Enkripsi SMS dengan Menggunakan *One Time Pad* (OTP) dan Kompresi Lempel-Ziv-Welch (LZW)

Fitri Diani[1], Yudi Widhiyasana[2]

***Abstract*—Short Message Service (SMS) is one of the features on a mobile phone. This feature is widely used because it is easy to use and does not require the latest telecommunications network connection. Short messages in the form of SMS consists of 140 characters at the most. The message is sent through infrastructure in telecommunications providers. Using this process, there is a possibility that the sent message is leaked. Therefore, data encryption is required to maintain the message confidentiality. Unfortunately, encryption mechanism uses cipher to encrypt data, which causes another problem. The type of cipher is symmetric or asymmetric, and both cipher mechanism will increase the length of the sent messages. In this paper, One Time Pad encryption method and LZW compression method is used to optimize the message length.**

***Intisari*—*Short Message Service* (SMS) adalah salah satu fitur pada telepon genggam. Fitur ini banyak digunakan karena mudah dan tidak membutuhkan koneksi jaringan telekomunikasi yang terbaru. Pesan singkat berupa SMS paling banyak terdiri dari 140 karakter. Pesan ini pada prosesnya melalui infrastruktur di penyedia jasa telekomunikasi. Hal ini sangat memungkinkan terjadi kebocoran pesan yang terkirim. Untuk itu diperlukan suatu enkripsi data agar kerahasiaan pesan dapat terjaga. Tetapi permasalahan enkripsi adalah tentang *cipher* yang digunakan. Jenis *cipher* yang ada adalah *symmetric* atau ataupun *asymmetric*. Karena *cipher* tersebut akan menambah panjang pesan yang dikirim. Pada penelitian ini akan dicoba menggunakan metode enkripsi *One Time Pad* dan metode kompresi LZW agar panjang pesan yang terkirim dapat dioptimalkan.**

***Kata kunci*—SMS, LZW, OTP, Keamanan Data.**

I. PENDAHULUAN

Di Indonesia, *Short Message Service* (SMS) pada perkembangannya digunakan oleh banyak orang untuk saling berkomunikasi, menyebarluaskan informasi, dan digunakan juga untuk transaksi *mobile banking*. Data dari Asosiasi Telekomunikasi Seluler Indonesia (ATSI) menyebutkan bahwa jumlah SMS yang terkirim pada tahun 2011 mencapai 260 miliar SMS [1].

Secara umum penyedia jasa telekomunikasi di Indonesia tidak menjamin kerahasiaan SMS yang dikirimkan oleh pengguna telepon seluler. Di lain pihak, SMS yang dikirim pengguna terkadang merupakan pesan yang bersifat rahasia dan pribadi, sehingga kerahasiaan pesan menjadi sangat penting untuk dijaga dari pihak yang tidak berhak mendapatkannya. Untuk mendukung penggunaan SMS dan meningkatkan kerahasiaan pesan yang dikirim, diperlukan mekanisme pengamanan pesan SMS. Salah satu alternatif solusinya adalah dengan melakukan enkripsi di pengirim dan dekripsi di penerima. Hal ini dapat mengurangi risiko kebocoran pesan pada SMS. Dengan menggunakan program yang tersedia dari vendor telepon seluler, tentu saja hal ini tidak dapat dilakukan. Untuk itu, diperlukan telepon seluler yang mendukung penggunaan aplikasi atau program yang dikembangkan oleh pihak lain. Dalam hal ini, sistem operasi pada telepon seluler menjadi perhatian khusus jika aplikasi enkripsi dan dekripsi ini dikembangkan.

Berbagai metode sudah banyak dikembangkan di dunia untuk membantu pengamanan SMS. Salah satu metode yang cukup andal adalah metode enkripsi *one time pad* (OTP) dengan mengunakan *hashing* pada enkripsi dan kompresinya [2].

Pengunaan metode ini akan mengakibatkan data menjadi lebih panjang karena *cipher* yang digunakan disertakan di dalam pesan yang dikirim. Untuk itu, diperlukan metode kompresi yang dapat mengefisienkan jumlah karakter yang dikirim. Untuk itu, diusulkan penggunaan enkripsi OTP dipadukan dengan metode kompresi Lempel-Ziv-Welch (LZW). Dengan memanfaatkan metode ini, layanan SMS dapat menjadi lebih aman karena algoritme enkripsi yang digunakan cukup tangguh dan kunci yang dipakai sukar dipecahkan.

