MEMBANGUN APLIKASI KEAMANAN DATA TEKS DENGAN METODE RSA – CRT BERBASIS ANDROID

Herix Saputra Budihani

Abstrak

Keamanan data merupakan sesuatu yang harus diperhatikan dalam kemajuan teknologi informasi, terutama dalam pesan yang berbentuk teks. Sistem keamanan pesan yang berbentuk teks salah satunya dengan sistem kriptografi. Sistem kriptografi adalah ilmu yang menggunakan teknik matematika sebagai landasan untuk membuat sebuah sistem keamanan informasi. Metode yang digunakan adalah RSA (Rivest, Shamir, Adleman) – CRT (*Chinese Remainder Theorem*). RSA – CRT menggunakan 2 buah kunci berbeda dalam melakukan enkripsi dan dekripsi yaitu kunci publik untuk enkripsi dan kunci privat untuk dekripsi. Dengan adanya aplikasi enkripsi data teks dengan menggunakan metode RSA – CRT diharapkan dapat membantu pengguna ponsel pintar dengan sistem operasi Android dalam hal meningkatkan keamanan data terutama data teks.

***Kata Kunci**:* enkripsi, dekripsi, RSA – CRT, kriptografi kunci asimetris

1. PENDAHULUAN

Keamanan data merupakan sesuatu yang harus diperhatikan dalam kemajuan teknologi informasi, terutama dalam pesan yang berbentuk teks. Sistem keamanan pesan yang berbentuk teks salah satunya dengan sistem kriptografi.

Sistem kriptografi adalah ilmu yang menggunakan teknik matematika sebagai landasan untuk membuat sebuah sistem keamanan informasi. Tujuan utama sistem kriptografi adalah mengamankan informasi yang bersifat rahasia serta menjaga keutuhan informasi tersebut. Salah satunya adalah menjaga kemanan terhadap pesan yang berbentuk teks.

Salah satu metode kriptografi adalah metode RSA (Rivest, Shamir, Adleman) – CRT (*Chinese Remainder Theorem*). Dalam hal ini RSA – CRT berbeda dengan RSA biasa, karena mengggunakan teorema CRT untuk memperpendek ukuran bit kunci privat dengan cara menyembunyikan kedalam sistem kongruen, sehingga dalam proses dekripsi menjadi lebih cepat [1].

2. KRIPTOGRAFI RSA DAN CRT

2.1. Sistem Kriptografi RSA (Rivest, Shamir, Adleman)

Pada tahun 1977, Ron (R)ivest, Adi (S)hamir, dan Lonard (A)dleman merumuskan algoritma praktis yang mengimplementasikan sistem kriptografi kunci publik disebut dengan kriptografi RSA. Meskipun pada tahun 1997 badan sandi memublikasikan bahwa Clifford Cock telah merumuskan sistem RSA 3 tahun lebih dahulu dari pada Rivest, Shamir, dan Adleman [2].

2.2. *Chinese Remainder Theorem* (CRT)

Chinese Remainder Theorem (CRT) merupakan problem klasik pada terori bilangan, pertama kali dirumuskan pada teks klasik china oleh Sun Tsu, CRT digunakan salah satunya sebagai varian sistem kriptografi publik RSA yang disebut RSA – CRT. CRT diformulasikan sebagai penyelesaian permasalahan kongruen dengan modulus berbeda [1].

2.3. RSA dengan CRT

RSA dengan CRT adalah sistem kriptografi RSA yang dimodifikasi dengan menggunakan teorema CRT. Teorema CRT dalam RSA – CRT bertujuan untuk memperpendek ukuran dekripsi d dengan cara menyembunyikan d pada sistem kongruen sehingga mempercepat waktu dekripsi [1].

2.3.1. Pembangkit kunci RSA – CRT

Pada pembangkit kunci RSA – CRT eksponen dekripsi d tidak secara langsung diberikan pada kunci privat namun dapat dihitung melalui parameter dP, dQ, dan qInv yang memiliki ukuran setengah dari panjangn bit d. Algoritma pembangkit kunci RSA – CRT adalah sebagai berikut :

