Journal of Information Sistem Research (JOSH)
Volume 1, No. 3, April 2020
ISSN 2686-228X (media online)
Hal: 103 - 108

Perancangan Aplikasi Enkripsi Pada File Suara Menggunakan Algoritma Spritz

Deni Anggara

Program Studi Teknik Informatika, STMIK Budi Darma, Medan, Indonesia
Email: denianggara@gmail.com

Abstrak− Masalah Keamanan dan kerahasiaan data merupakan salah satu aspek penting dalam hal pertukaran informasi. Salah satu solusi untuk menjaga keamanan dan kerahasiaan pada proses pengiriman file dengan teknik penyandian sebagai kunci enkripsi pada file suara. Penelitian ini memaparkan mengenai mengembangkan kriptografi (enkripsi dan dekripsi) yang menerapkan algoritma Spritz. Namun tidak semua perkembangan teknologi komunikasi memberikan dampak yang positif dan menguntungkan. Salah satu dampak negative dalam file suara adalah adanya penyadapan data, yang merupakan salah satu masalah yang paling ditakuti oleh para pengguna jaringan komunikasi maupun file suara. Penelitian ini menghasilkan sebuah program aplikasi bertujuan untuk pencegahan keamanan file suara pada media penyimpanan . dalam pembuatan aplikasi ini, metode yang digunakan adalah algoritma Spritz, karena sangat cocok untuk keamanan file suara yang menghasilkan penyelesaian proses enkripsi dan dekripsi. Aplikasi ini dibuat dengan perangkat lunak visual basic 2008 dan pembuatan aplikasi keamanan file dengan proses enkripsi dan dekripsi menggunakan algoritma algoritma Spritz, diharapkan untuk mengatasi permasalahan tersebut. Adapun format yang dilakukan dalam pembuatan aplikasi ini mengenkripsi dan mendekripsi file suara yang berekstensi. Wap, dan mp3.

Kata Kunci: Kriptografi, File Suara, Algoritma Spritz

Abstract−Issues Security and confidentiality of data is one important aspect in terms of information exchange. One solution to maintain security and confidentiality in the process of sending files with encryption techniques as encryption keys on voice files. This study describes developing cryptography (encryption and decryption) that applies the Spritz algorithm. But not all developments in communication technology have a positive and beneficial impact. One of the negative effects in sound files is data tapping, which is one of the most feared problems by network users and voice files. This research resulted in an application program aimed at preventing the security of sound files on storage media. in making this application, the method used is the Spritz algorithm, because it is very suitable for the security of voice files that result in the completion of the encryption and decryption process. This application was created with visual basic 2008 software and the creation of a file security application using encryption and decryption processes using the Spritz algorithm, is expected to overcome these problems. The format used in making this application encrypts and decrypts the sound files that have an extension. WAP, and MP3.

Keywords: Cryptography, Sound File, Spritz Algorithm

1. PENDAHULUAN

Kriptografi adalah salah satu teknik yang digunakan untuk meningkatkan aspek keamanan suatu informasi. Kriptografi mendukung kebutuhan dari dua aspek keamanan informasi, yaitu secrecy (perlindungan terhadap kerahasiaan data informasi) dan authenticity (perlindungan terhadap pemalsuan dan pengubahan informasi yang tidak diinginkan). Kriptografi merupakan metode untuk mengamankan data, baik itu data teks maupun file suara.

Keamanan dilakukannya penyandian file suara dapat mewakili banyak informasi yang bersifat rahasia atau penting, apalagi jika file suara tersebut dari departemen tertentu yang sangat dilindungi ataupun dijaga kerahasiaannya, di karenakan suara atau audio tersebut mempunyai nilai yang berharga contohnya seperti barang bukti suara record atau rekaman percakapan pada saat rapat. Lemahnya pengetahuan tentang keamanan data suara atau audio memudahkan para manipulative merekayasa suara seperti manipulation, editing, duplicating, sehingga perlu dilakukan pengamanan agar para manipulative sulit untuk memanipulasi data suara tersebut. Salah satu solusi yang dapat digunakan untuk menangani permasalahan tersebut adalah dengan teknik kriptografi maka dilakukan lah proses penyandian file suara tersebut ke dalam bentuk yang tidak dapat dimengerti yaitu berupa kriptografi, Salah satunya dengan menggunakan algoritma Spritz.