II. PEMBAHASAN

A. Short Message Service (SMS)

SMS adalah layanan global dengan sistem komunikasi nirkabel yang mentransmisikan pesan teks antara dua atau lebih telepon seluler dan sistem eksternal seperti surat elektronik, *pager*, dan pesan suara [3]. Sebuah pesan SMS maksimal terdiri atas 140 *bytes*, atau dengan kata lain sebuah pesan bisa memuat 140 karakter 8-bit, 160 karakter 7-bit, atau 70 karakter 16-bit untuk bahasa Jepang, bahasa Mandarin, dan bahasa Korea yang menggunakan Hanzi (aksara Kanji/Hanja). Selain 140 *bytes* ini, ada data-data lain yang termasuk. Adapula beberapa metode untuk mengirim pesan yang lebih dari 140 *bytes*, tetapi pengguna harus membayar lebih dari sekali [4].

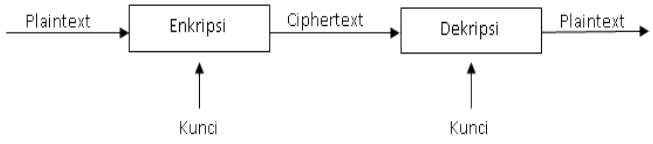
Layanan dari titik ke titik SMS menyediakan mekanisme pengiriman pesan pendek ke dan dari perangkat nirkabel seperti telepon seluler. Layanan ini menggunakan *Short Message Service Centre* (SMSC) yang berfungsi sebagai penyimpan sementara dan penerus pesan yang dikirim. Oleh karena itu, sebuah jaringan nirkabel harus memiliki kemampuan untuk mencari tujuan dan mengirim pesan antara SMSC dan perangkat tanpa kabel tersebut [5]. Pesan SMS memiliki beberapa karakteristik yang penting, yaitu sebagai berikut [5].

[1,2] *Dosen, JTK Politeknik Negeri Bandung Jl. Gegerkalong Hilir Ds. Ciwaruga, Bandung (40012) INDONESIA (tlp: 022 2013789; fax: 022-2013889; e-mail: fitri.d14ny@gmail.com; goezthel@gmail.com)*

1. Pesan SMS yang sampai atau tidak sama sekali akan diberikan informasi *(report)* status pengiriman pesan SMS.
2. Berbeda dengan fungsi panggilan, sekalipun saat mengirimkan SMS, telepon seluler tujuan tidak aktif, bukan berarti pengiriman SMS gagal. Namun, SMS akan ditampung dulu oleh SMSC sebelum diteruskan ke telepon tujuan setelah aktif.
3. Lebar pita *(bandwidth)* yang digunakan rendah.

B. One Time Pad

Kriptografi adalah suatu ilmu atau seni mengamankan pesan dan dilakukan oleh *cryptographer*. Sedangkan *cryptanalysis* adalah suatu ilmu dan seni membuka (*breaking*) *ciphertext* dan orang yang melakukannya disebut *cryptanalyst*. Ditinjau dari terminologinya, kata kriptografi berasal dari bahasa Yunani, yaitu *kryptos*, yang berarti menyembunyikan, dan *graphein*, yang berarti menulis, sehingga kriptografi dapat didefinisikan sebagai ilmu yang mengubah informasi dari keadaan/bentuk normal (dapat dipahami) menjadi bentuk yang tidak dapat dipahami [6]. Secara sederhana, proses kriptografi dapat digambarkan seperti pada Gbr. 1.



Gbr. 1 Proses kriptografi.

OTP adalah salah satu contoh metode kriptografi dengan algoritme jenis simetri, sehingga kunci yang digunakan untuk proses enkripsi sama dengan kunci yang digunakan untuk proses dekripsi.

OTP adalah algoritme kriptografi yang diklaim sempurna. OTP (*pad* = kertas *blocknote*) berisi deretan karakter-karakter kunci yang dibangkitkan secara acak.

Penerima pesan memiliki salinan (*copy*) *pad* yang sama. Satu *pad* hanya digunakan sekali (*one-time*) saja untuk mengenkripsi pesan. Sekali telah digunakan, *pad* dihancurkan supaya tidak dipakai kembali untuk mengenkripsi pesan yang lain [7].

Suatu algoritme dikatakan aman apabila tidak ada cara untuk menemukan *plaintext*-nya. Sampai saat ini, hanya algoritme OTP yang dinyatakan tidak dapat dipecahkan meskipun diberikan sumber daya yang tidak terbatas. Proses enkripsi dapat dilakukan dengan persamaan matematis seperti pada (1).

$$C_i = (P_i + K_r) \bmod 26 \quad (1)$$

Sedangkan untuk proses dekripsi dapat dilakukan dengan (2).

$$P_i = ((C_i - K_r) + 26) \bmod 26 \quad (2)$$

Dari (1) dan (2) dapat diketahui

C_i = pergeseran karakter pada *ciphertext*,

P_i = pergeseran karakter pada *plaintext*,

K_r = Kunci dalam bentuk decimal yang dihasilkan dari tabel konversi.

