keempat inisialisasikan nilai i = 151, j = 26, k = 208, z = 214, dan w adalah bilangan acak yang merupakan bilangan GCD atau relatif prima dengan panjang S yaitu 256. Pada contoh ini bilangan relatif prima w telah ditetapkan yaitu= 221. Selanjutnya lakukan proses seperti berikut:

$i = i + w$ = 151 + 221 mod 256 = 116

$j = k + S[j + S[i]]$ = 208 + S[26 + S[116]] mod 256

= 208+ S[26 + 129] mod 256
= 208 + S[155] mod 256

= 208 + 125 mod 256

= 107

$k = i + k + S[j]$ = 116 + 208 + S[207] mod 256

= 68 + 41 mod 256

= 109

$S[i]$, $S[j]$ = $S[i]$ = S[116] = 129, $S[j]$ = S[107] = 41

Swap $S[i]$ *with* $S[j]$ = $S[i]$ = S[116] = 41, $S[j]$ = S[107] = 129

$z = S[j + S[i + S[z + k]]]$ = S[107 + S[116 + S[214 + 109]]] mod 256

= S[107 + S[116 + S[67]]] mod 256

= S[107 + S[116 + 139] mod 256

= S[107 + S[255] mod 256

= S[107 + 46] mod 256

= S[153] mod 256

z = 148

Tahap ke lima inisialisasikan nilai i = 116, j = 107, k = 109, z = 148, dan w adalah bilangan acak yang merupakan bilangan GCD atau relatif prima dengan panjang S yaitu 256. Pada contoh ini bilangan relatif prima w telah ditetapkan yaitu= 221. Selanjutnya lakukan proses seperti berikut:

$i = i + w$ = 116 + 221 mod 256 = 81

$j = k + S[j + S[i]]$ = 109 + S[107 + S[81]] mod 256

= 109 + S[107 + 230] mod 256
= 109 + S[81] mod 256

= 109 + 81 mod 256

z = 190

$k = i + k + S[j]$ = 81 + 109 + S[190] mod 256

ADA Journal of Information System Research
Volume 1, No 2, February 2024 Page: 59−69
ISSN 3025-9568 (media online)
DOI: doi.org/10.64366/adajisr.v1i2.36

= 81 + 109 + 223 mod 256

= 157

$S[i]$, $S[j]$ = $S[i]$ = S[81] = 230, $S[j]$ = S[190] = 223

SwapS[*i*] *withS*[*j*] = $S[i]$ = S[81] = 223, $S[j]$ = S[190] = 230

$z = S[j + S[i + S[z + k]]]$ = S[190 + S[81 + S[148 + 157]]] mod 256

= S[190 + S[81 + S[49]]] mod 256

= S[190 + S[81 + 51] mod 256

= S[190 + S[132] mod 256

= S[190 + 6] mod 256

= S[196] mod 256

z= 7

Tahap ke enam inisialisasikan nilai i = 81, j = 190, k = 157, z = 7, dan w adalah bilangan acak yang merupakan bilangan GCD atau relatif prima dengan panjang S yaitu 256. Pada contoh ini bilangan relatif prima w telah ditetapkan yaitu= 221. Selanjutnya lakukan proses seperti berikut:

$i = i + w$ = 81 + 221 mod 256 = 46

$j = k + S[j + S[i]]$ = 157 + S[190 + S[46]] mod 256

= 157 + S[190 + 58] mod 256
= 109 + S[248] mod 256

= 109 + 248 mod 256

z = 101

$k = i + k + S[j]$ = 46 + 57 + S[101] mod 256

= 46 + 57 + 5 mod 256

= 118

$S[i]$, $S[j]$ = $S[i]$ = S[46] = 58, $S[j]$ = S[101] = 5

Swap $S[i]$ *with* $S[j]$ = $S[i]$ = S[46] = 5, $S[j]$ = S[101] = 58

$z = S[j + S[i + S[z + k]]]$ = S[101 + S[46 + S[7 + 118]]] mod 256

= S[101 + S[46 + S[125]]] mod 256

= S[101 + S[46 + 31] mod 256

= S[101 + S[77] mod 256

= S[101 + 121] mod 256

= S[222] mod 256

z= 63

4. Tahap Enkripsi
Pada tahap ini setiap nilai desimal dari *karakter* yang akan dienkripsi dirubah menjadi bilangan biner, lalu akan dilakukan operasi XOR dengan nilai *output z* dari perhitungan setiap tahap *Pseudo-Random Generation Algorithm (PRGA).* Nilai *output z* dari perhitungan setiap tahap *Pseudo-Random Generation Algorithm (PRGA)* akan dirubah menjadi bilangan biner. Proses enkripsi manual setiap nilai desimal *karakter*, yaitu :

Tabel 5. Sampel Karakter *Database* Desimal

Plaintext	Sampel Karakter Database					
	P	e	n	e	m	u
Desimal	80	101	110	101	109	117
Biner	01010000	01100101	01101110	01100101	01101101	01110101
Output z	34	31	214	148	7	63
Biner	00100010	00011111	11010110	10010100	00000111	00111111

a. Nilai Karakter “P”

ADA Journal of Information System Research
Volume 1, No 2, February 2024 Page: 59−69
ISSN 3025-9568 (media online)
DOI: doi.org/10.64366/adajisr.v1i2.36

Nilai karakter "P" pada sampel *database* bernilai desimal 80 akan di XOR dengan nilai *outup z* hasil perhitungan proses PRGA yang pertama yaitu 34.

