SISTEM ENKRIPSI MENGGUNAKAN ALGORITMA AES-128 PADA PROTOTYPE COMMUNITY MESSENGER BERBASIS ANDROID

ENCRYPTION SYSTEM USING AES-128 ALGORITHM ON PROTOTYPE COMMUNITY MESSENGER ANDROID-BASED

Andi Wijaya(andiwijayaar@gmail.com)[1]
[1]Prodi S1 Sistem Komputer, Fakultas Teknik, Universitas Telkom
info@telkomuniversity.ac.id

Abstrak

***Instant messaging* (IM) saat ini sangat diminati oleh seluruh kalangan penduduk dunia, termasuk di Indonesia. Sebagian besar masyarakat yang menggunakan *smartphone* menggunakan aplikasi ini untuk mempermudah komunikasi. Dengan maraknya penggunaan IM, maka aspek keamanan data atau informasi pada IM juga perlu dipertimbangkan. Oleh sebab itu, penggunaan sistem enkripsi pada IM sangat dibutuhkan untuk menjaga keamanan data atau informasi saat berkomunikasi. Algoritma enkripsi dan dekripsi yang akan digunakan adalah Algoritma AES-128. Algoritma ini merupakan algoritma *stream cipher* dan menggunakan kunci simetris 128-bit. Dengan menggunakan algoritma ini, data atau informasi yang akan dikirim ke penerima akan lebih aman. Sistem enkripsi dan dekripsi menggunakan algoritma AES-128 ini akan diimplementasikan pada aplikasi *Prototype Community Messenger* berbasis sistem operasi *Android* yang memiliki performansi baik, terlihat dari nilai *Avalanche Effect* dengan rata-rata bernilai 0,539069. Perbandingan waktu enkripsi dan dekripsi pesan, dimana semakin banyak masukkan pesan oleh pengguna maka waktu enkripsi dan dekripsi semakin lama.**

Kata Kunci : ***instant messaging* (IM), enkripsi, algoritma AES-128, android**

Abstract

Instant messaging (IM) is currently in great demand by all the population of the world, including in Indonesia. Most people who use a smartphone to use this application to facilitate communication. With the widespread use of IM, then the security aspects of the data or information on IM also need to be considered. Therefore, the use of encryption on the IM system is needed to maintain the security of data or information when communicating. Encryption and decryption algorithms that will be used is AES-128 algorithm. This algorithm is a stream cipher algorithm and uses 128-bit symmetric key. By using this algorithm, the data or information to be sent to the recipient will be safer. System encryption and decryption using AES-128 algorithm will be implemented on Prototype Community Messenger application based Android operating system that has a good performance, seen from the Avalanche Effect with an average worth 0.539069. Comparison of time encryption and decryption of messages, where a growing number of messages by the user enter the encryption and decryption time is getting longer.

Keywords : instant messaging (IM), encryption, AES-128 algorithm, android

I. Pendahuluan

Telepon selular merupakan alat komunikasi yang sudah dipakai oleh sebagian besar penduduk di dunia. Pada awalnya pengiriman pesan digital mengunakan telepon seluler dilayani olehprovider telekomunikasi melalui *Short Messaging Service* (SMS) atau *Multimedia Messaging Service* (MMS).

Namun dengan semakin berkembangnya teknologi telepon seluler yang berbasis *smartphone*, banyak para pengembang aplikasi maupun *vendor smartphone* mengembangkan aplikasi *Instant Messaging* (IM) sebagai layanan pengiriman pesan digital. IM menjadi semakin populer karena dianggap memiliki banyak kelebihan seperti memungkinkan seseorang untuk melakukan percakapan *(chat)private* dengan orang lain secara *real time*.

Seiring dengan semakin banyak digunakannya layanan IM, maka kekhawatiran mengenai keamanan data atau pesan yang akan dikirim sangat tinggi. Oleh karena itu, suatu proses enkripsi diperlukan untuk mengamankan pesan teks yang dikirimkan. Sistem enkripsi dapat meningkatkan tingkat keamanan pesan. Hal ini dapat mengurangi bocornya informasi kepada

pihak-pihak yang tidak berkepentingan.