Berikut adalah contoh proses enkripsi.

Plaintext : ONETIMEPAD

Kunci : TBFRGFARFM

Misalkan A = 0, B = 1, …, Z = 25.

ciphertext: HOJKOREGHP

yang diperoleh dengan cara sebagai berikut.

(O + T) mod 26 = H

(N + B) mod 26 = O

(E + F) mod 26 = J, dan seterusnya

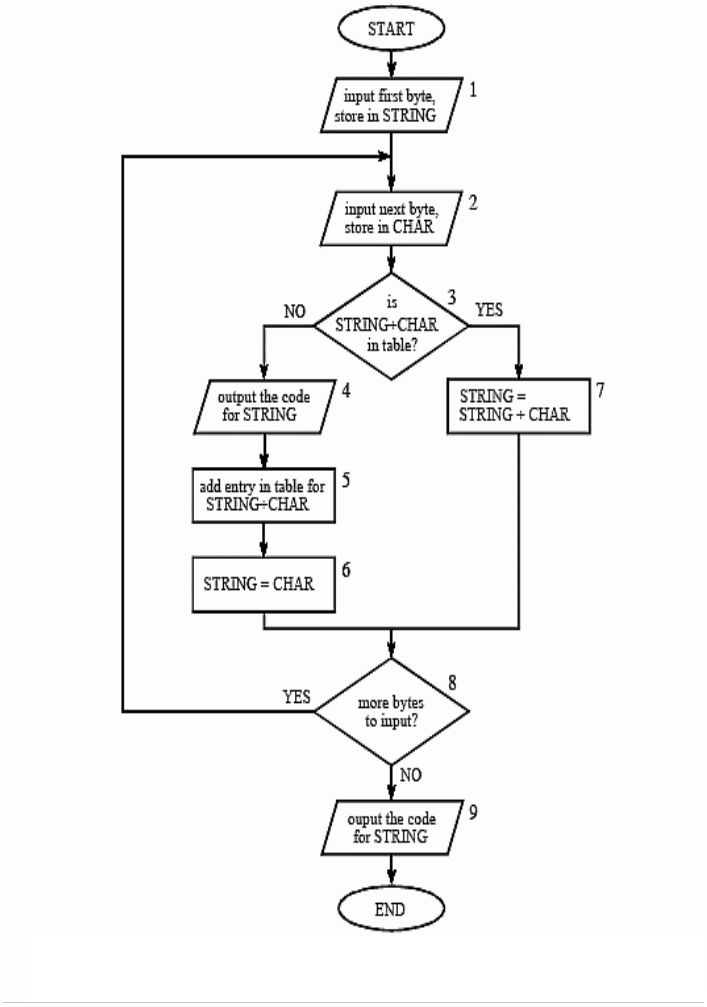
C. Lempel-Ziv-Welch (LZW)

Algoritme LZW dikembangkan oleh Abraham Lempel, Jacob Ziv, dan Terry Welch dan dipublikasikan pada tahun 1984 oleh Terry Welch. LWZ dirancang sebagai peningkatan dari algoritme LZ78. Algoritme ini mereduksi jumlah *token* yang dibutuhkan menjadi satu simbol saja. Simbol ini merujuk kepada *index* dalam *dictionary*. Proses kerjanya mirip dengan algoritme LZ78, tetapi jika pada algoritme LZ78 *dictionary* dimulai dari keadaan kosong, LZW mengisi *dictionary* ini dengan seluruh simbol alfabet yang dibutuhkan. Pada kasus yang umum, 256 *index* pertama dari *dictionary* diisi dengan karakter ASCII dari 0-255. Karena *dictionary* telah diisi dengan semua kemungkinan karakter terlebih dahulu, maka karakter masukan pertama akan selalu dapat ditemukan dalam *dictionary*. Inilah yang menyebabkan *token* pada LZW hanya memerlukan satu simbol saja, yang merupakan *pointer* pada *dictionary* [8].