Algoritma Spritz merupakan varian dari algoritma RC4. Algoritma Spritz termasuk algoritma modern. Algoritma Spritz meningkatkan kompleksitas dari RC4, maka waktu yang digunakan untuk mengenkripsi sebuah data juga meningkat dan tingkat keamanan dari algoritma Spritz lebih baik dari RC4 [1]. Algoritma Spritz termasuk ke dalam kode aliran (stream cipher) dengan kunci rahasia/kunci simetri (kunci yang sama digunakan untuk proses enkripsi dan dekripsi). .

Penelitian yang telah dilakukan oleh Taronisoki Zebua dengan judul "Mengkodekan Database Rekaman Berbasis Komputer Ujian Tes Berdasarkan Algoritma Spritz " dan memberikan hasil bahwa Penelitian ini mendeskripsikan penggunaan algoritma spritz yang mana adalah salah satu algoritma kriptografi untuk menyandikan catatan basis data teks dari komputer berbasis uji. Hasil dari proses encoding dapat membuat lebih sulit bagi penyerang untuk mengetahui teks asli dari ujian, sehingga meminimalkan penyalahgunaan ujian. Algoritma ini bekerja berdasarkan pada konsep stream cipher yaitu enkripsi satu per satu. Salah satu kelebihan ini Algoritma adalah proses menghasilkan kunci yang digunakan dalam proses enkripsi dan dekripsi. Kunci yang dihasilkan berikutnya selalu bergantung pada alur dari kunci sebelumnya Itu kompleksitas tinggi dari kinerja

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**

Journal of Information Sistem Research (JOSH)
Volume 1, No. 3, April 2020
ISSN 2686-228X (media online)
Hal: 103 - 108

algoritma spritz menyebabkan kompleksitas cryptanalysts untuk menemukan kunci dan memecahkan algoritma ini[2].

Penelitian selanjutnya yang telah dilakukan oleh Ronald L. Rivest Mitssel dengan judul "Spritz sebuah stream chiper RC4 seperti spons dan fungsi hash" dan memberikan hasil bahwa bahwa Spritz dapat menghasilkan output dengan sekitar 24 siklus / byte perhitungan. Selanjutnya, uji statistik kami menunjukkan bahwa sekitar 281 byte output dibutuhkan sebelum seseorang dapat membedakan secara wajar Output spritz dari keluaran acak; ini adalah sebuah peningkatan yang ditandai atas RC4[3]

Dalam kriptografi terdapat beberapa metode yang cukup penting dalam penyandian pada file suara, Adapun yang diterapkan dalam penyandian pada file suara ini adalah menggunakan algoritma algoritma Spritz karena untuk menjaga kerahasian suatu data salah satunya adalah enkripsi (encryption). Enkripsi adalah suatu proses yang dilakukan untuk mengubah pesan asli menjadi ciphertext. Sedangkan suatu proses yang dilakukan untuk mengubah pesan tersembunyi menjadi pesan biasa (yang mudah dibaca) disebut dekripsi. Aplikasi penyandian pada file suara ditujukan untuk membantu mengatasi masalah keamanan file suara yang dibuat atau disimpan menggunakan file yang berformat wav, mp3 dari pencurian file baik yang tidak penting maupun yang penting dan rahasia. Sehingga orang lain tidak dapat mengetahui isi dari file tersebut.

2. METODOLOGI PENELITIAN

2.1 Kriptografi

Kata Cryptography berasal dari bahasa Yunani yang terdiri dari dua kata yaitu kryptos yang berarti rahasia dan graphein yang berarti tulisan (Mollin, 2007). Menurut terminologinya, kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan ketika pesan dikirim dari suatu tempat ke tempat lain[4].

