PENGEMBANGAN APLIKASI PENGAMANAN *FILE* SEBAGAI SOLUSI KEAMANAN DATA PADA *SMARTPHONE* BERBASIS ANDROID

Yuri Prihantono[1)], Gusari Bagio[2)]
Lembaga Sandi Negara
Jl.Harsono RM No.70 Ragunan, Jaksel - 12550
Telp : (021) 7805814, Fax : (021) 78844104
E-mail : prihantono.yuri@gmail.com[1)] , gusari.bagio@lemsaneg.go.id[2)]

Abstrak

Meningkatnya penggunaan Handphone yang semula hanya untuk telepon dan kirim/terima SMS, kini beralih ke teknologi yang memiliki fitur yang lebih banyak dan lebih canggih terutama dalam hal koneksi internet. Salah satu Smartphone yang sangat banyak digunakan oleh masyarakat adalah Smartphone dengan Sistem Operasi Android. Smartphone dapat bertukar data file dengan pemilik Smartphone lain melalui media internet maupun fasilitas layanan yang ada di Smartphone, salah satu contohnya adalah dengan menggunakan fasilitas email publik maupun aplikasi penyimpanan file melalui media Cloud Storage (misalnya Dropbox, Drive, dll). Namun terdapat kerawanan pada layanan tersebut, yaitu adanya potensi pencurian data atau manipulasi data dari pihak yang tidak berkepentingan. Tujuan dari penulisan paper ini adalah mengembangkan aplikasi pengamanan file pada Smartphone berbasis Android dengan menggunakan algoritma simetrik AES dan meningkatkan jaminan integritas data dengan menerapkan integrity checksum menggunakan algoritma MD5.

Kata kunci: *kriptografi, pengamanan file, android, email publik*

Abstract

The increasing use of mobile phones were originally only for phones and send / receive SMS, are now turning to technology that has more features and more sophisticated, especially in terms of internet connection. One smartphones are widely used by the public is a Smartphone with Android Operating System. The owners of smartphones can exchange data files with the owners of other smartphones via the Internet or facilities that exist in the Smartphone, one example is to use the email facility public or application file storage through the media cloud storage (eg Dropbox, Drive, etc). But there are vulnerabilities in the service, namely the potential for data theft or manipulation of data from unauthorized parties. The purpose of this paper is to develop security applications on the Android-based smartphone files using AES symmetric algorithms and improve data integrity assurance to enforce the integrity checksum using the MD5 algorithm.

Keywords: *cryptography, file security, android, public email*

1. PENDAHULUAN

Saat ini, penggunaan *Smartphone* (telepon pintar) semakin banyak. Telepon seluler yang semula hanya digunakan untuk melakukan panggilan telepon dan kirim/terima SMS kini memiliki tambahan fitur dan lebih canggih dalam hal koneksi internet. Dengan fitur koneksi internet tersebut pengguna *smartphone* dapat melakukan *browsing* internet, bermain *game*, mengirim pesan melalui email, dan sebagainya [11]. Salah satu *Smartphone* yang sangat banyak digunakan oleh masyarakat adalah *Smartphone* dengan Sistem Operasi Android, sesuai dengan data statistik yang didapatkan dari *website* StatCounter [5].

Smartphone Android memiliki teknologi untuk menyimpan data-data dalam *memory* internal maupun *memory* eksternal. Dengan adanya fasilitas internet, pengguna dapat menyimpan data ke media penyimpanan *online* (*Cloud Storage*), saling bertukar data *file* melalui layanan email, dan sebagainya. Namun terdapat kerawanan pada layanan tersebut, yaitu adanya potensi pencurian data atau manipulasi data dari pihak yang tidak berkepentingan.

Penerapan Algoritma Kriptografi dapat digunakan untuk melakukan enkripsi data pada *file* yang disimpan baik dalam *memory* internal, *memory* eksternal, penyimpanan data melalui media penyimpanan *online*, maupun saat bertukar data *file* melalui layanan email. Teknik enkripsi tersebut diharapkan dapat menjadi salah satu solusi untuk menjaga keamanan dan kerahasiaan penyimpanan data pada *Smartphone* berbasis Android.