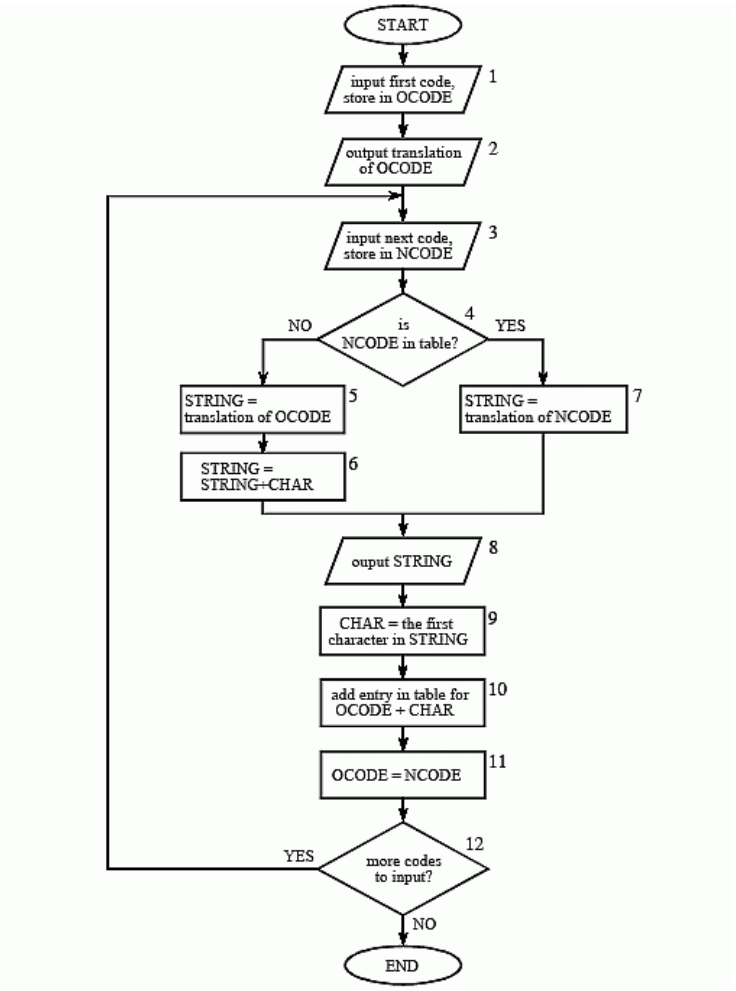
Prinsip kerja LZW dimulai dengan membaca karakter masukan satu per satu dan diakumulasi pada sebuah *string* I. Lalu, dilakukan pencarian dalam *dictionary*, terdapat *string* I atau tidak. Selama *string* I ditemukan di dalam *dictionary*, *string* ini ditambahkan dengan satu karakter berikutnya, lalu dicari lagi dalam *dictionary*. Pada saat tertentu, menambahkan satu karakter x pada *string* I akan menyebabkan tidak ditemukan dalam *dictionary*. *String* I ditemukan, tetapi *string* Ix tidak. Dalam tahap ini, algoritme akan menulis *index* dari string I sebagai keluaran, menambahkan *string* Ix ke dalam *dictionary*, dan menginisialisasikan *string* I dengan x, lalu proses dimulai lagi dari awal. Kumpulan keluaran yang berupa angka-angka inilah yang merupakan hasil kompresi. Karena menggunakan konsep *dictionary* yang sama seperti LZ78, pengorganisasian *dictionary* juga diperlukan, misalnya besar *dictionary* yang disediakan dan hal yang akan dilakukan jika *dictionary* sudah penuh. Diagram alir proses kompresi ditunjukkan pada Gbr. 2, sedangkan diagram alir proses dekompresi diperlihatkan pada Gbr. 3.

D. Android

Android adalah sistem operasi yang berbasis Linux untuk telepon seluler seperti telepon pintar dan komputer tablet. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat peranti lunak untuk telepon seluler. Kemudian, untuk mengembangkan Android, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, TMobile, dan Nvidia.



Gbr. 2 Diagram alir kompresi LZW.



Gbr. 3 Diagram alir dekompresi LZW.

Pada saat perilisan perdana Android, 5 November 2007, Android bersama Open Handset Alliance menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode-kode Android di bawah lisensi Apache, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler.

Di dunia ini terdapat dua jenis distributor sistem operasi Android. Pertama yang mendapat dukungan penuh dari Google atau *Google Mail Services* (GMS) dan kedua adalah yang benar–benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai *Open Handset Distribution* (OHD).

Dari distribusi sistem tersebut terdapat informasi tentang *native language* dalam pengembangan aplikasi di sistem Android. Dalam hal ini, bahasa yang harus digunakan adalah bahasa pemrograman Java.

E. Pengembangan Perangkat Lunak

Pengembangan perangkat lunak dilakukan pada sistem operasi Android dengan menggunakan bahasa Java. Versi Android yang dijadikan target pengembangan adalah Marshmallow. Pengembangan aplikasi SMS di Android ini harus memenuhi kaidah-kaidah sebagai berikut.