2.2. Algoritma Spritz

Algoritma Spritz merupakan varian dari algoritma RC4 didesain oleh Ron Rivest yang berasal dari RSA Security pada tahun 1987. Spritz sendiri mempunyai singkatan resmi yaitu "Rivest Chiper", namun juga dikenal sebagai "Ron's Code". RC4 sebenarnya dirahasiakan dan tidak dipublikasikan kepada khalayak ramai, namun ternyata ada orang yang tidak dikenal menyebarkan RC4 ke mailing list Cypherpunks. Kemudian berita ini dengan cepat diposkan ke sci.crypt newsgroup, dan dari newsgroup ini kemudian menyebar luas di internet. Kode yang dibocorkan tersebut dipastikan keasliannya karena output yang dikeluarkan sama dengan software-software yang menggunakan RC4 yang berlisensi[5]. Proses Algoritma Spritz terdiri atas 2 bagian yaitu Key Scheduling algorithm (KSA) dan Pseudo Random Generation Algorithm (PRGA).

2.3 Key Schedulling Algorithm

Tahap pertama dari algoritma *RC4* adalah *key schedulling* (KSA). Pada tahap ini *State* diberi nilai awal berupa larik yang merepresentasikan suatu permutasi dengan 256 elemen. Jadi, hasil dari KSA adalah permutasi awal.[5]. Larik yang mempunyai 256 elemen ini (dengan indeks 0-255) dinamakan *S* (Kromodimoeljo, 2010). Langkahlangkah pembentukan *S* adalah sebagai berikut :

1. Inisialisasi kotak-*S* sesuai dengan indeksnya: *S*[0] = 0, *S*[1] = 1, …, *S*[255] = 255.
2. Kunci dimasukkan ke dalam larik berukuran 256. Jika ukuran kunci kurang dari 256 *byte*, maka dilakukan *padding* dengan mengulang kunci hingga larik terisi penuh. Misalkan kunci "jaya" dengan kode ASCII 106, 97, 121, dan 97.
3. Kotak-*S* disusun kembali membentuk kotak-*S* akhir. Langkah pertama, penghitung *j* diinisialisasi dengan nol. Kemudian nilai *j* yang baru = (*j* + *S*[*i*] + K[*i*]) mod 256. Lalu, nilai *S*[*i*] dan *S*[*j*] ditukarkan. Proses penentuan nilai *j* yang baru dan penukaran nilai *S*[*i*] dengan *S*[*j*] tersebut dilakukan dari i = 0 sampai dengan i = 255. Untuk *i* = 0, *j* = (0 + *S*[0] + *K*[0]) mod 256 = (0 + 0 + 106) mod 256 = 106. Nilai *S*[0] dan *S*[106] ditukarkan sehingga *S*[0] = 106 dan *S*[106] = 0. Demikian seterusnya sampai i = 255.

2.4 Pseudo Random Generation Algorithm

Tahap selanjutnya dari algoritma *RC4* dinamakan *Pseudo Random Generation Algorithm* (PRGA). Tahap ini menghasilkan nilai *pseudo-random* yang kemudian di- *XOR*-kan dengan *plaintext* untuk proses enkripsi atau dengan *ciphertext* pada proses dekripsi). Langkah pertama adalah menginisialisasikan nilai *i* dan *j* dengan nol. Untuk *k* = 0, …, *k* = panjang pesan – 1, nilai *i* dan *j* yang baru dihitung dengan cara :

i = (*i* + 1) mod 256
j = (*j* + S[*i*]) mod 256

Nilai *S*[*i*] dan *S*[*j*] ditukarkan. Kemudian, nilai *S* dengan indeks *t* = (*S*[*i*] + *S*[*j*]) mod256 diambil. Nilai *S*[*t*] tersebut terakhir di-XOR-kan dengan *plaintext* atau *ciphertext* dengan indeks *k*.

