PEMBUATAN APLIKASI SMS KRIPTOGRAFI RSA DENGAN ANDROID

I.Joko Dewanto[1], Verdy Yanto[1]
[1]Program Studi Teknik Informatika Universitas Esa Unggul , Jakarta
Jln. Arjuna Utara Tol Tomang-Kebon Jeruk Jakarta
djoko.dewanto@esaunggul.ac.id

Abstract

Sending a message is an activity who use by everyone for a day now. But we a citizen must beware of message spy without permission or tap a message content while a message inside a process sending from cellular owner who has that message. Of course cellular owner doesn't know there's a other who know that a message content that is a private message or steal an idea content of message. For thatim initiative for build a security message using cryptograph with RSA Method which a cellular owner can sending or receive a message without spy by other (cryptonalyst). For me cryptograph is the only way for secure a message from cryptonalyst (hacker) and a cryptography has been called like a code language. And why I am implement RSA Method for a securing a message than an other method of cryptograph ? That thing because a common method like blowfish , chipper cigne , etc only use one key for encrypt and decrypt a message , but with a RSA Method a message will be encrypt and decrypt with two key and its call public key for encrypt a message and private key for decrypt a message. And this application will be build in android platform because there's many people have an android mobile.

Keywords: *cryptograph , RSA , message*

Pendahuluan

SMS (*Short Message*) mungkin sudah tidak asing lagi dimata masyarakat, banyak sekali orang menggunakan fitur SMS untuk berinteraksi dengan orang serta alternatif lain jika selular orang yang ingin dituju dalam keadaan *off* dan saat orang mengaktifkan selular mereka akan ada notifikasi pesan yang masuk. SMS sendiri dari tahun ke tahun selalu berkembang dan semakin lebih mudah untuk digunakan oleh para *user*. Banyak yang telah menggunakan fitur SMS dalam kehidupan sehari hari, tetapi seiring berkembangnya waktu proses keamanan dalam melakukan pengiriman data pun semakin rawan dikarenakan banyaknya pihak ketiga seperti *hacker* dapat mengintip atau melihat pesan yang bersifat penting ataupun tidak ,sebelum sampai ke orang yang dituju pesannya. Sudah banyak cara yang telah dilakukan oleh*progammer* yang membuat fitur SMS untuk mencegah jebolnya pesan dan data yang dikirim dari tangan *hacker*. Dan cara yang tepat adalah menggunakan **Ilmu Kriptografi**. Ilmu Kriptografi merupakan ilmu yang dikenal sebagai bahasa persandian sehingga pesan atau kalimat yang dikirimkan menjadi tersandikan dan tidak mampu untuk dibaca oleh para *hacker*.Untuk Istilah*hacker* dalam kriptografi dapat juga disebut sebagai *cryptonalyst*. Metode kriptografi pun sangat banyak untuk diterapkan dan masing masing dari metode mempunyai kelemahannya masing – masing.

Dalam penerapan kriptografi saat melakukan pengiriman pesan ataupun

penerimaan pesan,Aplikasiini menerapkan metode kriptografi RSA yang ditemukan pada tahun 1976 oleh Peneliti MIT (*Massachussets Institute of Technology*) oleh Ron (**R**)ivest , Adi (**S**)hamir dan Leonard (**A**)dleman. Mengapa penulis menerapkan metode kriptografi RSA dibanding metode kriptografi yang lain seperti *blowfish , AES , TwoFish , chiper cignere* dll? Hal ini dikarenakan karena metode RSA menggunakan 2 kunci untuk melakukan proses persandian data yang dimana kunci pertama *(public key)* yang digunakan untuk melakukan persandian dan kunci kedua *(private key)* yang digunakan untuk menterjemahkan bahasa yang sudah disandikan menjadi bahasa yang dapat dibaca oleh manusia.Penelitian ini meneliti metode yang aman dalam menjaga keamanan pesan dari serangan *cryptonalyst.* Untuk fitur yang digunakan dalam mengimplementasikan aplikasi SMS dengan metode kriptografi RSA ini , dengan menggunakan *platfform* berbasis *android* ya ng dikarenakan untuk zaman era sekarang dari tahun 2012 dan mungkin sampai tahun 2014 *android* masih akan terus berjaya di kehidupan masyrakat maupun teknologi karena sistemnya yang bersifat ringan , multifungsi dan mudah digunakan karena sifat ponsel yang berifat layar sentuh *(touch-screen).*Untuk itu dibuatlah perancangan alur proses aplikasi SMS Kriptografi RSA ini dengan menggunakan *android.*