Algoritma AES-128 adalah sistem kriptografi kunci simetris 128-bit. Alasan dipilihnya algoritma ini adalah karena algoritma ini tergolong algoritma *stream chipper*. Selain itu, algoritma ini bisa diimplementasikan secara efisien pada berbagai prosesor maupun hardware khusus. Oleh karena itu dengan memanfaatkan keunggulan dari algoritma AES-128 dan melihat masalah keamananan data pesan yang tengah dihadapi, maka dalam penelitian ini penulis akan membuat sebuah sistem enkripsi *prototype community messaging* berbasis Android.

II. Dasar Teori

2.1 Komunikasi Data

Komunikasi data adalah proses pengiriman dan penerimaan data/informasi dari dua atau lebih *device* (*personal computer/ printer / handset* /dan alat komunikasi lain) yang terhubung dalam sebuah jaringan. Baik lokal maupun yang luas, seperti internet. Secara umum ada dua jenis komunikasi data, yaitu: Melalui Infrastruktur Terestrial dan Melalui Satelit.

Melalui Infrastruktur Terestrial Menggunakan media kabel dan nirkabel sebagai aksesnya. Beberapa layanan yang termasuk teresterial antara lain: Sambungan Data Langsung(SDL), Frame Relay, VPN MultiService dan Sambungan Komunikasi Data Paket (SKDP).

Melalui Satelit Menggunakan satelit sebagai aksesnya. Biasanya wilayah yang dicakup akses satelit lebih luas dan mampu menjangkau lokasi yang tidak memungkinkan dibangunnya infrastruktur terestrial namun membutuhkan waktu yang lama untuk berlangsungnya proses komunikasi.

2.2 Algoritma AES-128

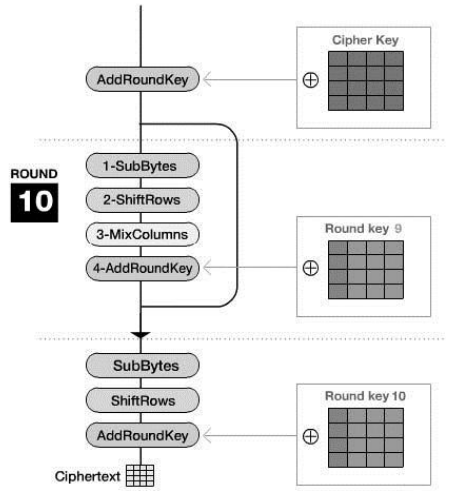
Advanced Encryption Standard (AES) dipublikasikan oleh NIST *(National Institute of Standard and Technology)* pada tahun 2001. AES merupakan block kode simetris untuk menggantikan DES *(Data Encryption Standard)*. DES terbukti menjadi algoritma enkripsi yang aman didunia selama puluhan tahun. Pada tahun 1990 panjang kunci DES dianggap terlalu pendek dan terbukti pada tahun 1998, 70 ribu PC di internet berhasil membobol satu kunci DES dalam tempo 96 hari, tahun 1999 dalam tempo 22 hari. Karena sudah berhasil dipecahkan, maka dibuatlah mesin khusus untuk memecahkan algoritma DES yang mampu memecahkan 25% kunci DES dalam waktu 2,3 hari dan dapat memcahkan seluruh kunci DES dalam waktu rata-rata 4,5 hari.

Karena alasan tersebut maka kemudian diadakan kompetisi oleh NIST untuk mengganti algoritma DES. Melalui seleksi yang ketat, maka pada 2 Oktober 2000 terpilih algoritma Rijndael atau yang kita kenal sekarang algoritma AES sebegai pemenang.

AES mempunyai kunci 128, 192 dan 256 bit sehingga berbeda dengan panjang dari putaran Rijndael.

1. Proses Enkripsi

Proses enkripsi algoritma AES terdiri dari 4 jenis transformasi bytes, yaitu *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey*. Awal proses enkripsi, state akan mengalami transformasi byte *AddRoundKey*. Setelah itu *state* akan mengalami transformasi *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey* secara berulang sebanyak Nr. Proses ini disebut juga sebagai *round function*. Pada *Round* terkahir, proses berbeda dari sebelumnya dimana *state* tidak mengalami transformasi *MixColumns*.