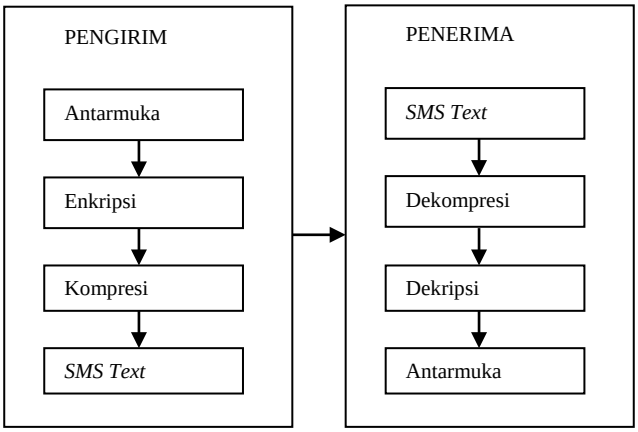
1) Memenuhi Protocol Pengiriman SMS: Pada saat mengirim pesan teks SMS, ada empat rangkaian karakter/pengkodean yang relevan untuk dipertimbangkan, yaitu ASCII, GSM 03.38, dan Unicode. ASCII atau *American Standard Code for Information Interchange* adalah suatu standar internasional dalam kode huruf dan simbol yang dibuat oleh Amerika Serikat. GSM 03.38 adalah pengembangan ASCII dengan menambahkan karakter dari banyak bahasa Eropa, misalnya, karakter seperti Σ, Ç, dan Æ. Perangkat seluler diwajibkan mendukung pengiriman pesan teks SMS menggunakan set karakter GSM ini. Standar Unicode memungkinkan pengguna mengirim pesan teks SMS menggunakan karakter dari bahasa tertulis mana pun. Perangkat sambungan sepenuhnya mendukung pengiriman pesan teks SMS menggunakan set karakter Unicode, tetapi pesan dibatasi hingga 70 karakter per SMS [5].

2) Menggunakan Algoritme OTP pada Proses Enkripsi dan Dekripsi: OTP sebagai salah satu algoritme kriptografi diklaim sebagai algoritme yang mampu menghasilkan *ciphertext* yang tidak dapat dipecahkan (*unbreakable cipher*). Untuk memecahkan *ciphertext* diperlukan kunci yang digunakan sebagai pergeseran karakter. Kunci tersebut harus

sesuai dengan *ciphertext* yang akan dipecahkan. Dengan begitu, maka pada aplikasi yang akan dikembangkan, SMS yang akan dikirim harus berisi *ciphertext* beserta kunci untuk memecahkannya. Namun, OTP memiliki keterbatasan yang menyebabkan algoritme ini tidak dipergunakan secara universal. Panjang kunci yang harus sama dengan panjang pesan menjadikan algoritme ini sulit diimplementasikan pada lingkungan yang mempunyai keterbatasan memori [9]. Hal inilah yang menyebabkan OTP membutuhkan sebuah mekanisme kompresi. Proses enkripsi pada aplikasi ini adalah proses yang mempunyai masukan dan keluaran berupa teks. Perbedaannya adalah keluaran dari metode OTP ini mengembalikan dua nilai, yaitu *chipertext* dan kunci sebagai pemecah *chipertext*, sesuai dengan paparan pada bagian sebelumnya [10]. Oleh karena itu, perlu dilakukan penggabungan kedua nilai tersebut agar hasil keluaran dari proses ini hanya mengembalikan satu nilai saja. Hal tersebut dilakukan dengan cara menambahkan karakter '/' sebagai pemisah antara *chipertext* dan kucni. Berikut adalah contoh proses pada OTP.

Plaintext : ONETIMEPAD
Kunci : TBFRGFARFM
Output proses OTP : ONETIMEPAD/TBFRGFARFM

Tampak bahwa pesan yang akan dikirim menjadi dua kali lipat beserta karakter pemisah '/' dari pesan yang akan dikirim.

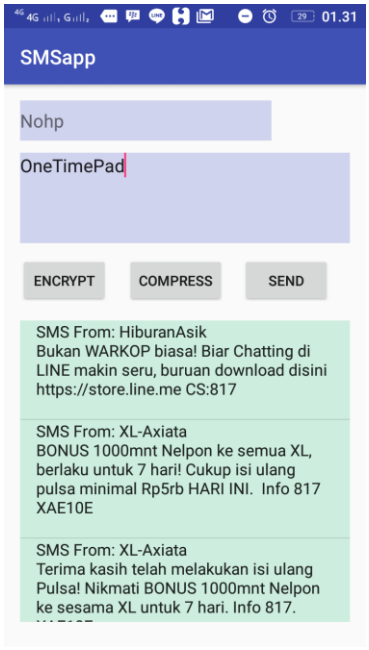


Gbr. 4 Mekanisme aplikasi.