Landasan Teori

Algoritma RSA

Menurut (Stalling , 1995) dari sekian banyak metode kriptografi asimetris yang pernah dibuat, algoritma yang paling popular adalah algoritma RSA. Algoritma RSAdiciptakan oleh 3 orang peneliti dari *MIT* (*Masschussets Institute of Technology*) pada tahun 1976, yaitu **Ron (R)ivest, Adi (S)hamir dan Leonard (A)dleman.** Keamanan algoritma *RSA* terletak pada sulitnya menfaktorkan bilangan yang besar menjadi faktor – faktor prima.Penfaktoran dilakukan untuk memperoleh kunci *private*.Selama penfaktoran bilangan besar menjadi bilangan prima belum tentu menemukan algoritma yang benar, maka selama itu pula keamanan algoritma RSAterjamin.

Algoritma RSA memiliki besaran seperti berikut :

1. p dan q bilangan prima (rahasia)
2. n = p.q (tidak rahasia)
3. Φ(n) = (p-1)(q-1) (rahasia)
4. e = kunci enkripsi (tidak rahasia)
5. d = kunci dekripsi (rahasia)
6. m = *plaintext* (rahasia)
7. c = chipertext (tidak rahasia)

Algoritma Pembangkitan Kunci RSA

Berikut cara algoritma pembangkitan pasangan kunci adalah sebagai berikut :

1. Pilih dua bilangan prima sembarang, p dan q secara acak. $p \neq q$. Bilangan ini harus cukup besar (minimal 100 digit).
2. Hitung n = p . q (sebaiknya $p \neq q$, sebab jika p = q maka $n = p^2$sehingga, p dapat diperoleh dengan menarik akar pangkat dua dari n).
3. Hitung Φ(n) = (p - 1)(q - 1)
4. Pilih kunci *public key* , e yang relatif prima dengan Φ(n).
5. Bangkitkan kunci *private* dengan menggunakan persamaan (V), yaitu e . d = 1 (mod Φ(n)) ekivalen dengan e . d = k Φ(n)+1 , sehingga secara sederhana d dapat dihitung dengan

$$d = \frac{1 + k\Phi(n)}{e}$$

Hasil dari algoritma di atas :

- Kunci *public*adalah pasangan (e,n)
- Kunci *private* adalah pasangan (d,n)

Catatan :n tidak bersifat rahasia, sebab ia diperlukan pada perhitungan enkripsi / dekripsi.

Contoh :

Misalkan Alice akan membangkitkan kunci *public* dan kunci *private* miliknya. Alice memiliki p = 27 danq = 71 (dalam prkateknya, kedua bilangan ini haruslah bilangan besar).Selanjutnya Alice menghitung

$$n = p \cdot q = 47 \cdot 71 = 3337$$

dan

$$\Phi(n) = (p - 1)(q - 1) = 3220$$

Alice memilih *public* e = 79, karena 79 relatif prima dengan 3220. Alice mengumumkan nilai e dan n kepada *public*, selanjutnya Alice menghitung kunci dekripsi d seperti dituliskan pada langkah intruksi 5 dengan menggunakan persamaan (XI)

$$d = \frac{1 + k\,\Phi(n)}{e}$$

$$d = \frac{1 + (k \times 3220)}{79}$$

Dengan mencoba nilai – nilai k = 1, 2, 3,diperoleh nilai d yang bulat adalah 1019. Ini adalah kunci *private* untuk mendekripsikan pesan. Kunci *private* ini harus dirahasiakan oleh Alice. Jadi perhitungan kunci ini menghasilkan pasangan kunci *private* dan *public* :