Pada penelitian ini, ada berbagai literatur yang menjadi landasan teori penulis dalam melakukan penelitian, literatur tersebut terdiri dari berbagai buku ilmu pengetahuan dan makalah-makalah yang sesuai dengan penelitian ini. Berikut ini adalah beberapa literatur utama yang menjadi landasan teori penulis:

1.1. Kriptografi

Kriptografi adalah seni dan ilmu menjaga keamanan pesan [15]. Kriptografi juga merupakan studi mengenai teknik matematika yang dihubungkan dengan aspek keamanan informasi seperti kerahasiaan, integritas data, otentikasi entitas, dan otentikasi keaslian data [13]. Tujuan kriptografi antara lain , kerahasiaan, integritas data, otentikasi, anti penyangkalan.

1.2. *Advanced Encryption Standard* (AES)

AES merupakan algoritma standar NIST Amerika tahun 2001. Algoritma AES merupakan algoritma *block cipher* yang merupakan fungsi enkripsi untuk blok-blok berukuran tetap [10]. Algoritma yang diadopsi dari Rijndael ini melakukan proses enkripsi/dekripsi dengan ukuran blok 128-bit dan kunci input bervariasi yaitu 128, 192 dan 256 bit. Algoritma yang digunakan dalam penelitian ini adalah AES 256 bit.

1.3. *Fungsi Hash*

Pada sistem komunikasi berbasis jaringan terbuka seperti internet sangat rawan terhadap berbagai serangan oleh pihak-pihak yang tidak berwenang, yang secara bebas dapat melakukan pemantauan, penyadapan, replay, modifikasi, pemalsuan maupun injeksi terhadap pesan yang ditransmisikan. Dengan adanya ancaman tersebut khususnya modifikasi atau injeksi oleh pihak lawan, maka jaminan integritas data sangatlah penting, selain kerahasiaan dari data itu sendiri. Untuk menjamin pesan atau data yang dikirim valid dan tidak mengalami modifikasi, pemalsuan atau injeksi selama transmisi, maka diperlukan suatu teknik otentikasi untuk menjamin kebenaran pesan tersebut. Teknik kriptografik yang sering digunakan untuk otentikasi data adalah fungsi hash [10]. Algoritma hash yang digunakan dalam penelitian ini adalah MD5 dan SHA256.

1.4. Android

Penelitian ini dikembangkan untuk Smartphone berbasis android dengan pertimbangan bahwa Smartphone Android merupakan Smartphone yang sangat popular saat ini dan pengembangannya terbuka (*open source*). Penelitian ini menghasilkan file berekstensi APK. File dengan ekstensi ini dapat didistribusikan sebagai aplikasi dan diinstall pada perangkat *mobile* Android [1].

1.5. *Software* Testing

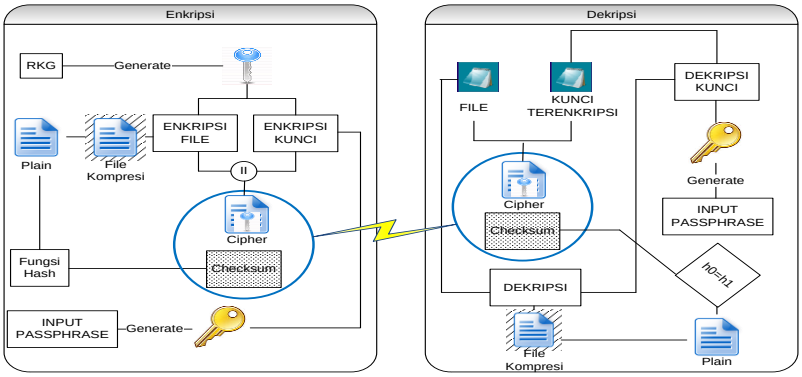
Persiapan untuk pengujian dimulai selama tahap analisis, dengan mengidentifikasi *test cases* menggunakan *robustness diagrams* [14]. Pada pengujian aplikasi pengamanan *file*, penulis membatasi hanya melakukan *acceptance testing*. *Acceptance testing* adalah Pengujian formal yang dilakukan oleh pelanggan, untuk menentukan apakah sistem sesuai dengan kebutuhan yang diinginkan.