2.5 Suara (Audio)

Audio (Suara) adalah fenomena fisik yang dihasilkan oleh getaran suatu benda yang berupa sinyal analog dengan amplitude yang berubah secara kontiniu terhadap satuan waktu yang disebut frekuensi.Sedangkan suara yaitu suara

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**

Journal of Information Sistem Research (JOSH)
Volume 1, No. 3, April 2020
ISSN 2686-228X (media online)
Hal: 103 - 108

getaran yang dihasilkan oleh gesekan, pantulan dan lain-lain, antara benda-benda. Media audio merupakan suatu alat media yang isi pesannya hanya dapat diterima melalui indera pendengaran saja. media untuk menyampaikan pesan yang akan disampaikan dalam bentuk lambang-lambang auditif, baik verbal (kedalam kata-kata untuk bahasa lisan) maupun non verbal. [6]

Terdapat berbagai macam audio yang dikelompakan berdasarkan media ataupun perangkat yang sering digunakan, diantaranya :

1. Audio Streaming adalah suatu istilah yang dipakai untuk mendengarkan siaran langsung atau live melalui jaringan internet. Seperti contohnya : Winamp (MP3, Real audio (RAM) dan juga liquid radio).
2. Pengertian audio visual adalah suatu istilah yang digunakan untuk seperangkat soundsystem yang dilengkapi dengan tampilan gambar, biasanya dipakai presentasi.
3. Audio modem Riser (AMR) adalah suatu istilah yang dipakai untuk sebuah kartu plug-in untuk motherboard intel yang memuat sirkuit audio ataupun modem.

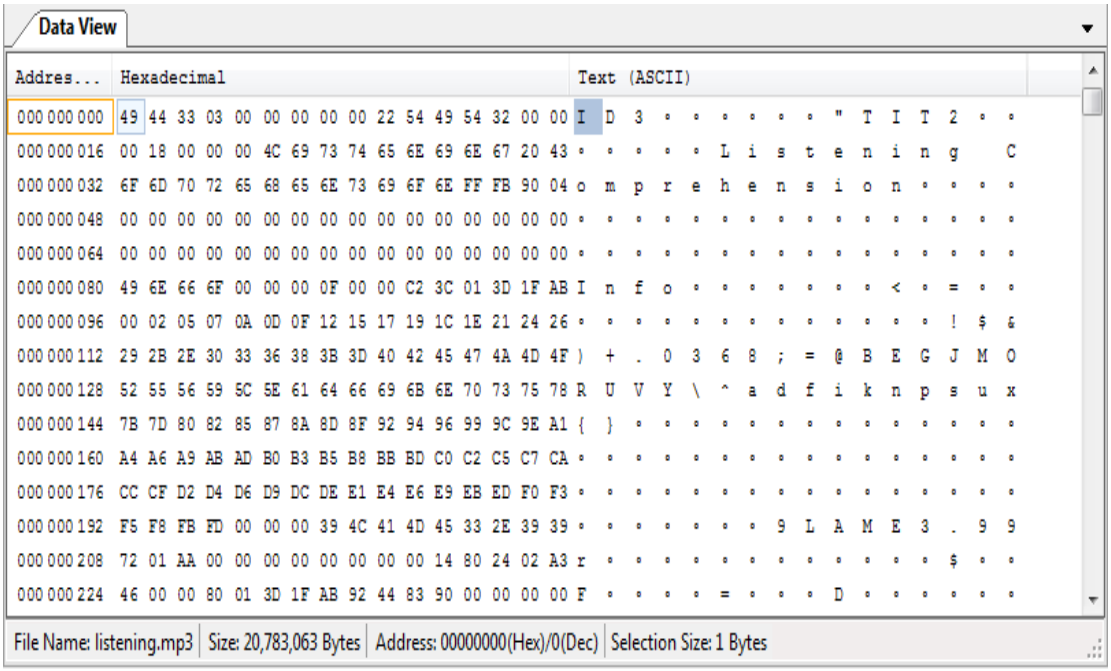
3. HASIL DAN PEMBAHASAN

Analisa adalah penguraian dari suatu pembahasan, dalam hal ini pembahasan mengenai membuat aplikasi untuk penerapan metode *Spritz* dalam mengamankan file suara yang akan dibuat. Analisa file suara merupakan tahapan dimana dilakukannya análisa terhadap file-file apa saja yang diolah dalam sistem atau prosedur sebuah rancangan, dalam hal ini file suara yang akan dienkripsi dan dekripsi pada aplikasi kriptografi adalah berupa *file* berformat (*.mp3).