80 = 01010000

34 = 00100010⊕

01110010 = 114

b. Nilai Karakter "e"
Nilai karakter "e" pada sampel *database* bernilai desimal 101 akan di XOR dengan nilai *output z* hasil perhitungan proses PRGA yang pertama yaitu 31.

101 = 01100101

31 = 00011111⊕

01111010 = 122

c. Nilai Karakter "n"
Nilai karakter "n" pada sampel *database* bernilai desimal 110 akan di XOR dengan nilai *output z* hasil perhitungan proses PRGA yang pertama yaitu 214.

110 = 01101110

214 = 11010110⊕

10111000 = 184

d. Nilai Karakter "e"
Nilai karakter "e" pada sampel *database* bernilai desimal 101 akan di XOR dengan nilai *outup z* hasil perhitungan proses PRGA yang pertama yaitu 148.

101 = 01100101

148 = 10010100⊕

11110001 = 241

e. Nilai Karakter "m"
Nilai karakter "m" pada sampel *database* bernilai desimal 109 akan di XOR dengan nilai *output z* hasil perhitungan proses PRGA yang pertama yaitu 7.

109 = 01101101

7 = 00000111 ⊕

01101010 = 106

f. Nilai Karakter "u"
Nilai karakter "u" pada sampel *database* bernilai desimal 117 akan di XOR dengan nilai *output z* hasil perhitungan proses PRGA yang pertama yaitu 63.

117 = 01110101

63 = 00111111 ⊕

01001010 = 74

Berdasarkan proses enkripsi manual pada sampel *database* dengan *plaintext* "Penemu", didapati nilai *karakter* baru yang dinamakan dengan *ciphertext* kemudian dirubah kembali kedalam karakter. Adapun hasilnya sebagai berikut:

Tabel 6. Nilai *Karakter Ciphertext*

	Nilai Ciphertext					
Desimal	114	122	184	241	106	74
Karakter	r	z	¸	ñ	j	J

Berdasarkan pada tabel *chipertext* di atas, adapun hasil dari bentuk ciphertext didalam database seperti pada tabel di bawah ini:

ADA Journal of Information System Research
Volume 1, No 2, February 2024 Page: 59−69
ISSN 3025-9568 (media online)
DOI: doi.org/10.64366/adajisr.v1i2.36

Tabel 7. *Ciphertext* Sampel Tabel Soal Ujian

id_soal	Soal
1	rz¸ñjJ..

Berdasarkan pada tabel 7 sampel database dienkripsi dan tidak dapat dipahami maknanya. Apabila terjadi kebocoran *database*, maka pihak attacker tidak akan mengetahui maksud dan makna dari soal tersebut, sehingga soal *database* tetap aman ketika terjadi penyerangan.

4. KESIMPULAN

Berdasarkan pengujian yang telah dilakukan sebelumnya, dapat disimpulkan bahwa adanya aplikasi sistem ujian berbasis komputer dengan proses pengamanan *database* soal ujian, dapat memberikan *level* keamanan *database* yang lebih baik jika terjadi serangan pihak yang tidak bertanggung jawab, hal ini disebabkan karena soal yang diinputkan *admin* akan dienkripsi terlebih dahulu kemudian disimpan kedalam *database*. Dengan proses enkripsi soal menggunakan algoritma *Spritz* berbasis *web ujian*, soal mengalami perubahan yang siginifikan. Soal tidak dapat dibaca dan dipahami maknanya sehingga meminilisi kebocoran soal. Program aplikasi ujian semester berbasis komputer dengan bahasa pemograman *web* dapat membantu pihak sekolah dalam proses ujian yang lebih efesien dari segi waktu dan tingkat kejujuran siswa.