- Kunci *public :* (e = 79 , n = 3337)
- Kunci *private:* (d=1019, n = 3337)

Algoritma Enkripsi / Dekripsi Enkripsi

1. Ambil kunci *public* menerima pesan, e , dan modulus n.
2. Nyatakan *plaintext* m menjadi blok – blok m_1, m_2 , …sedemikian sehingga setiap blok mempresentasikan nilai di dalam selang [0, n - 1].
3. Setiap blok m dienkripsi menjadi blok c_i dengan rumus $c_i = m_i^e \bmod n$.

Deskripsi

Setiap blok *chipertext* c_i didekripsikan kembali menjadi blok $m_i = c_i^d \bmod n$

Contoh :

Misalkan Bob mengirim pesan kepada Alice.Pesan(*plaintext*)yang akan dikirim ke Alice adalah

m = HARI INI

atau dalam sistem pengkodean ASCII adalah

7265827332737873.

Bob memecah m menjadi blok yang lebih kecil, misalnya m dipecah menjadi enam blok yang berukuran 3 digit :

m_1 =726 m_4 = 273

m_2= 582 m_5= 787

m_3=733 m_6= 003

Nilai – nilai m_i ini masih terletak dalam selang [0, 3337 - 1] agar transformasi menjadi satu ke satu. Bob mengetahui kunci *public* Alice adalah e = 79 dan n = 3337. Alice dapat mengenkripsikan setiap blok *plaintext* sebagai berikut :

- $C_1 = 726^{79} \bmod 3337 = 215$;
- $C_2 = 589^{79} \bmod 3337 = 776$;
- $C_3 = 733^{79} \bmod 3337 = 1743$;
- $C_4 = 273^{79} \bmod 3337 = 933$;
- $C_5 = 787^{79} \bmod 3337 = 1731$;
- $C_6 = 003^{79} \bmod 3337 = 158$;

Jadi *chipertext* yang dihasilkan adalah

c = 215 776 1743 933 1731 158

Dekripsi dilakukan dengan menggunakan kunci *private*

d = 1019

Blok – blok *chipertext* didekripsikan sebagai berikut :

1. $m_1 = 215^{1019} \bmod 3337 = 726$
2. $m_2 = 776^{1019} \bmod 3337 = 589$
3. $m_3 = 1743^{1019} \bmod 3337 = 733$
4. $m_4 = 933^{1019} \bmod 3337 = 273$
5. $m_5 = 1731^{1019} \bmod 3337 = 787$
6. $m_6 = 158^{1019} \bmod 3337 = 003$

Blok *chippertext* diubah menjadi seperti semula menjadi *plaintext* dan mendapat hasil

m = 7265827332737873.

Yang dalam pengkodean ASCII menjadi

m = HARI INI

Metode Penelitian

Cara Kerja RSA

Proses pembuatan kunci dibuat dalam jumlah dua kunci yang berbeda dimana kunci pertama disebut seagai *public key* yang berfungsi menyandikan pesan / data sebelum dikirim dan kunci kedua adalah *private key* yang berfungsi sebagai menterjemahkan pesan yang sudah disandikan menjadi pesan yang dapat dibaca oleh manusia.

Proses pembuatan dua buah kunci ini digunakan dengan operasi pendaktoran FPB (bilangan pembagi terbesar) yang dimana dua buah kunci dicetak dalam format numerik / angka. Semakin panjang bit dalam proses RSA , semakin panjang pula panjang dua buah kunci.

Metode Pengembangan Aplikasi

Metode pengembangan yang diambil dalam aplikasi ini adalah *prototype* yang dikarenakan aplikasi ini merupakan aplikasi skala kecil , bergantung pada *user* , dapat ditambahkan fitur – fitur aplikasi dan berfokus untuk meningkatkan kepuasan pengguna aplikasi ini dalam mengirim pesan.