Solusi dalam mengamankan file suara ini adalah menggunakan teknik kriptografi, Adapun untuk penerapan teknik kriptografi ini menggunakan kunci metode sekaligus diantaranya metode *Spritz*. Karena proses file dari kunci yang dibuat berdasarkan kode ASCII dibutuhkan input plain atau penyandian menggunakan (*stream cipher*) dengan kunci rahasia/kunci simetri (kunci yang sama digunakan untuk proses enkripsi dan dekripsi).

Alasan karena tekni kriptografi *Spritz* merupakan pembangkitan kunci aliran (*keystream*) yang bersifat acak semu (*pseudo random*). Algoritma *Spritz* terdiri atas 2 bagian yaitu *Key Scheduling algorithm* (KSA) dan *Pseudo Random Generation Algorithm* (PRGA) dengan salah satu teknik yang digunakan untuk meningkatkan aspek keamanan suatu informasi. Berdasarkan teknik penyandian *Spritz* dalam mengamankan file suara, diantaranya untuk menjaga suatu file agar file tersebut aman. Hasil proses enkripsi file suara akan menjadi sebuah cipher, sehingga file tersebut tidak dapat terbaca.

File suara seperti data suara listening ujian nasional atau rekaman percakapan pada saat rapat. Dimisalkan File suara listening yang akan di enkripsi dengan format mp3 dengan ukuran file 19,8 MB dengan durasi 21:38 Menit dan setelah dilakukan penyandian pada proses enkripsi file suara tersebut menjadi sebuah chiper dengan ukuran 15,8 MB dengan durasi 15:20 Menit. Pada proses Algoritma *Spritz* data yang akan dienkripsi adalah file suara listening yang berupa bilangan hexa decimal berdasarkan kode ASCII yang ditampilkan menggunakan bantuan binary_viewer. Contoh file suara listening yaitu dengan ukuran kapasitas file sebesar 19,8 MB (20.783.063 bytes).



Gambar 1. Nilai File audio listening.Mp3

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**

Journal of Information Sistem Research (JOSH)
Volume 1, No. 3, April 2020
ISSN 2686-228X (media online)
Hal: 103 - 108

3.1 Penerapan Algoritma *Spritz*

Tahapan Enkripsi file suara dengan Algotima *spritz* sebelum melakukan Enkripsi dimulai *key scheduling* dengan menginisialisasikan *state* awal berupa larik yang terdiri dari 256 elemen file suara, jadi hasil dari KSA adalah permutasi awal setelah itu Kunci dimasukan kedalam larik berukuran 256 jika ukuran kunci file suara kurang dari 256 *byte* maka dilakukan padding dengan mengulang kunci hingga larik terisi penuh kemudian Kunci file suara diubah ke dalam kode ASCII. selanjutnya *Pseudo-Random Generation Algorithm* untuk Enkripsi Setelah melakukan tahap *key scheduling* maka akan dilakukan proses enkripsi file suara. Tahap ini menghasilkan nilai *pseudo-random* yang kemudian di- *XOR*-kan dengan *plain* (file suara) untuk proses enkripsi atau dengan *cipher* file suara pada proses enkripsi.
Proses Enkripsi :

Dalam penelitian ini menggunakan sampel dari nilai yang ditampilkan nilai dari binary_viewer dengan mengambil sebagian sampel. Adapun sebelum di enkripsi mempunya nilai Plain chiper sebagai berikut:
Selanjutnya akan dilakukan perhitungan kunci algoritma *spritz* untuk nilai *j*. Untuk melakukan perhitungan nilai *j* maka kunci yang digunakan harus diubah ke dalam kode ASCII.
Plain ASCII untuk “iste” yaitu ::
i = 105
s = 115
t = 116
e = 101
Nilai ASCII untuk kunci “itu” yaitu :
i = 105
t = 116
u = 117