2. PERANCANGAN DAN IMPLEMENTASI

2.1. Perancangan

Penerapan algoritma kriptografi dapat digunakan untuk melakukan enkripsi data pada file yang disimpan baik dalam memory internal, memory eksternal, penyimpanan data melalui media penyimpanan online, maupun saat bertukar data file melalui layanan email.

Pada pengembangan rancangan aplikasi enkripsi dan dekripsi data file, algoritma kriptografi yang dibutuhkan adalah AES256 dan fungsi *hash* MD5 dengan menggunakan *library* bouncycastle [4]. Berikut ini adalah skema enkripsi dan dekripsi file :



Gambar 1. Skema Enkripsi dan Deskripsi File

Sesuai dengan skema enkripsi dan dekripsi file pada gambar 2, ada beberapa teknik yang digunakan penulis dalam mengimplementasikan aplikasi tersebut, antara lain :

a) Proses Enkripsi
   1) Penulis menggunakan RKG (*Random Key Generator*) yang berupa inputan kunci acak untuk mengenkripsi file. Algoritma RKG yang digunakan adalah algoritma SecureRandom Java [7].
   2) Penulis menggunakan teknik kompresi data sebelum dienkripsi. Kompresi *file* bertujuan untuk mengurangi ukuran data tanpa merusak tujuan dari data tersebut [3]. Algoritma kompresi yang digunakan adalah algoritma *Deflate* [8].
   3) Penulis menggunakan fungsi hash MD5 untuk menghasilkan file *checksum* dari file *plain* (file sebelum di-enkripsi.
   4) Penulis menggunakan parameter *passphrase* untuk membangkitkan kunci dengan menggunakan Algoritma SHA-256 yang digunakan untuk mengenkripsi kunci hasil dari RKG dengan algoritma AES. Teknik tersebut merupakan teknik distribusi kunci menggunakan *Key-Encrypting-Key* [10].
   5) File hasil kompresi tersebut dienkripsi menggunakan algoritma AES dengan kunci yang dihasilkan dari RKG, kemudian menggabungkannya (*concate*) dengan kunci yang telah terenkripsi sesuai dengan nomor 4 menjadi satu file enkripsi.
   6) Penulis mengirimkan file enkripsi beserta checksum sesuai nomor 2 melalui Mail Client yang tersedia di aplikasi pengguna. Penulis menggunakan Android Intent [2] untuk memanggil aplikasi *mail client*. Intent yang digunakan adalah ACTION_SEND. Intent ini berfungsi untuk menjalankan/ memanggil activity untuk mengirim data dengan tipe mime [12]. Dengan Intent ini, pengguna dapat mengirim file enkripsi beserta checksumnya tanpa harus keluar dari aplikasi utama.

b) Proses Dekripsi

   Proses dekripsi merupakan kebalikan dari proses enkripsi yang bertujuan untuk mendapatkan file plain (file yang dapat dibaca). Teknik yang digunakan untuk mendapatkan file *plain* sebagai berikut :
   1) Penulis memisahkan file enkripsi dan kunci terenkripsi (truncate).
   2) Penulis men-dekripsi kunci terenkripsi tersebut dengan input parameter *passphrase* yang sama dengan inputan *passphrase* saat enkripsi.
   3) Penulis men-dekripsi file enkripsi dengan kunci hasil dekripsi pada nomor 2.
   4) Penulis melakukan teknik dekompresi untuk mendapatkan file plain.
   5) Penulis menggunakan *tools generate MD5 checksum* untuk menghasilkan *checksum* dari file *plain (output)*, kemudian membandingkan dengan checksum yang telah dikirim melalui *email*. Jika *checksum* sama, dapat dipastikan bahwa file tersebut dapat dijamin keasliannya. *Tools* dapat digunakan secara *online* dengan mengunjungi website *Online* MD5 [6] atau dapat diunduh di *website* http://www.toast442.org/md5/ [9].