3) Menggunakan Algoritma LZW pada Proses Kompresi dan Dekompresi: LZW merupakan salah satu algoritme kompresi yang bersifat *lossless* (tidak ada data yang hilang pada saat kompresi). Mekanisme yang ada pada algoritme ini menuntut penggunaan struktur data yang dinamis selama proses kompresi atau dekompresi. *Indexing Tree* pada proses ini menjadi komponen utama dalam proses seperti yang telah dijelaskan pada bagian sebelumnya. Agar dapat berjalan pada proses teks untuk mengolah SMS, *Indexing Tree* harus disiapkan untuk karakter-karakter yang didukung oleh SMS pada sistem Android [11]. Pada pesan yang akan didekripsi (hasil dekompresi dari LZW), karakter '/' menjadi acuan untuk memisahkan *chipertext* dan kunci. Setiap karakter pada *chipertext* akan digeser sesuai dengan karakter kunci sesuai dengan posisi, sebagaimana telah dijelaskan pada bagian sebelumnya. Dengan memperhatikan kaidah-kaidah tersebut, gambaran mekanisme aplikasi yang akan dikembangkan diperlihatkan pada Gbr. 4.

F. Hasil Pengembangan Aplikasi

Sesuai dengan penjelasan sebelumnya, aplikasi dikembangkan dengan Android Studio dan berjalan pada versi Marshmallow. Adapun hasil dari pengembangan aplikasi ditunjukkan pada Gbr. 5 sampai Gbr. 7.



Gbr. 5 Masukan *plaintext*.

Pada Gbr. 5 sampai Gbr. 7 tampak bahwa fungsi dekripsi dan kompresi sudah berjalan sesuai rancangan. Kemudian akan disiapkan mekanisme pengujian terhadap aplikasi ini.

G. Rencana Pengujian Aplikasi

Pengujian perangkat lunak dilakukan berdasarkan hal-hal sebagai berikut ini.

1. Panjang karakter yang akan dikirimkan via SMS.
2. Jenis karakter yang akan dikirimkan via SMS.

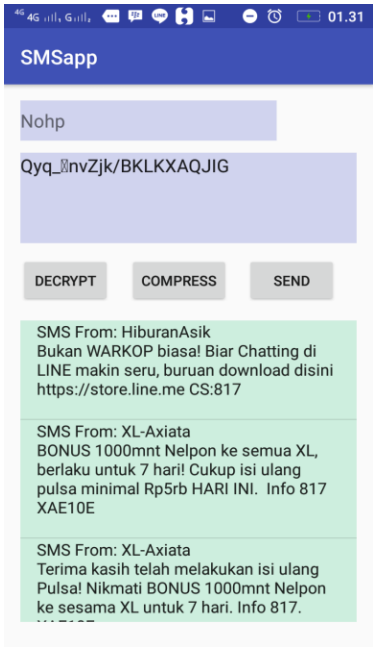
Panjang karakter menjadi fokus pertama karena panjang karakter yang dihasilkan oleh aplikasi maksimal 140 karakter. Hal ini akan memberikan gambaran tentang efektivitas dari proses enkripsi dan kompresi yang dilakukan. Untuk itu, pada pengujian ini desain SMS yang akan dikirimkan direncanakan memiliki panjang sebagai berikut.

1. SMS dengan karakter 1-30 (pendek).
2. SMS dengan karakter 30-100 (sedang).
3. SMS dengan karakter 100-140 (panjang).

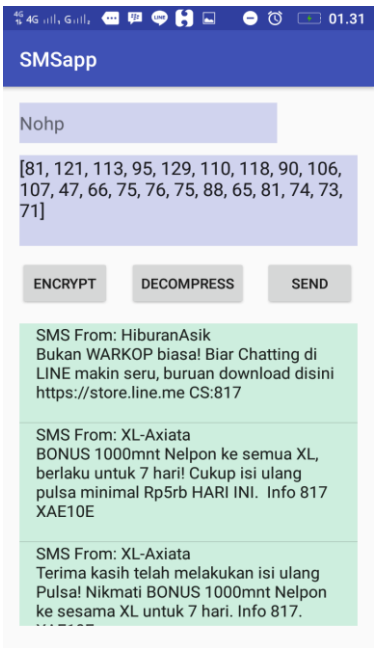
Kemudian, dari jenis karakter pada SMS, tipe data yang ada adalah alfanumerik dan simbol. Pada kedua tipe data ini tidak seluruh kode ASCII yang ada didukung oleh *keypad* pada SMS. Oleh karena itu, proses pengacakan teks awal SMS sebelum dienkripsi harus dibatasi pada karakter-karakter yang

ada pada *keypad* SMS. Adapun rencana isi dari SMS yang dikirim adalah sebagai berikut.