Gambar 1 Proses Enkripsi AES

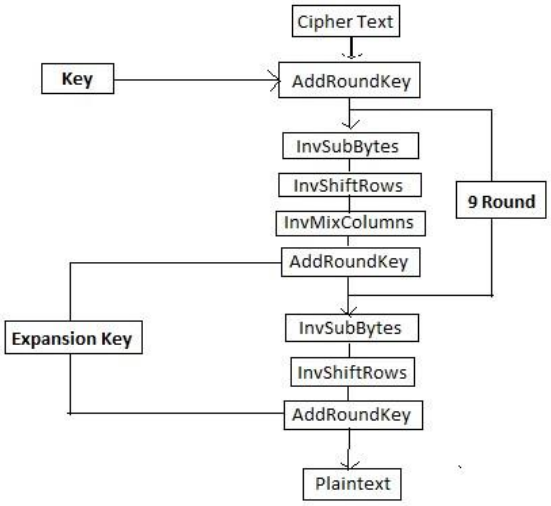
Garis besar Algoritma AES *Rijndael* yang beroperasi pada blok 128-bit dengan kunci 128-bit adalah sebagai berikut (di luar proses pembangkitan *round key*):

1. *AddRoundKey*: melakukan *XOR* antara *state* awal (*plainteks*) dengan *cipher key*. Tahap ini disebut juga *initial round***.**
2. *Round :* Putaran sebanyak *Nr* – 1 kali. Proses yang dilakukan pada setiap putaran adalah:
   a. *SubBytes*: substitusi *byte* dengan menggunakan table substitusi (*S-box*).
   b. *ShiftRows*: pergeseran baris-baris *array state* secara *wrapping*.
   c. *MixColumns*: mengacak data di masing-masing kolom *array state.*
   d. *AddRoundKey*: melakukan XOR antara *state* sekarang *round key*.
3. *Final round***:** proses untuk putaran terakhir:
   a. *SubBytes*
   b. *ShiftRows*
   c. *AddRoundKey*

2. Proses Dekripsi

Transformasi *cipher* dapat dibalikkan dan

diimplementasikan dalam arah yang berlawanan untuk menghasilkan *inverse cipher* yang mudah dipahami untuk algoritma AES. Transformasi byte yang digunakan pada *invers cipher* adalah *InvShiftRows*, *InvSubBytes*, *InvMixColumns*, dan *AddRoundKey*. Algoritma dekripsi dapat dilihat pada skema berikut ini :



Gambar 2 Prosese Dekripsi

a. InvShiftRows

InvShiftRows adalah transformasi byte yang berkebalikan dengan transformasi *ShiftRows*. Pada transformasi *InvShiftRows*, dilakukan pergeseran bit ke kanan sedangkan pada *ShiftRows* dilakukan pergeseran bit ke kiri.

b. InvSubBytes

InvSubBytes juga merupakan transformasi bytes yang berkebalikan dengan transformasi *SubBytes*. Pada *InvSubBytes*, tiap elemen pada *state* dipetakan dengan menggunakan tabel *Inverse S-Box*.

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xa	xb	xc	xd	xe	xf
0x	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
1x	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
2x	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
3x	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
4x	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
5x	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
6x	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
7x	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
8x	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
9x	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
ax	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
bx	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
cx	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
dx	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
ex	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
fx	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Gambar 3 Tabel Inverse S-box

c. *InvMixColumns*

Setiap kolom dalam *state* dikalikan dengan matrik perkalian dalam AES.

$$\begin{bmatrix} r_0 \\ r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} 14 & 11 & 13 & 9 \\ 9 & 14 & 11 & 13 \\ 13 & 9 & 14 & 11 \\ 11 & 13 & 9 & 14 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Gambar 4 InvMixColulmns

3. Proses Ekspansi Kunci

Algoritma AES mengambil kunci *cipher* dan melakukan rutin ekspansi kunci (*key expansion*) untuk membentuk *key schedule*. Ekspansi kunci menghasilkan total Nb(Nr+1) *word*. Algoritma ini membutuhkan set awal *key* yang terdiri dari Nb word , dan setiap *round* Nr membutuhkan data kunci sebanyak Nb *word*. Hasil *key schedule* terdiri dari *array* 4 byte *word* linear yang dinotasikan dengan [w_i]. *SubWord* adalah fungsi yang mengambil 4 byte *word* input dan mengaplikasikan *S-Box* ke tiap-tiap data 4 byte untuk menghasilkan *word output*.