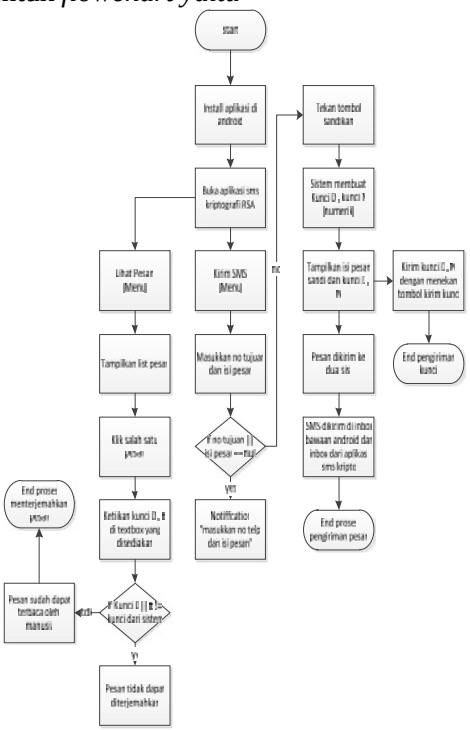
Pengumpulan Kebutuhan Aplikasi

Proses kebutuhan aplikasi ini meliputi , pertama – tama pengguna1 dan pengguna2 akan menginstalasi aplikasi SMS Kriptografi ini. Setelah itu pengguna 1 membuka aplikasi tersebut dan memilih menu “KIRIM SMS”, Pengguna 1 menginputkan pesan dan nomor tujuan dari pengguna2 dengan menekan tombol “SANDIKAN”, Saat ditekan , maka sistem otomatis akan menampilkan isi pesan yang sudah disandikan serta memperlihatkan dua buah kunci yaitu Kunci D dan Kunci N, dan pengguna1 mengirimkan dua buah kunci dengan tombol “KIRIM KUNCI”.

Pesan yang dikirim akan dikirim ke dua lokasi yaitu” LIST PESAN” pada aplikasi dan ke inbox sms pada selular android. Pesan yang dikirim akan berwujud pesan yang sudah disandikan.

Pengguna2 akan melihat isi pesan dari dua sisi , yaitu yang pertama pesan ditampilkan di *inbox* sms pada android yang pastinya isi pessan tidak dapat dimengerti oleh pengguna2 , dan yang kedua adalah pengguna2 melihat pesan di aplikasi SMS Kriptogrfi tersebut dengan memilih menu “LIHAT PESAN” dan melihat isi pesan serta menginputkan Kunci E dan Kunci N untuk menterjemahkan pesan yang disandikan menjadi pesan yang dapat dibaca oleh manusia.

Proses jalannya aplikasi dalam bentuk *flowchart yaitu*



Gambar 1
Proses Aplikasi SMS Kripografi RSA

Evaluasi Prototype

Tahap ini menjelaskan apakah kebutuhan aplikasi SMS Kriptografi RSA ini sudah sesuai dengan keinginan pengguna :

Tabel 1
Evaluasi *Prototype*

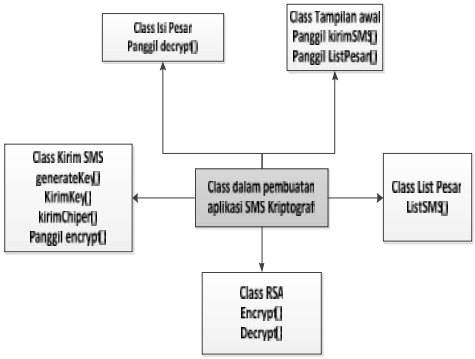
Kebutuhan pengguna android	Sesuai kebutuhan
Proses pengiriman sms dengan menggunakan dua sisi (inbox sms dari bawaan android dan inbox dari aplikasi)	Sesuai
Proses cara kerja metode rsa dalam mengirim atau menerima pesan yang efektif dan kuat pertahanan terhadap *hacker*	Sesuai
Proses pembuatan kunci d , kunci e , dan kunci n sudah cukup aman dalam mengirimkan pesan	Sesuai
Aplikasi mudah digunakan oleh pengguna	Sesuai

Saat Tahap ini sudah disetujui maka akan dilanjutkan ke tahap pengkodean sistem .