1. Pilih dua buah bilangan prima sembarang, p dan q
2. Hitung $\mathrm{n} = \mathrm{p}\,.\,\mathrm{q}$ (sebaiknya $\mathrm{p} \neq \mathrm{q}$, sebab jika $\mathrm{p} = \mathrm{q}$ maka $\mathrm{n} = \mathrm{p}^2$ sehingga p dapat diperoleh dengan menarik akar pangkat dua dari n)
3. Hitung $\emptyset(\mathrm{n}) = (\mathrm{p}-1)(\mathrm{q}-1)$
4. Pilih e, yang relatif prima terhadap $\emptyset(\mathrm{n})$
5. $\mathrm{d} = \mathrm{e}^{-1}$ pada $\mathbb{Z}_{\emptyset(\mathrm{n})}$
6. $\mathrm{dP} = \mathrm{d} \bmod (\mathrm{p}-1)$
7. $\mathrm{dQ} = \mathrm{d} \bmod (\mathrm{q}-1)$
8. $\mathrm{qInv} = \mathrm{q}^{-1}$ pada $\mathbb{Z}_{\mathrm{p}}$
9. $\mathrm{K}_{\mathrm{publik}} = (\mathrm{e}, \mathrm{n}), \mathrm{K}_{\mathrm{privat}} =$ (dP, dQ, qInv, p, q).

2.3.2. Enkripsi RSA – CRT

Dalam proses enkripsi RSA – CRT tidak mengalami perbedaan dengan enkripsi pada algoritma RSA biasa dengan menggunakan fungsi

$$c_i = m_i{}^e \bmod n \qquad (2.1)$$

Dimana:

- $c_i = chipertext$
- $m_i = plaintext$
- $e, n =$ kunci publik

2.3.3. Dekripsi RSA – CRT

Dalam proses dekripsi RSA – CRT dengan menghitung kembali d dengan menggunakan parameter pada kunci privat, yaitu dP, dQ, dan qInv. Berdasarkan penyelesaian persoalan CRT d dapat dihitung kembali sehingga memulihkan teks sandi untuk mendapatkan kembali teks asli. Fungsi yang digunakan untuk mendekripsi teks sandi adalah sebagai berikut :

1. $m_1 = c_i^{dP} \bmod p$
2. $\mathrm{m}_2 = \mathrm{c}_\mathrm{i}^{\mathrm{dQ}} \bmod \mathrm{q}$
3. $\mathrm{h} = \mathrm{qInv}.(\mathrm{m}_1 - \mathrm{m}_2) \bmod \mathrm{p}$
4. $\mathrm{m}_\mathrm{i} = \mathrm{m}_2 + \mathrm{h}.\mathrm{q}$

Dimana:

- $m_i = plaintext$
- $c_i =$ chipertext
- $dP, dQ, p, q,$ dan qInv = kunci privat

2.4. Android

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi. Android menyediakan *platform* yang terbuka bagi para pengembang untuk menciptakan aplikasi mereka [3].

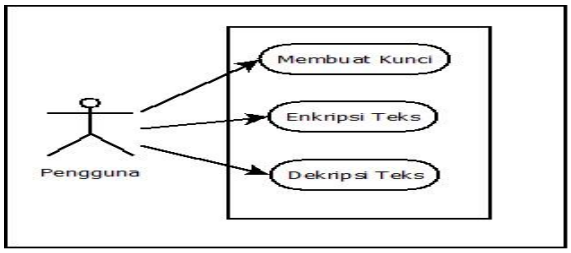
Pengertian lain Android adalah sistem operasi bergerak (*mobile operating system*) yang mengadopsi sistem operasi linux, namun telah dimodifikasi. Android diambil oleh Google pada

tahun 2005 dari Android, Inc sebagai bagian dari strategi untuk mengisi pasar sistem operasi bergerak [5].

3. PERANCANGAN APLIKASI

UML (*Unified Modelling Language*) digunakan dalam perancangan aplikasi ini. UML adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi obyek [4].

3.1. *Use Case* Utama

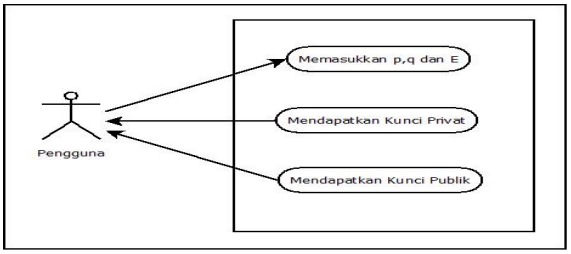


Gambar 1. *Use Case* Utama

Pada gambar diatas menjelaskan kegiatan yang dapat dilakukan oleh pengguna diantaranya adalah sebagai berikut

1. Membuat kunci
   Sebelum melakukan enkripsi dan dekripsi pengguna dapat membuat kunci terlebih dahulu.
2. Enkripsi teks
   Setelah pengguna mendapatkan kunci publik, pengguna dapat melakukan enkripsi *plaintext* menjadi *chipertext*.
3. Dekripsi teks
   Setelah pengguna mendapatkan *chipertext* dan mempunyai kunci *privat*, pengguna dapat melakukan dekripsi terhadap *chipertext* tersebut menjadi *plaintext* kembali.