Tahap kunci algoritma spritz *Key Scheduling Algorithm (KSA).* Langkah *key scheduling* dimulai dengan menginisialisasikan *state* awal berupa larik yang terdiri dari 256 elemen. larik *state awal* berbentuk seperti dibawah ini:

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										

Gambar 2. Tabel Larik *S* awal

Kemudian dilakukan perhitungan nilai *j* yang pertama dengan nilai awal $i = 0$ dan $j = 0$ sebagai berikut :
$j = (j + S[i] + key[i \bmod keylength]) \bmod 256$
$j = (0 + 0 + 105) \bmod 256 = 105$

Tukarkan nilai *S*[0] dengan *S*[105]. Kemudian dilakukan perhitungan kembali untuk nilai *j* yang kedua dengan nilai $i = 1$ dan $j = 105$ sebagai berikut :
$j = (j + S[i] + key[i \bmod keylength]) \bmod 256$
$j = (105 + 1 + 116) \bmod 256 = 222$

Tukarkan nilai *S*[1] dengan *S*[222]. Langkah ini diulangi hingga *i* mencapai nilai 255 dan hasil dari tahap *key scheduling* dapat dilihat pada tabel 3.2 dimana nilai *i* berada pada baris yang berwarna biru.

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**

Journal of Information Sistem Research (JOSH)
Volume 1, No. 3, April 2020
ISSN 2686-228X (media online)
Hal: 103 - 108

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

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

Tahap *Pseudo-Random Generation Algorithm (PRGA)* untuk Enkripsi Setelah melakukan tahap *key scheduling* maka akan dilakukan proses enkripsi pesan "iste" dengan proses per karakter. 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. Selanjutnya lakukan proses seperti berikut:

Karakter "i"

Tahap awal dilakukan inisialisasi $i = 0$, $j = 0$, k= 0 dan z = 0 dan setiap baris di modulo dengan 256. Kemudian dilakukan perhitungan dengan cara sebagai berikut :

$i = i + w = 0+221 \bmod 256 = 221$

$j = k + S[j + S[i]] = 0 + S[0 + S[221]] \bmod 256$
$= 0 + S[0 + 230] \bmod 256$
$= S[\ 230] = 51$

$K = i + k + S[j] = 221 + 0 + S[51] \bmod 256$
$= 221 + 162 \bmod 256$
$= 127$

$S[i]$, $S[j] = S[i] = S[221] = 230$, $S[j] = S[\ 51] = 162$

$SwapS[i]\ withS[j] = S[i] = S[221] = 162$, $S[j] = S[51] = 230$

$z = S[j + S[i + S[z + k]]] = S[51 + S[221 + S[0 + 127]]] \bmod 256$
$= S[51 + S[221 + 226]] \bmod 256$
$= S[51 + S[191] \bmod 256$
$= S[51 + 211] \bmod 256$
$= S[6] = 50$

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**

Journal of Information Sistem Research (JOSH)
Volume 1, No. 3, April 2020
ISSN 2686-228X (media online)
Hal: 103 - 108

95 = 01011111
50 = 00110010 ⊕
= 01101101 = 109_{10}. dalam tabel ASCII merupakan karakter "**m** ".

Lakukan proses yang sama untuk karakter s,t dan e maka menghasilkan file yang tersandi atau terenkripsi menjadi sebuah cipher yaitu **m Ĩ ă Ɗ** .