Tahap Pengkodean Sisten

Dalam tahap pengkodean, saya menggunakan perangkat lunak *eclipse Galileo* untuk membangun aplikasi berbasis *android* serta *android SDK* dan *android ADT 1.8.0* sebagai *emulator android.*Dengan tipe v*ersion* ROM android yaitu dari 2.3.1 ke atas.

Berikut Alur *flowchart* pembuatan fun*ction* dan *class* yang akan digunakan



Gambar 2
flowchart function dan class

Tahap Pengujian Sistem

Tahap pengujian sistem dilakukan pertama kali di emulator android yang sudah disediakan dari *eclipse Galileo* dan setelah itu dibuat file apk dari aplikasi dan diimplementasikan ke selular android apakah berjalan sesuai dengan yang ada di emulator.

Analisis Dan Pembahasan

Pembahasan Aplikasi SMS Kriptografi dengan menerapkan metode RSA pada Android

Adapun aplikasi yang saya buat dibuat di perangkat lunak *Eclipse Galileo* dengan spesifikasi komputer antara lain ; logo laptop adalah axio dengan *windows 7* 64bit dan RAM 1 GB. Rincian dari aplikasi ini meliputi :

1. Nama Aplikasi : Skipsi_SMS
2. Android_version : 2.3 ke atas dapat menggunakan aplikasi ini.
3. Jumlah emulator yang digunakan dalam tahap ini adalah dua emulator yang meliputi emulator pertama dengan nomor 5554 dan emulatoe kedua dengan nomor 5556
4. Nama Package : com.willis.skripsiku
5. Aplikasi diuji di selular android Samsung Galaxy Young

Dalam proses aplikasi ini terdapat tiga proses , meliputi :

1. Proses mengirimkan pesan sandi dan kunci
2. Proses pembuatan *inbox* yang dibuat dalam bentuk *list* atau *array*
3. Proses melakukan persandian pesan dan menterjemahkan pesan

4.1.1 Proses Pengiriman SMS

Dalam proses pengiriman proses SMS pada aplikasi ini terletak pada *class Kirim_SMS()* yang kode programnya meliputi :

```
publicvoidkirimSMS(String noTelp , String pesannya)
{
        String sent    = "SMS_SENT";
        String deliver = "SMS_DELIVERED";

PendingIntent sentPI = PendingIntent.getBroadcast(this, 0, new Intent(sent), 0);

PendingIntent deliveredPI = PendingIntent.getBroadcast(this, 0, new Intent(deliver), 0);

//ketikasmsdikirim
registerReceiver(new BroadcastReceiver()
{

publicvoid onReceive(Context context, Intent intent) {
switch (getResultCode()) {
        case Activity.RESULT_OK:
        Toast.makeText(getBaseContext(), "SMS SUDAH DIKIRIM", Toast.LENGTH_LONG).show();
break;
case android.telephony.SmsManager.RESULT_ERROR_GENERIC_FAILURE:

Toast.makeText(getBaseContext(), "Error nih", Toast.LENGTH_LONG).show();
break;
case android.telephony.SmsManager.RESULT_ERROR_NO_SERVICE:
Toast.makeText(getBaseContext(), "NO sinyal", Toast.LENGTH_LONG).show();
break;

case android.telephony.SmsManager.RESULT_ERROR_NULL_PDU:
Toast.makeText(getBaseContext(), "Null PDu", Toast.LENGTH_LONG).show();
break;

case android.telephony.SmsManager.RESULT_ERROR_RADIO_OFF:
Toast.makeText(getBaseContext(), "Handphone anda mati", Toast.LENGTH_LONG).show();
break;

default:
Toast.makeText(getBaseContext(),"Other Error", Toast.LENGTH_LONG).show();

}
}
},new IntentFilter(sent));

//ketikasmsditerima
registerReceiver(new BroadcastReceiver()
{

publicvoid onReceive(Context context, Intent intent) {
switch (getResultCode()) {
case Activity.RESULT_OK:
Toast.makeText(getBaseContext(), "SMS SUDAH DIKIRIM",Toast.LENGTH_LONG).show();
break;
```

```
case Activity.RESULT_CANCELED:

        Toast.makeText(getBaseContext(),
"SMS GGAL",
Toast.LENGTH_LONG).show();
break;

default:
Toast.makeText(getBaseContext(), "wadu
saya ga tau",
Toast.LENGTH_LONG).show();
}
}
},new IntentFilter(deliver));

android.telephony.SmsManager sms =
android.telephony.SmsManager.getDefault(
);

sms.sendTextMessage(noTelp, null,
pesannya, sentPI, deliveredPI);
}
```