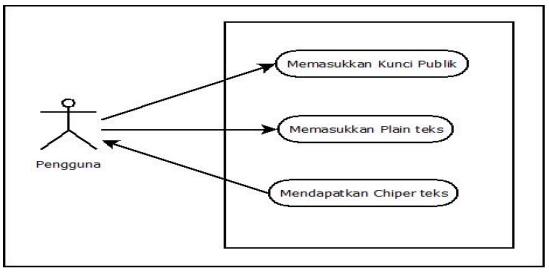
3.2. *Use Case* Membuat Kunci



Gambar 2. *Use Case* pembuatan kunci

Pada gambar diatas dijelaskan pengguna yang akan membuat kunci harus memasukkan p, q dan E kedalam aplikasi tersebut. Dengan ketentuan bahwa p dan q adalah bilangan prima. Setelah memasukkan p, q dan E kedalam aplikasi tersebut pengguna akan mendapatkan kunci privat dan kunci publik dari aplikasi tersebut. Kunci privat nantinya akan disimpan oleh pengguna untuk melakukan dekripsi sedangnkan kunci publiknya dikirim kepada pengguna lain yang nantinya digunakan untuk melakukan enkripsi.

3.3. *Use Case* Enkripsi Teks

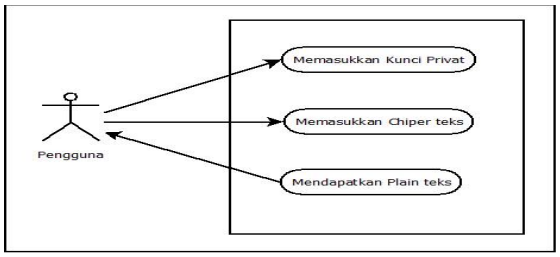


Gambar 3. *Use Case* Enkripsi Teks

Pada gambar diatas digambarkan bahwa pengguna yang telah mendapatkan kunci publik dapat memasukkan kunci publik tersebut kedalam aplikasi. Kunci publik terdiri dari 2 angka yang didapatkan dari proses pembuatan kunci. Setelah memasukkan kunci publik pengguna dapat memasukkan teks yang akan dienkripsi. Setelah pengguna memasukkan teks

nantinya aplikasi tersebut akan memproses. Hasil dari proses tersebut nantinya berupa teks yang telah dienkripsi yang nantinya akan dapat disimpan sendiri atau dikirim tergantung dari pengguna tersebut.

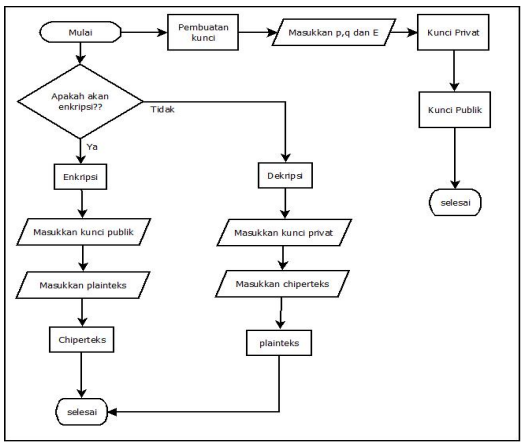
3.4. *Use Case* Dekripsi Teks



Gambar 4. *Use Case* Dekripsi Teks

Gambar 4 menggambarkan tentang proses dekripsi teks yang telah dienkripsi. Pertama pengguna memasukkan privat key yang merupakan pasangan publik key yang digunakan enkripsi teks tersebut. Setelah itu pengguna memasukkan teks hasil dari enkripsi. Setelah teks tersebut dimasukkan aplikasi tersebut akan memproses sehingga menjadi teks asli sebelum dienkripsi.

3.5. Flow Chart

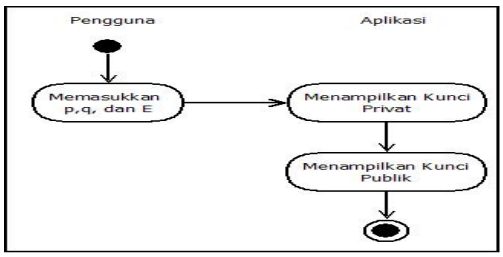


Gambar 5. *Flow Chart* RSA – CRT

3.6. *Activity Diagram*

Activity diagram dalam sistem ini menjelaskan tentang rincian aktivitas yang dilakukan oleh sistem selama digunakan dan menjelaskan keluaran dari sistem tersebut.