2.3 Android Development

Seperti kita ketahui bahwa android adalah sebuah sistem operasi berbasis linux kernel yang sangat handal dan dirancag untuk gadget telepon selular. Adapun alasan memilih smartphone Android sebagai target implementasi adalah sebagai berikut:

a. Android menyediakan platform terbuka (open sources)
b. Android dapat menjalankan beberapa Aplikasi pada waktu yang bersamaan (multitasking OS).
c. Sistem notifikasi yang mudah dan lebih baik.

2.4 Android Software Development Kit (SDK)

Android SDK adalah *tools* API (*Application programming interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada platform Android menggunakan bahasa pemrograman Java. Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, middleware dan aplikasi kunci yang di release oleh Google. Saat ini disediakan Android SDK (software Development kit) sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada platform Android menggunakan bahasa pemrograman Java.

III. Pembahasan

3.1 Perancangan Sistem

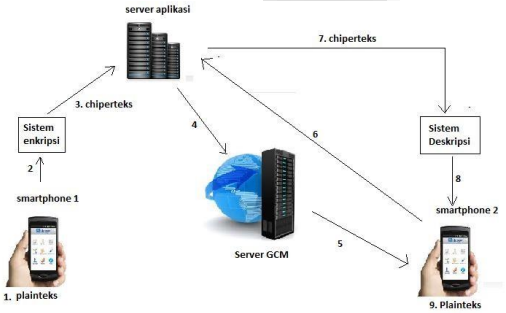
1. Kebutuhan Sistem

Kebutuhan sistem pada pembuatan aplikasi meliputi beberapa hal, diantaranya adalah :

a. Sistem Operasi Windows 7 Home Premium 64-bit
b. Eclipse
c. Java Development Kit Versi 1.6 (JDK)
d. Android Development Kit 18
e. Android SDK
f. Microsoft Office 2007
g. Microsoft Visio 2007
h. Evolus Pencil Versi 2.0.5

2. Model Sistem

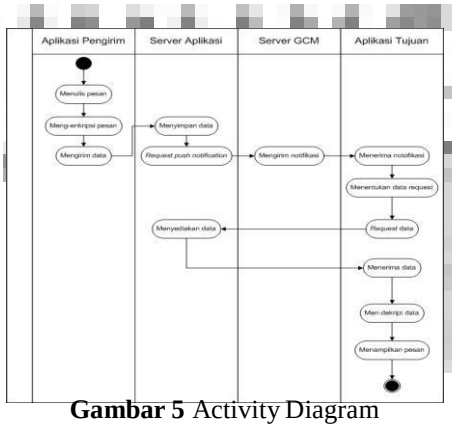
Model sistem perangkat lunak yang dibuat sebagai berikut.



Gambar 4 Model Sistem

3. Activity Diagram

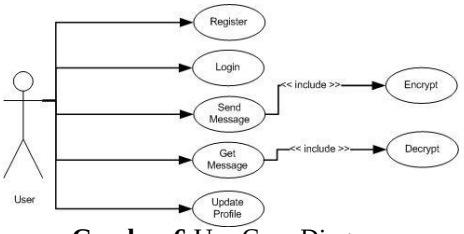
Diagram aktivitas menggambarkan suatu urutan proses yang terjadi pada sistem dari dimulainya aktivitas hingga aktivitas berhenti.



Gambar 5 Activity Diagram

4. Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem.



Gambar 6 Use Case Diagram

3.2 Pengujian Sistem

1. Pengujian Performansi

a. Sumber Daya

Pengujian sumber daya dilakukan untuk mengetahui seberapa besar memori yang dibutuhkan untuk menjalani aplikasi "*Prototye Community Messenger*". Pengukuran memori dilakukan menggunakan DDMS *(Dalvik Debug Monitor Server)* yang terdapat pada *Eclipse.* Untuk skenario pengamatan dengan *DDMS*, aplikasi dijalankan kemudian membuka seluruh menu aplikasi dan mengukur total *heap size* dan *allocated heap size* yang terlihat.