1. Bertipe data alfanumerik.
2. Bertipe data simbol.
3. Bertipe data alfanumerik dan simbol.



Gbr. 6 Hasil enkripsi.



Gbr. 7 Hasil kompresi.

Adapun isi dari SMS akan diacak sehingga dapat dilihat efektivitas proses *pairing* pada OTP dan kompresi pada LZW. Efektivitas dapat dilihat karena proses *pairing* pada OTP juga pasti dilakukan pengacakan. Hal ini menyebabkan kamus pada tahap *indexing* pada proses kompresi akan berbeda-beda untuk setiap panjang SMS yang diujikan.

III. PENGUJIAN

Sesuai dengan rencana pengujian pada paparan sebelumnya, data uji menggunakan set karakter GSM 03.38. Karakter yang didukung oleh GSM 03.38 diperlihatkan pada Gbr. 8.

				b7	0	0	0	0	1	1	1	1
				b6	0	0	1	1	0	0	1	1
				b5	0	1	0	1	0	1	0	1
b4	b3	b2	b1		0	1	2	3	4	5	6	7
0	0	0	0	0	@	Δ	SP	0	¡	P	¿	p
0	0	0	1	1	£	_	!	1	A	Q	a	q
0	0	1	0	2	$	Φ	"	2	B	R	b	r
0	0	1	1	3	¥	Γ	#	3	C	S	c	s
0	1	0	0	4	è	Λ	¤	4	D	T	d	t
0	1	0	1	5	é	Ω	%	5	E	U	e	u
0	1	1	0	6	ù	Π	&	6	F	V	f	v
0	1	1	1	7	ì	Ψ	'	7	G	W	g	w
1	0	0	0	8	ò	Σ	(	8	H	X	h	x
1	0	0	1	9	Ç	Θ	)	9	I	Y	i	y
1	0	1	0	10	LF	Ξ	*	:	J	Z	j	z
1	0	1	1	11	Ø	1)	+	;	K	Ä	k	ä
1	1	0	0	12	ø	Æ	,	<	L	Ö	l	ö
1	1	0	1	13	CR	æ	-	=	M	Ñ	m	ñ
1	1	1	0	14	Å	ß	.	>	N	Ü	n	ü
1	1	1	1	15	å	É	/	?	O	§	o	à

Gbr. 8 Set karakter GSM 03.38.

TABEL I
CONTOH UJI VARIANS

Karakter Set	Contoh Uji Avariansi	Set
A,I,M,N,K,D,H	MAIN, MAKAN, DIAM, KAIN, NIKAH, HAID, DIDIH, IMAN, dll	1
	AIM, MIA, KIM, KIKIM, KIH, IHK, MIADA, KINKIN, dll	2

TABEL II
HASIL PENGUJIAN

Tipe Data	Panjang	Set	Panjang Hasil Kompresi
Numerik	1 s/d 30	1	71,7 %
	31 s/d 100	1	66,3 %
	101 s/d 150	1	61,8 %
	1 s/d 30	2	66,9 %
	31 s/d 100	2	62,6 %
	101 s/d 150	2	57,9 %
Alfabet	1 s/d 30	1	72,9 %
	31 s/d 100	1	68,1 %
	101 s/d 150	1	62,7 %
	1 s/d 30	2	67,1 %
	31 s/d 100	2	63,9 %
	101 s/d 150	2	59,1 %
Alfanumerik	1 s/d 30	1	73,6 %
	31 s/d 100	1	69,6 %
	101 s/d 150	1	63,8 %
	1 s/d 30	2	67,8 %
	31 s/d 100	2	64,4 %
	101 s/d 150	2	60,1 %

Dengan karakter set tersebut, pengujian dilakukan sesuai dengan desain awal pengujian, yaitu

1. menggunakan karakter numerik,
2. menggunakan karakter alphabet, dan
3. menggunakan karakter alfanumerik (termasuk simbol).

Kemudian dari kemungkinan karakter yang ada akan diujikan dengan menggunakan panjang pesan SMS yang berbeda-beda. Selain panjang pesan SMS, juga dilakukan proses pengacakan dari varian data uji yang ada.