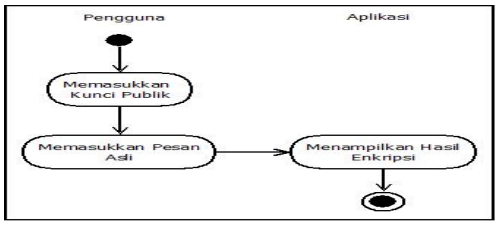
a. *Activity diagram* pembuatan kunci



Gambar 6. *Activity Diagram* Pembuatan Kunci

Pada *activity diagram* pembuatan kunci pengguna terlebih dahulu memasukkan p, q dan E kedalam aplikasi. Setelah itu aplikasi akan memproses. Setelah proses selesai maka aplikasi akan menampilkan kunci privat dan menampilkan kunci publik.

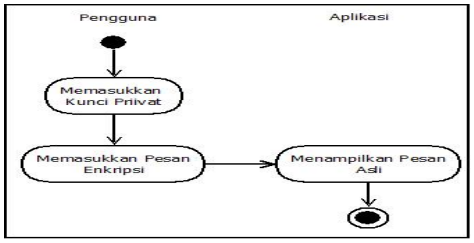
b. *Activity diagram* enkripsi



Gambar 7. *Activity Diagram* Enkripsi

Pada *activity diagram* enkripsi pengguna terlebih dahulu memasukkan kunci publik kedalam aplikasi tersebut. Setelah itu pengguna memasukkan teks asli yang akan dienkripsi. Setelah itu aplikasi akan memproses kunci publik dan pesan asli menjadi pesan yang telah dienkripsi yang nantinya akan ditampilkan.

c. *Activity diagram* dekripsi

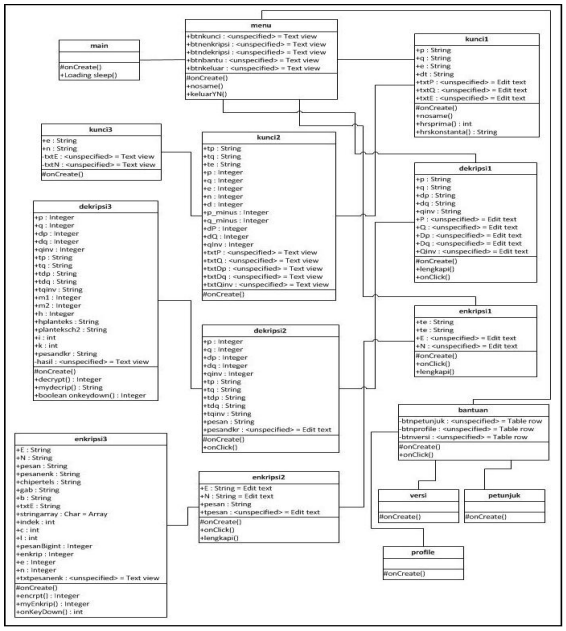


Gambar 8. *Activity Diagram* Dekripsi

Pada *activity diagram* dekripsi pengguna terlebih dahulu memasukkan kunci privat kedalam aplikasi. Setelah itu pengguna memasukkan teks hasil dari enkripsi. Setelah itu aplikasi akan memproses kunci privat dan teks hasil enkripsi menjadi pesan asli sebelum dienkripsi. Setelah itu aplikasi akan menampilkan hasil pemprosesan yang berupa teks asli. Setelah hasil ditampilkan maka proses dianggap selesai.

3.7. *Class Diagram*

Gambaran hubungan antar *class – class* pada aplikasi enkripsi teks dengan metode RSA – CRT akan digambarkan dengan *class diagram* sebagai berikut :



Gambar 9. *Class Diagram*

Pada aplikasi enkripsi teks dengan metode RSA – CRT menggunakan beberapa *class*, antara lain sebagai berikut :

a. *Class* main

Pada *class* main digunakan untuk menampilkan tampilan *layout* main dan menampilkan tampilan awal aplikasi tersebut

b. *Class* menu

Class menu akan muncul setelah tampilan main selesai, dan digunakan untuk menampilkan beberapa menu yang diantaranya menu pembuatan kunci, enkripsi, dekripsi serta menu bantuan.