Dekripsi Plain cipher file suara listening yang akan didekripsi dimana akan dilakukan operasi XOR dengan cara sebagai berikut :

i = *i* + *w* = 0+221 mod 256 = 221
j = *k* + *S*[*j* + *S*[*i*]] = 0 + *S*[0 + *S*[221]] mod 256
= 0 + *S*[0 + 230] mod 256
= *S*[230] = 51
k = *i* + *k* + *S*[*j*] = 221 + 0 + *S*[51] mod 256
= 221 + 162 mod 256
= 127
S[*i*] , *S*[*j*] = *S*[*i*] = *S*[221] = 230, *S*[*j*] = *S*[51] = 162
SwapS[*i*] *withS*[*j*] = *S*[*i*] = *S*[221] = 162, *S*[*j*] = *S*[51] = 230
z = *S*[*j* + *S*[*i* + *S*[*z* + *k*]]] = S[51 + S[221 + S[0 + 127]]] mod 256
= S[51 + S[221 + 226]] mod 256
= S[51 + S[191] mod 256
= S[51 + 211] mod 256
= S[6] = 50
127 = 01011011
50 = 00110010 ⊕
= 01101001 = 105_{10} dalam tabel ASCII merupakan karakter " **i** ".

Lakukan proses yang sama untuk karakter **Ĩ ă Ɗ**

4. KESIMPULAN

Berdasarkan hasil studi literatur, analisis, perancangan, implementasi, dan pengujian sistem ini, maka kesimpulan yang didapat adalah sebagai berikut :

1. Proses enkripsi pada file suara menghasilkan chiper dan dimana spesifikasi durasi file suara menjadi lebih berkurang seperti file suara listening durasi awal 15.36 stelah di enkripsi menjadi 15.00.
2. Pada penyandian file suara dengan menggunakan kode Algoritma spritz akan lebih optimal jika spritz karakter dari informasi tersebut tidak banyak walaupun frekuensi kemunculannya tinggi. Algoritma Spritz termasuk ke dalam kode aliran (stream cipher) dengan kunci rahasia/kunci simetri (kunci yang sama digunakan untuk proses enkripsi dan dekripsi).
3. Dengan melakukan enkripsi file suara dapat membantu dalam mengamankan file suara yang dibuat atau disimpan menggunakan file yang berformat wav, mp3.

REFERENCES

[1] Taronisokhi Zebua, "Encoding the record database of computer based test exam based on spritz algorithm," Lontar komputer, vol. 9, no. 1 , pp. 52–62, 2018.
[2] Ronald L. Rivest, 2014. "Spritz a spongy RC4-like stream cipher and hash function". Schuldt Research Institute for Secure Systems AIST, Japan.
[3] Sri Widianti, "Pengantar Basis Data",Yogyakarta, Informatika, 2000.
[4] Arius, D. 2008, "Awal Sejarah Kriptografi Didunia", STMIK AMIKOM, Yogyakarta.
[5] Munir, R., 2006, "Kriptografi", Informatika, Bandung.
[6] Nurasyiah, "Perancangan Aplikasi Kompresi File audio" ,Jakarta, Andi, 2013.
[7] Sadikin,R. 2012. "Kriptografi Untuk Keamanan Jaringan dan Implementasinya Dalam Bahasa Java". C.V. ANDI OFFISET, Yogyakarta.
[8] Ariyus, Dony. & Andri Rum, 2008. "KODE ASCII 7 BIT". Jurusan Teknik Informatika, STMIK AMIKOM, Yogyakarta.
[9] Yuri ariyanto, 2009. "Algoritma RC4 Dalam Protekisi Transmisi Dan Hasil Query Untuk ORDBMS POSTGRESQL". Jurnal Informatika, Vol.10, No. 1 53 – 59.
[10] Jogiyanto.HM. 2005."Analisis dan Desain Sistem Informasi", ANDI. Yogyakarta
[11] Adi Nugroho, 2010,"Rekayasa Perangkat Lunak menggunakan UML dan Java", penerbit ANDI : Yogyakarta
[12] Priyanto, Rahmat, 2009, "Langsung Bisa Visual Basic.Net 2008" ,Penerbit ANDI, Yogyakarta.

Submitted: **31/03/2020**; Accepted: **29/04/2020**; Published: **30/04/2020**