Dari 30 kali percobaan, didapat nilai *heapsize* aplikasi yang bervariasi. Nilai maksimum yang didapat pada percobaan adalah 20,980 MB dan nilia minimum 17,062 MB. Dari percobaan disimpulkan bahwa penggunaan memori untuk menjalankan aplikasi sebesar 68,882% dari alokasi memori yang diberikan untuk aplikasi.

b. Jaringan

Pengujian network ditujukan untuk menguji seberapa cepat pesan yang dikirimkan dari satu perangkat ke perangkat lain yang dituju. Pengujian dicoba dengan empat kemungkinan koneksi internet, yakni : EDGE-HSDPA, HSDPA-EDGE, HSDPA–HSDPA dan EDGE–EDGE. Pengujian dilakukan sebanyak 30 kali percobaan dengan isi pesan yang sama berdasarkan kondisi kemungkinan koneksi internet pada *smartphone*.

Hasil percobaan pengiriman teks tanpa gambar membutuhkan waktu rata-rata 5,626 detik untuk EDGE-HSDPA, 3,901 detik HSDPA- EDGE, 3,554 detik HSDPA-HSDPA, dan 6,921 detik untuk EDGE-EDGE.

Sedangkan untuk percobaan pengiriman teks dan gambar membutuhkan waktu rata-rata 5,731 detik EDGE-HSDPA, 5,369 detik HSDPA-EDGE, 4,415 detik HSDPA-HSDPA, dan 7,073 detik untuk EDGE-EDGE.

c. *Push Notification Google Cloud Messaging* (GCM)

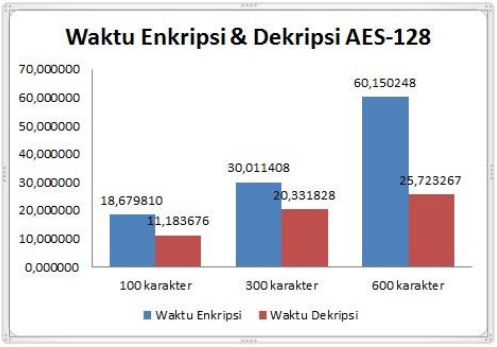
Pengujian GCM ditujukan untuk menguji seberapa cepat data sampai pada perangkat lain. Pengujian dicoba dengan empat kemungkinan koneksi internet, yakni : EDGE-HSDPA, HSDPA-EDGE, HSDPA–HSDPA dan EDGE–EDGE. Pengujian dilakukan sebanyak 30 kali percobaan

untuk fitur *invite* berdasarkan kondisi kemungkinan koneksi internet pada *smartphone*.

Dari hasil percobaan didapat hasil rata-rata 2,643 detik untuk EDGE-HSDPA, 2,582 detik HSDPA-EDGE, 1,956 detik HSDPA-HSDPA, dan 2,714 detik untuk EDGE-EDGE.

2. Pengujian Waktu Enkripsi dan Dekripsi

Pengujian dilakukan dengan mengukur waktu proses enkripsi ketika dikirim ke penerima sedangkan waktu proses dekripsi dilakukan ketika pesan diterima. Teknik pengukuran dilakukan dengan panjang kunci 128 bit dan dilakukan sebanyak 30 kali percobaan untuk masing-masing panjang karakter 100, 300, 600. Teknik pengukuran waktu proses enkripsi dan dekripsi adalah nanosecond / 1000000 hasilnya akan menjadi dalam milisecond.



Gambar 7 Waktu Enkripsi dan Dekripsi

Terlihat dari gambar di atas semakin banyak plainteks yang dimasukkan maka semakin lama waktu yang dibutuhkan untuk enkripsi dan dekripsi pesan. Nilai rata-rata enkripsi untuk plainteks 100 karakter 18,679810 ms sedangkan waktu dekripsi 11,183676 ms, 300 karakter waktu enkripsi 30,011408 ms dan dekripsi 20,331828 ms, 600 karakter waktu enkripsi 60,150248 ms dan dekripsi 25,723267 ms.

3. Pengujian Avalanche Effect

Pada kriptografi, hasil yang diberikan sangat unik, berbeda dari data yang menjadi masukan dari proses tersebut. Sedikit perubahan pada data masukan dapat memberikan perubahan yang signifikan pada hasil proses kriptografi, dan perubahan tersebut dinamakan *avalanche effect*. Semakin besar perubahan yang terjadi, semakin baik performansi dari algoritma kriptografi tersebut.