c. *Class* kunci 1

Class kunci 1 digunakan untuk memasukkan angka p, q dan e yang nantinya akan digunakan pada *class* kunci 2.

d. *Class* kunci 2

Setelah p, q dan e diisi pada class kunci 1 pada class kunci 2 akan diolah dan menghasilkan kunci privat dan kunci publik, tetapi pada *class* kunci 2 hanya ditampilkan kunci privat saja, sedangkan hasil pembuatan kunci publik nantinya akan ditampilkan pada *class* kunci 3.

e. *Class* kunci 3

Pada class kunci 3 akan menampilkan hasil pembuatan kunci pada class kunci 2, dan hanya kunci publik saja yang ditampilkan.

f. *Class* enkripsi1

Class enkripsi1 digunakan untuk memasukkan kunci publik dari hasil pembuatan kunci yang nantinya akan dibawa pada *class* enkripsi2 dan dibawa ke *class* enkripsi3 untuk diproses dengan teks yang akan dienkripsi.

g. *Class* enkripsi2

Pada *class* enkripsi2 akan digunakan untuk menampung masukan yang berupa kunci publik pada *class* enk 1 dan digunakan untuk memasukkan pesan yang akan dienkripsi. Setelah itu, pesan dan kunci publik akan dibawa ke *class* enkripsi3.

h. *Class* enkripsi3

Setelah pesan dan kunci publik dibawa ke dalam *class* enkripsi3, maka *class* enkripsi3 akan menampung pesan dan kunci publik tersebut. Setelah itu, *class* enkripsi3 akan memproses dan menampilkan hasil dari enkripsi pesan tersebut.

i. *Class* dekripsi1

C*lass* dekripsi1 digunakan untuk memasukkan kunci privat yang nantinya akan dibawa kedalam *class* dekripsi2 dan dibawa kembali ke *class* dekripsi3 untuk diproses bersamaan dengan pesan yang telah dienkripsi.

j. *Class* dekripsi2

Setelah memasukkan kunci privat dibawa masuk kedalam *class* dekripsi2, maka *class* dekripsi2 akan menampung kunci privat tersebut. Selain itu, dalam *class* dekripsi2 juga digunakan untuk memasukkan pesan hasil dari proses enkripsi. Kunci privat dan pesan hasil enkripsi nantinya akan dibawa ke *class* dekripsi3.

k. *Class* dekripsi3

Kunci privat dan pesan enkripsi yang berasal dari *class* dekripsi2 ditampung oleh *class* dekripsi3 untuk diproses dan ditampilkan menjadi pesan asli sebelum pesan tersebut dienkripsi.

l. *Class* bantuan

Pada *class* ini akan menampilkan 3 sub menu yaitu petunjuk, versi, dan profile yang berupa *table row*.

m. *Class* petunjuk

Class petunjuk akan menampilkan tahapan penggunaan aplikasi mulai dari pembuatan kunci, proses enkripsi dan proses dekripsi.

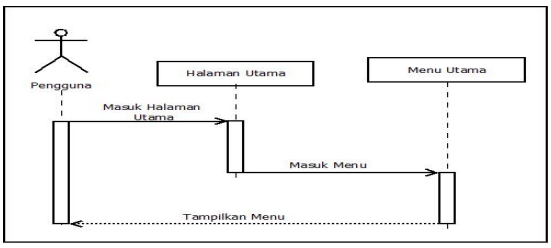
n. *Class* profile

Class profile akan menampilkan profile dari pembuat aplikasi.

3.8. *Sequence Diagram*

Sequence Diagram menggambarkan tentang sekenario atau langkah-langkah yang dilakukan sistem sebagai umpan balik (*feedback*) dari yang pengguna lakukan pada sistem tersebut. Berikut adalah *sequence diagram* yang ada pada sistem enkripsi data teks dengan metode RSA – CRT.

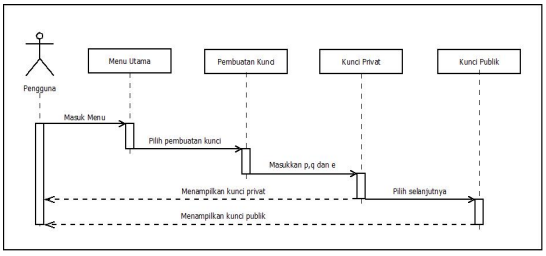
1. *Sequence diagram* menu utama



Gambar 10. *Sequence Diagram* Menu Utama

Pada *Sequence Diagram* menu utama adalah proses dimana pengguna akan masuk kehalaman utama dan masuk kedalam menu utama. Dalam menu tersebut pengguna dapat memilih beberapa menu seperti : pembuatan kunci, enkripsi, dekripsi, serta menu bantuan.