Rancangan aplikasi pengamanan *file* tersebut diimplementasikan pada *Smartphone* Android. Adapun perangkat lunak dan perangkat keras yang digunakan dalam pengembangan rancangan aplikasi yang dibuat penulis adalah sebagai berikut :

Tabel 1. Kebutuhan Perangkat Lunak dan Perangkat Keras dalam Pengembangan Aplikasi

Perangkat Lunak	**Deskripsi**
Operating System	Windows 8.1 Single Language
Platform IDE	Eclipse with ADT
Development Kit	Android SDK

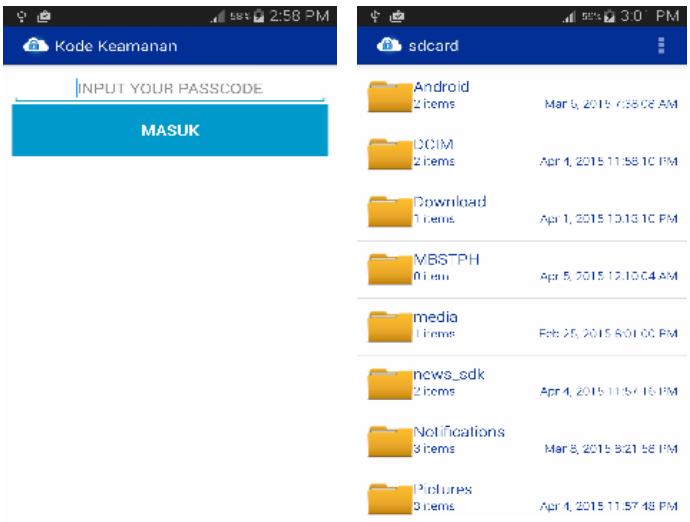
Perangkat Keras	Deskripsi
Laptop	Prosessor : Intel ® Core™ i5-4200U
	RAM : 4 GB
	Hardisk : 102 GB
Smartphone	Android versi 4.4.4 (Kitkat)
	CPU : Quad-core 1.3 GHz Cortex-A53 & Quad-core 1.9 GHz Cortex-A57
	Internal Memory : 32 GB, 3 GB RAM

2.2. Implementasi

Dalam menjalankan aplikasi pengamanan *file*, tahap-tahap yang dilakukan oleh pengguna aplikasi adalah sebagai berikut :

a. Instalasi

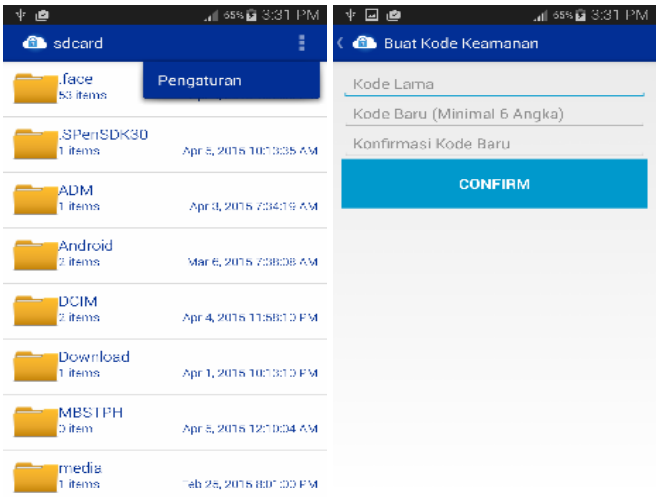
Instalasi aplikasi pengamanan *file* ke dalam *Smartphone* Android dengan membuka *file* aplikasi Afiction.apk. Setelah aplikasi ter-*install*, pengguna meng-*input* kode keamanan yang di-*set* secara *default* yaitu "123456". Setelah meng-input kode keamanan, pengguna dapat masuk ke tampilan utama aplikasi. Berikut adalah tampilan awal dan tampilan utama aplikasi :