Avalanche Effect=Jumlah bit berbeda/total bit

Pada pengujian ini, dilakukan pengujian pengiriman pesan dan dilakukan percobaan sebanyak 30. Pesan yang dikirim adalah dengan menginputkan *plaintext* "*hai apa kabar*?".

Dari hasil pengujian, didapat banyak perubahan yang sangat besar pada bagian keluaran yaitu berkisar rata-rata sebesar 0,539069. Hasil ini menunjukkan bahwa algoritma AES-128 memiliki performa yang sangat baik. Suatu algoritma kriptografi akan sulit untuk dipecahkan ketika kunci yang digunakan tidak diketahui, dan hasil yang diberikan sangat unik.

4. Pengujian Waktu Proses

Pada potongan kode aplikasi proses enkripsi, didapat jumlah eksekusi kode (dengan asumsi n = in.length). Sehingga nilai kompleksitas adalah 13+3*n*, atau dapat dituliskan sebagai *O*(*n*).

Pada potongan kode aplikasi untuk dekripsi diasumsikan sama dengan proses enkripsi. Sehingga nilai kompleksitas adalah 13+3*n*, atau dapat dituliskan sebagai *O*(*n*) sama seperti pada proses enkripsi.

IV. Penutup

4.1 Kesimpulan

Dari pengujian yang telah dilakukan, diambil kesimpulan sebagai berikut.

1. Penggunaan memori *heap size* aplikasi *"Prototype Community Messenger"* dari total memori *heap size* yang disediakan oleh perangkat android adalah 68,882%.
2. Dari hasil percobaan pengiriman pesan atau data menunjukan bahwa, performa kecepatan pengiriman pesan ditentukan oleh *bandwith* internet yang digunakan oleh pengirim.
3. Waktu rata-rata enkripsi 100 karakter pada algoritma AES dengan kunci 128 bit adalah 18,679810 ms, sedangkan waktu dekripsi 11,183676 ms. 300 karakter waktu enkripsi 30,011408 ms dan dekripsi 20,331828 ms, 600 karakter waktu enkripsi 60,150248 ms dan dekripsi 25,723267 ms. Algoritma kriptografi yang digunakan memiliki performa yang baik, dilihat dari nilai *Avalanche effect* pada pengujian pertama dan kedua yang rata-rata bernilai 0,539069.

4.2 Saran

Dari aplikasi yang telah dibangun tentunya masih perlu pengembangan agar bisa lebih baik. Saran untuk melakukan pengembangan pada aplikasi ini adalah sebagai berikut.

1. Membuat sistem *Push notification* khusus pada *server* (tidak menggunakan *Google Cloud Messaging*).
2. Dikembangkan fitur-fitur tambahan seperti: *voice call*, *video call* dan *sending video*.

Daftar Pustaka

[1] Ariyus, Dony, 2008, *Pengantar Ilmu KRIPTOGRAFI (Teori, Analisis, dan Implementasi)*, Yogyakarta, ANDI OFFSET.

[2] Budi Raharjo, 2002, *Keamanan Sistem Informasi Berbasis Internet*, Bandung, PT Insan Infonesia, Jakarta, PT INDOCISC.

[3] Budiyono, Avon, 2004, *Enkripsi Data Kunci Simetris dengan Algoritma Kriptografi LOKI97*, Bandung: Institut Teknologi Bandung.

[4] Galbreath, Nick, 2002, *Cryptography for Internet and Database Applications (Developing Secret and Public Key Techniques with Java)*, Indiana, Wiley Publishing, Inc.

[5] Hakim S, Rachmad, dan Ir. Sutarto,M.Si, 2009,*Mastering Java*, Jakarta : PT. Gramedia.

[6] Kurniawan, Yusuf, 2004, *KRIPTOGRAFI (Keamanan Internet dan Jaringan Komunikasi)*, Bandung, Informatika.

[7] R. Stinson, Douglas, 2006, *Cryptography (Theory and Practice : Third Edition)*, Boca Raton, Taylor & Francis Group.

[8] Safaat H, Nazruddin, 2011, *Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android*, Bandung : Informatika.

[9] Siregar, Ivan Michael, 2011, *Membongkar Source Code Berbagai Aplikasi Android*, Yogyakarta:Gava Media.