4.1.2 Proses Enkripsi dan Dekripsi SMS

Proses kriptografi pada aplikasi ini terletak pada *class* yang bernama *class* RSA() dan berikut kode program pada isi dari *class RSA()* tersebut

```
publicclass RSAA {
```

Variabel

```
        private BigInteger n, d, e , p , q;
        private BigInteger phi;
        privateintbitlength = 30;
privateintblocksize = 120;
        privateintbitlen = 256;

        public RSAA(BigInteger newn,
BigInteger newe) {
        n = newn;
        e = newe;
          }

public RSAA(int bits) {
Random r = newRandom();
p = BigInteger.probablePrime(bitlength, r);
q = BigInteger.probablePrime(bitlength, r);
n = p.multiply(q);
```

Buat kunci e,n,d

```
phi =
p.subtract(BigInteger.ONE).multiply(q.subt
ract(BigInteger.ONE));
e = BigInteger.probablePrime(bitlength/2,
r);

while
(phi.gcd(e).compareTo(BigInteger.ONE) >
0 &&e.compareTo(phi) < 0 ) {
e.add(BigInteger.ONE);
  }
d = e.modInverse(phi);
                }


publicsynchronized BigInteger
encrypt(BigInteger message , BigInteger E ,
BigInteger N  )
 {
return message.modPow(E, N);
        }
```

Enkrip dan dekrip

```
        }

publicsynchronized BigInteger
decrypt(BigInteger message , BigInteger D ,
BigInteger N)
 {
return message.modPow(D, N);
  }


publicsynchronized BigInteger getN() {
returnn;
  }

/** memanggil kunci supaya dapat
ditampilkan */
publicsynchronized BigInteger getE() {
returne;
  }

publicsynchronized BigInteger getD() {
returnd;        }}
```

Proses enkripsi nya dilakukan dengan fungsi parameter , dan parameter yang

digunakan adalah **parameter BigInteger message , parameter BigInteger E dan parameter BigInteger N**
BigInteger Message merupakan parameter isi pesan yang akan dikirim , jadi untuk mengirimkan pesan yang bertipe text , mula – mula **text harus diubah tipe datanya menjadi BigInterger dalam proses enkripsi dan BigInteger dikembalikan lagi ke text untuk proses dekripsi.**
Sintak yang sangat penting dalam proses enkripsi dan dekripsi pada aplikasi ini adalah
return message.modPow(E, N);
//**enkripsi**
return message.modPow(D, N); //**dekripsi**
Penggunaan ModPow merupakan operasi matematka yang sangat berperan penting dalam **metode kriptografi RSA.**

Implementasi Aplikasi SMS RSA

Pada tahap implementasi ini masih menggunakan emulator android , dan pada Samsung galaxy y dan galaxy tab.
Contoh :
Pengguna a dengan no telp 5554 ingin mengrimkan pesan rahasia ke pengguna B dengan no telp 5556. Berikut Tahap Implementasi dari aplikasi SMS Kriptografi ini:

1. Pengguna 1 membuka aplikasi sms kriptografi RSA dan pilih menu BUAT SMS

Gambar 3
Tampilan Awal Aplikasi

2. Ketik no tujuan pengguna 2 (5556) dan isi pesan (“Hi”) dan klik tombol sandikan , Dan Klik Tombol Kirim SMS untuk mengirim kunci .