Gambar 2. Skema Enkripsi dan Deskripsi File

b. Ubah Kode Keamanan

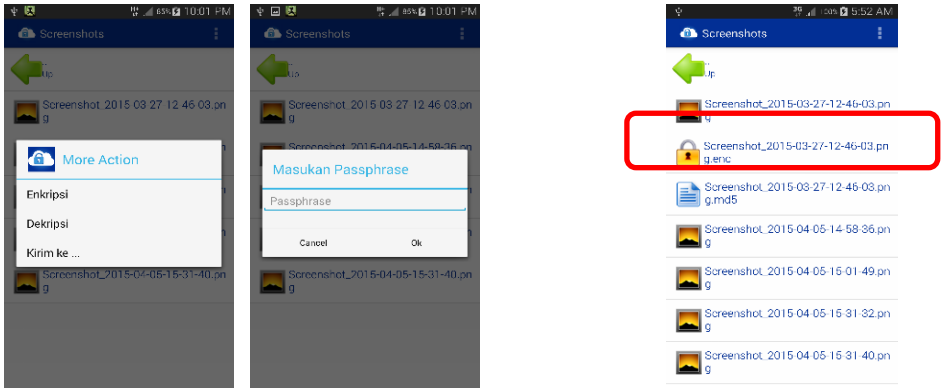
Pada tampilan utama, terdapat menu pengaturan yang digunakan untuk mengganti kode keamanan default ke kode keamanan yang diinginkan oleh pengguna. Penggantian kode keamanan ini dimaksudkan untuk meningkatkan keamanan aplikasi pengamanan *file*, sehingga hanya pengguna yang mengetahui kode keamanan tersebut yang dapat membuka aplikasi. Berikut ini adalah tampilan ubah kode keamanan :



Gambar 3. Tampilan Ubah Kode Keamanan Aplikasi

c. Enkripsi/Dekripsi File

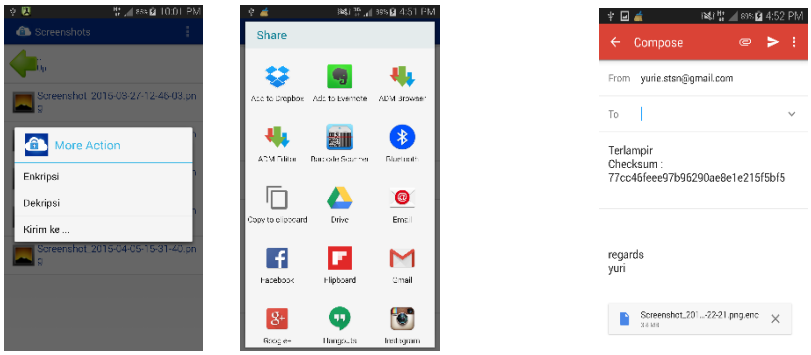
Enkripsi *file* dilakukan dengan cara menekan lama (hold) *file* yang akan dienkripsi. Setelah muncul pop-up menu, pilih menu enkripsi. Pada menu enkripsi, terdapat textbox untuk memasukkan passphrase. Passphrase tersebut digunakan untuk mengamankan kunci enkripsi. *File* tersebut dienkripsi dengan menggunakan algoritma AES256. Sebelum dienkripsi, *file* tersebut di-*hash* menggunakan algoritma MD5 untuk mendapatkan integrity *checksum*-nya. Setelah *file* dienkripsi, kemudian digabungkan dengan kunci yang dienkripsi dengan *passphrase* ke dalam satu *file* yang merupakan *file* output. Berikut adalah tampilan menu enkripsi/dekripsi :



Gambar 4. Tampilan Menu Enkripsi/Dekripsi dan Hasil Enkripsi

d. Kirim File Enkripsi

Aplikasi pengamanan *file* yang dikembangkan penulis mempunyai fitur untuk mengirimkan *file* terenkripsi ke layanan publik yang disediakan oleh *device* pengguna, misalnya email dan layanan penyimpanan *online* (Drive, Dropbox, dll). Pengguna dapat mengirimkan *file* dengan memilih menu "*Kirim Ke*". Berikut adalah tampilan untuk mengirim *file* terenkripsi beserta *checksum*-nya melalui email :