Set 1 pada Tabel I adalah kata-kata yang terdapat pada Kamus Besar Bahasa Indoensia (KBBI), sedangkan set 2 adalah kata-kata yang diacak dari karakter set dengan panjang yang dapat ditentukan.

Dari Tabel II yang berisi hasil pengujian, tampak bahwa *index* yang disusun pada metode LZW mempunyai efektivitas yang berbeda-beda, bergantung pada data masukannya. Oleh karena itu, pengacakan pada proses OTP dapat menjadi pembeda kepada data masukan untuk proses kompresi LZW.

Penelitian sebelumnya yang menggunakan metode enkripsi yang berbeda telah mengungkapkan hal yang serupa dengan hasil pengujian ini [12].

IV. KESIMPULAN

Hasil pengujian memperlihatkan bahwa *ciphertext* mempunyai panjang dua kali lipat dari *plaintext* dan hasil kompresi menunjukkan panjang teks yang beragam, tergantung *index tree* pada proses kompresi. Pada penelitian sebelumnya telah diungkapkan hal yang serupa walaupun menggunakan metode enkripsi yang berbeda.

Dari pengujian yang dilakukan, dapat disimpulkan bahwa enkripsi OTP yang digunakan dapat memberikan data enkripsi yang baik sehingga dapat menjadi data masukan pada proses kompresi LZW. Hasil pengujian memperlihatkan bahwa pengacakan karakter kunci pada OTP akan mempengaruhi *index tree* yang digunakan pada proses kompresi LZW. Secara umum, penggabungan kedua metode ini dapat menjadi solusi permasalahan. Hanya saja ada kemungkinan masalah yang timbul jika karakter yang digunakan lebih dari 120 karakter. Hasil pengujian menunjukkan bahwa *plaintext* yang berjumlah di atas 120 karakter dapat menghasilkan keluaran hasil kompresi di atas batas pengiriman SMS, yaitu 150 karakter. Hal ini dapat terjadi jika *plaintext* banyak menggunakan singkatan-singkatan yang belum terdefinisi.

REFERENSI

[1] Firman Nugraha (2012) Jumlah Pelanggan Seluler di Indonesia mendekati Jumlah Penduduk Indonesia, [Online] http://teknojurnal.com/jumlah-pelanggan-seluler-di-indonesia-hampir-mendekati-jumlah-penduduk-indonesia/, tanggal akses: 10 Maret 2016.

[2] N.J. Croft dan M.S. Olivier, *Using an Approximated One-Time Pad to Secure Short Messaging Service (SMS)*, D Browne (ed), *Southern African Telecommunication Networks and Applications Conference 2005 (SATNAC 2005) Proceedings*, Vol 1, hal. 71-76, 2005.

[3] B. Patil, "SMS Security Using RC4 & AES", *Indian J.Sci.Res.*, Vol. 11. No. 1, hal 034-038, 2015.

[4] (2009) GSM Association Homepage, [Online] http://www.gsma.com/, tanggal akses: 11 Maret 2016.

[5] (2016) SMS Service, [Online] http://www.smsitaly.com/eng/smsc.asp/, tanggal akses: 20 Maret 2016.

[6] H. Bodur dan R. Kara, "Secure SMS Encryption using RSA Encryption Algorithm on Android Message Application," *ISITES2015*, 2015, hal. 1-10.

[7] M. Rinaldi, "Kriptografi", Diktat Kuliah, Institut Teknologi Bandung, 2006.

[8] Linawati dan H.P. Panggabean, "Perbandingan Kinerja Algoritma Kompresi Huffman, LZW, dan DMC pada Berbagai Tipe File," *INTEGRAL*, Vol. 9, No. 1, hal. 7-16, 2004.

[9] T.St. Denis, *Cryptography for Developers*, Kanada: Syngress Publishing, 2007.

[10] S. Sugeng, "Rancang Bangun Aplikasi Pesan Menggunakan Algoritma Vigenere Cipher dan One Time Pad," Skripsi, Universitas Dian Nuswantoro, Semarang, Indonesia, Nov. 2015.

[11] S. Jha., U. Dutta, P. Gupta., "SMS Encryption using NTRU Algorithms on Android Application," *International Journal of Scientific Engineering and Applied Science (IJSEAS)*, Vol. 2, No. 1, hal. 331-338, 2016

[12] I.W.D. Satriawan, I.G.M.A. Sasmita, I.P.A. Bayupati, "Aplikasi Enkripsi SMS dengan Metode RSA pada Smartphone Berbasis Android," *MERPATI*, Vol. 2, No. 2, hal. 127-134, Agt. 2014.