Gambar 4
Tampilan Kirim SMS

Dapat dilhat bahwa pesan yang ingin dikirim adalah (“Hi”)
Kunci E = 35879
Kunci D = 525591732184681191
Kunci N = 1126371150077146231
Pesan yang disandikan = 50576878686821942

3. Pengguna 2 akan menerima pesan dari pengguna 1 dalam 2 sisi yaitu :

Gambar 5
Tampilan *List Inbox*

Sisi pertama ditampilkan di list inbox bawaan android dan sisi kedua adalah

list pesan ditampilkan di list pesan pada aplikasi sms kriptografi RSA

4. Pengguna dua klik pesan yang ingin didekripsikan dan pengguna dua harus mengisi kunci D dan N yang sudah dikirimkan oleh pengguna 1

Gambar 6
Tampilan Isi Pesan

Jika kunci yang diinputkan benar maka pesan asli dapat muncul di bawah button trejemahan dengan bentuk label. Jika salah satu kunci atau dua kunci yang diinput salah , maka akan menghasilkan

Gambar 7
Tampilan Isi Pesan 2

Pesan yang tampil pada label adalah symbol yang tidak dapat dibaca oleh manusia.

Kesimpulan

Berdasarkan hasil dari pembuatan aplikasi SMS kriptografi dengan menerapkan metode RSA pada android , maka ddidapatkan kesimpulan seperti : (1) Dengan aplikasi ini , dapat memudahkan pengguna dalam melakukan proses pengiriman dan penerimaan pesan yang bersifat sangat penting dan rahasia; (2) Dapat memahami dari ilmu kriptografi dan metode – metode yang ada dalam kriptografi terutama Metode RSA; (3) Membuktikan bahwa metode RSA tidak hanya cuma digunakan untuk mengamankan data dan *digital signature* , tetapi metode ini dapat diterapkan untuk proses pengiriman pesan dan penerimaan pesan berbasis SMS; (4) Aplikasi ini dapat digunakan oleh siapa saja dan kapasitas *memory* tang dibutuhkan pun sangat kecil , hanya 30 kb saja; (5) Dengan panjang bit yang digunakan hanya 256 bit dapat melakukan proses persandian yang cukup aman.

Daftar Pustaka

Android Developers, *Android Documentation*, Download 19 November 2009,http://developer.Android.com/guide/topics/data/data-storage.html, 22 November 2012.

Bruce Schneier, *Applied Cryptography : Protocols, Algorithms, and Source Code in C*, USA, John Wiley & Sons, Inc., USA 1996

Christof Paar,*"APPLIED CRYPTOGRAPHY AND DATA SECURITY"*,www.crypto.rub.de,version 2. 5 January 2005.

Google IO, *Android Anatomy and* Physiology, Download 26 Oktober 2009, http://sites.google.com/site/io/anato

my--physiology-of-an-Android, 22November 2012

Menezes, A. dkk.,*"Handbook of Applied Cryptography"*, CRC Press, Inc.1996.

Munir, Rinaldi. Bahan Kuliah IF5054 Kriptografi. Departemen Teknik Informatika, Institut Teknologi Bandung. 2004

Rhee, Man Young, *Cryptography and Sesure Communications*, Singapore, McGraw-Hill Book Co., Singapore. 1994

Roger S Pressman, "*SOFTWARE ENGINERING Sixth Edition*", , McGrawHill, USA 2005

Rosidi, R., I. *Membuat Sendiri SMS Gateway (ESME) Berbasis Protokol SMPP*. ANDI. Yogyakarta. 2004

Sayed Y. Hashimi and Satya Komatineni, *Pro Android*, Apress Inc. 2009

William Stallings, *Network and Internetwork Security Principles and Practice*, Prectice-Hall, New Jersey. 1995.