Gambar 5. Tampilan Mengirim File Melalui Email

3. PENGUJIAN APLIKASI

Pengujian aplikasi dilakukan dengan metode *acceptance testing* untuk menentukan apakah sistem telah berjalan sesuai dengan kebutuhan. Teknik yang digunakan dalam pengujian ini yaitu dengan menggunakan menggunakan masukan yang valid dan tidak valid, dan skenario penggunaan kedua data tersebut. Berikut ini adalah tabel *acceptance testing* aplikasi pengamanan *file*.

Tabel 2. Hasil *Acceptance Testing* Aplikasi Pengamanan *File*

Proses	Input	Kebutuhan	Berhasil/ Tidak
Pengaturan kode keamanan	*Passcode*	Aplikasi berhasil jika *passcode* yang dimasukkan valid, gagal jika *passcode* yang dimasukkan tidak valid	**Berhasil**
Enkripsi	*File*	Aplikasi berhasil jika *file* yang dienkripsi menghasilkan output berenkstensi .enc, gagal jika tidak menghasilkan *output* tersebut	**Berhasil**
Dekripsi	*File* *.enc	Aplikasi berhasil jika *file* yang dienkripsi menghasilkan *output file* terbaca, gagal jika tidak menghasilkan output tersebut	**Berhasil**

Kirim *File*	*File* *.enc	Aplikasi berhasil jika pengguna dapat memilih layanan untuk mengirimkan *file* terenkripsi, gagal jika pengguna tidak dapat memilih layanan untuk mengirim *file* tersebut	**Berhasil**

Dari hasil pengujian di atas, maka dapat disimpulkan bahwa secara umum aplikasi pengamanan *file* yang dikembangkan telah berjalan sesuai dengan kebutuhan.

Selain *acceptance testing,* penulis juga melakukan uji performa berdasarkan lama waktu proses yang dilakukan dalam aplikasi. Pengujian dilakukan terhadap 5 (lima) *file* dengan tipe *file* dan ukuran yang berbeda, dienkripsi dengan kunci yang sama untuk semua *file* tersebut. Tabel 3 menunjukkan hasil pengujian performa enkripsi dan dekripsi *file* pada aplikasi pengamanan *file*.

Tabel 3. Hasil Pengujian Performa Aplikasi

Tipe *File*	**Ukuran *File* Asli (byte)**	**Ukuran *File* Terkompresi (byte)**	**Waktu (ms)**	
			Enkripsi	**Dekripsi**
PNG	2347110	2347831	14974	15114
PDF	242096	215304	1521	1445
DOC	304128	254524	1753	1688
TXT	10398	1375	52	28
MP3	4421610	4396262	34664	27603

Dari hasil pengujian di atas, maka dapat disimpulkan bahwa penerapan algoritma enkripsi AES kurang efisien untuk melakukan enkripsi dan dekripsi *file* berukuran besar (ukuran *file* : 4 mb ke atas).

4. SIMPULAN DAN SARAN

Berdasarkan data hasil penelitian dan analisis data, dapat ditarik simpulan dan saran, sebagai berikut :

4.1. Simpulan

a. Aplikasi Pengamanan File ini menerapkan algoritma kriptografi AES untuk enkripsi *file* dan fungsi hash MD5 untuk menghasilkan *file checksum*.
b. Proses pembangkitan kunci enkripsi secara acak untuk setiap *file*, kunci enkripsi diamankan dengan teknik distribusi kunci *Key-Encrypting-Key* menggunakan parameter *passphrase*, dan digabung menjadi satu *file* dengan *file* terenkripsi, dengan tujuan agar setiap *file* memiliki kunci enkripsi yang berbeda dan hanya yang memiliki aplikasi dengan metode yang sama *yang* dapat membuka *file* terenkripsi tersebut.
c. Aplikasi Pengamanan File ini menggunakan kompresi *Deflate* dengan tujuan untuk mengurangi ukuran data sebelum dienkripsi.
d. Aplikasi Pengamanan File ini memiliki fitur mengirimkan *file* terenkripsi via *mail client*.
e. Aplikasi ini memiliki fitur keamanan aplikasi menggunakan PIN/*Passcode*. *Passcode* tersebut dimasukkan setiap aplikasi ini dijalankan dengan tujuan agar aplikasi ini hanya dapat dijalankan oleh orang yang memiliki akses.