REFERENCES

[1] N. W. Hidayatulloh, M. Tahir, H. Amalia, N. A. Basyar, A. F. Prianggara, and M. Yasin, "Mengenal Advance Encryption Standard (AES) sebagai Algoritma Kriptografi dalam Mengamankan Data," *Digit. Transform. Technol.*, vol. 3, no. 1, pp. 1–10, 2023.
[2] S. Oktaviani, F. Rizky, and I. Gunawan, "Analisis Keamanan Data Dengan Menggunakan Kriptografi Modern Algoritma Advance Encryption Standar (AES)," *J. Media Inform.*, vol. 4, no. 2, pp. 97–101, 2023.
[3] K. Andrea, A. Wardana, B. S. Wanandi, and A. Ikhwan, "Penerapan Kriptografi Caesar Cipher Pada Fitur Aplikasi Chatting Whatsapp," *J. Penelit. Dan Pengkaj. Ilm. Eksakta*, vol. 2, no. 1, pp. 6–11, 2023.
[4] M. Wasil, "Implementasi Matriks Dalam Kriptografi Hill Cipher Dalam Mengamankan Pesan Rahasia," *Zeta-Math J.*, vol. 8, no. 2, pp. 71–78, 2023.
[5] M. Hidayat, M. Tahir, A. Sukriyadi, and A. Sulton, "PENERAPAN KRIPTOGRAFI CAESAR CHIPER DALAM PENGAMANAN DATA," *J. Ilm. Multidisiplin*, vol. 2, no. 03, pp. 35–41, 2023.
[6] C. H. Harianja, "Analisis Algoritma Spritz dan Algoritma Goldbach G0 Codes pada Pengamanan Pesan Teks." Universitas Sumatera Utara, 2021.
[7] S. Kabeaken and I. Saputra, "Perancangan Aplikasi Enkripsi Pada File Teks Menggunakan Algoritma Spritz," *JURIKOM (Jurnal Ris. Komputer)*, vol. 7, no. 3, pp. 353–357, 2020.
[8] I. Z. H. Nasution, "Sistem Keamanan Citra Digital Menggunakan Algoritma Spritz." Universitas Islam Negeri Sumatera Utara, 2020.
[9] T. Harumy, "Implementasi Algoritma Knapsack dan Algoritma Spritz dalam Mengamankan File." Universitas Sumatera Utara, 2021.
[10] W. A. Prabowo, M. Mesran, and S. N. Hutagalung, "Perancangan Aplikasi Penyandian Pesan Chat Client dan Server Berdasarkan Algoritma Spritz," *JURIKOM (Jurnal Ris. Komputer)*, vol. 7, no. 3, pp. 358–364, 2020.
[11] D. Anggara, "Perancangan Aplikasi Enkripsi Pada File Suara Menggunakan Algoritma Spritz," vol. 1, no. 3, pp. 103–108, 2020.
[12] K. Ndr, "IMPLEMENTASI ALGORITME KRIPTOGRAFI ADVANCED ENCRYPTION STANDARD (AES-256) UNTUK MENGAMANKAN IMPLEMENTATION OF ADVANCED ENCRYPTION STANDARD CRYPTOGRAPHIC ALGORITHM (AES-256) TO SECURE EMPLOYEE ASSESSMENT DATABASE AT KJPP NDR," vol. 20, no. 1, pp. 44–49, 2023.
[13] L. Belakang, "Metode algoritma rc4 (rivest code 4) untuk pengamanan database transaksi pada glory digital sablon," vol. 13, no. 1, 2023.
[14] M. A. A. Widodo, M. Thasandra, S. O. Sutra, A. B. Nasution, and A. Ikhwan, "Pemanfaatan Kriptografi dalam Mewujudkan Keamanan Informasi pada E-Voting di Kota Medan dengan Menggunakan Algoritma AES," *J. Educ.*, vol. 5, no. 3, pp. 6780–6787, 2023.
[15] D. I. G. Hutasuhut, M. R. Aldizar, I. F. Nasution, and M. F. Nasution, "Perbandingan Algoritma Kriptografi Simetris dan Asimetris," *UNES J. Inf. Syst.*, vol. 8, no. 1, pp. 42–47, 2023.
[16] A. P. R. Tarigan, P. S. Ramadhan, and K. Ibnutama, "Penerapan Kriptografi Untuk Pengamanan Data Penjualan Sepatu Dengan Metode AES (Advanced Encryption Standard)," *J. Cyber Tech*, vol. 5, no. 1, pp. 26–35, 2023.
[17] M. S. D. Dairi and M. S. Asih, "Implementasi Algoritma Kriptografi RSA Dalam Aplikasi Sistem Informasi Perpustakaaan," *J. Ilmu Komput. dan Sist. Inf.*, vol. 2, no. 1, pp. 214–223, 2023.
[18] F. A. Sya'bani, "Hardening Sistem Informasi Sekawan V2 menggunakan Framework Owasp." Universitas Islam Indonesia, 2023.

ADA Journal of Information System Research
Volume 1, No 2, February 2024 Page: 59−69
ISSN 3025-9568 (media online)
DOI: doi.org/10.64366/adajisr.v1i2.36

[19] I. Mahgafhira, A. Wahid, and J. M. Parenreng, "Pengembangan Sistem Otentikasi Dokumen Digital Jurusan Teknik Informatika Dan Komputer Fakultas Teknik UNM Berbasis Digital Signature," *Progress. Information, Secur. Comput. Embed. Syst.*, vol. 1, no. 2, pp. 115–125, 2023.

[20] I. Rahim, N. Anwar, A. M. Widodo, K. K. Juman, and I. Setiawan, "Komparasi Fungsi Hash Md5 Dan Sha256 Dalam Keamanan Gambar Dan Teks," *IKRA-ITH Inform. J. Komput. dan Inform.*, vol. 7, no. 2, 2023.

[21] T. A. Irvida, "Implementasi Algoritma Spritz dan Algoritma Kunci Publik Rabin-p dengan Skema Hybrid Cryptosystem pada Pengamanan Teks." Universitas Sumatera Utara, 2020.