4.2. Saran

a. Aplikasi ini memiliki kekurangan dalam hal pengamanan kunci yang hanya menggunakan pertukaran *passphrase* secara offline. Penulis menyarankan penggunaan Kriptografi kunci publik untuk pengamanan pertukaran *passphrase*.
b. Penulis hanya memiliki studi literatur mengenai penggunaan MD5 untuk *checksum*. Pada pengembangan selanjutnya, penulis menyarankan untuk melakukan penelitian mengenai algoritma *checksum* selain MD5.

5. DAFTAR RUJUKAN

[1] Afandi, Moch Ikhsan, 2014.*Rancang Bangun Aplikasi FileCryptor untuk Enkripsi File dengan Metode AES-128*. Universitas Islam Negeri Sultan Syarif Kasim Riau : Pekanbaru.
[2] Android Developer, 2016. Intent. (https://developer.android.com/reference/android/content/Intent.html)

[3] Kurniawan Saputra, Aan, dkk, 2015. Aplikasi Kompresi File Citra Menggunakan Algoritma Arithmetic Coding Berbasis Java. Kendari : Universitas Halu Oleo.
[4] Legion of The Bouncy Castle Inc, 2013. *SIGNED JAR FILES*[online]. Available at : (http://bouncycastle.org/latest_releases.html) (Accessed 18 Juli 2016).
[5] StatCounter Global Stats, 2016. *Top 8 Mobile & Tablet Operating System form May 2015 to May 2016*[online]. Available at : (http://gs.statcounter.com/#mobile+tablet-os-ww-monthly-201505-201605) (Accessed 18 Juli 2016).
[6] Online MD5, 2016. *MD5 Hash Generator* [online]. Available at : (http://onlinemd5.com/) (Accessed 18 Juli 2016).
[7] Oracle, 2016. Java SE Documentation. (http://www.oracle.com/technetwork/java/javase/documentation/index.html)
[8] Ralf Quebbermann, 2013. *Howto compress and uncompress a Java byte array using JDK Deflater/Inflater* [online]. Available at : (http://qupera.blogspot.com/2013/02/howto-compress-and-uncompress-java-byte.html) (Accessed 18 Juli 2016).
[9] Graphical MD5Sum. Graphical MD5Sum [online]. Available at : (http://www.toast442.org/md5/) (Accessed 18 Juli 2016).
[10] Lembaga Sandi Negara, 2007. Jelajah Kriptologi. Jakarta : Lembaga Sandi Negara.
[11] Maskes, Three & Rahman, Abdul, 2013. *Implementasi Algoritma Serpent untuk Enkripsi dan Dekripsi Data File pada Ponsel Berbasis Android*[online]. Palembang : STMIK MDP.
[12] Meier, Reto, 1997.*Professional Android™ Application Development*. Indianapolis : Wiley Publishing, Inc, 2009.
[13] Menezes, J. Alfred, Van Ooschot, C. Paul, Vanstone dan A.Scott A, 1997. *Handbook of Applied Cryptography*, Boca Raton: CRC press LLC.
[14] Rosenberg, Doug & Stephens, Matt, 2007. *Use Case Driven Object Modeling with UML Theory and Practice*. USA : Doug Rosenberg and Matt Stephens.
[15] Schneier, Bruce, 1996. *Applied Cryptography, Second Edition : Protocols, Algorithm, and Source Code in C*. Oak Park : John Wiley & Sons, Inc.

Halaman ini sengaja dikosongkan