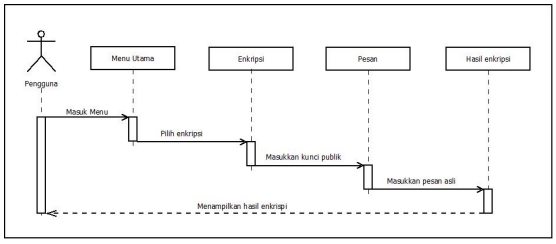
2. *Sequence diagram* pembuatan kunci



Gambar 11. *Sequence Diagram* Pembuatan Kunci

Pada *sequence diagram* pembuatan kunci digambarkan bahwa pengguna terlebih dahulu masuk menu utama dan setelah itu memilih menu pembuatan kunci. Setelah memilih menu pembuatan kunci pengguna memasukkan p,q dan e. Setelah memasukkan p,q dan e pengguna akan menunggu pemprosesan sehingga menghasilkan kunci privat dan kunci publik.

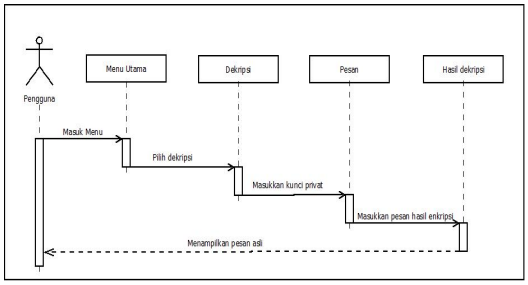
3. *Sequence diagram* enkripsi



Gambar 12. *Sequence Diagram* Enkripsi

Pada *sequence diagram* enkripsi pengguna masuk menu utama dan memilih menu enkripsi setelah itu pengguna diharuskan mengisi kunci publik yang diperoleh dari hasil pembuatan kunci. Setelah itu pengguna memasukkan pesan asli yang akan dienkripsi. Nantinya aplikasi tersebut akan memproses dan menghasilkan sebuah hasil pesan enkripsi dari pesan asli tersebut.

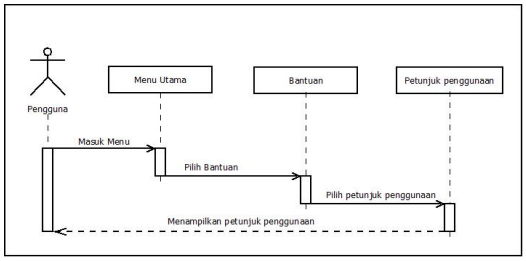
4. *Sequence diagram* dekripsi



Gambar 13. *Sequence Diagram* Enkripsi

Pada *sequence diagram* ekripsi digambarkan bahwa pengguna masuk ke menu utama setelah itu masuk kemenu dekripsi. Setelah masuk kedalam menu dekripsi pengguna wajib memasukkan kunci privat yang berasal dari proses pembuatan kunci. Setelah memasukkan kunci privat pengguna memasukkan pesan yang telah dienkripsi yang nantinya akan diproses oleh aplikasi tersebut dan menghasilkan pesan asli yang nantinya akan ditampilkan sebagai umpan balik dari aplikasi tersebut.

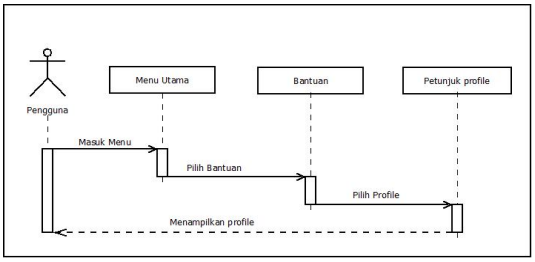
5. *Sequence diagram* petunjuk



Gambar 14. *Sequence Diagram* Petunjuk

Pada *sequence diagram* petunjuk menggambarkan tahapan untuk masuk kedalam sub menu petunjuk penggunaan. Pengguna masuk kedalam menu utama dan memilih bantuan. Setelah itu pengguna memilih petunjuk penggunaan yang nantinya akan ditampilkan berisin petunjuk penggunaan sebagai umpan balik dari aplikasi tersebut.

6. *Squence diagram* profile



Gambar 15. *Squence Diagram* Profile

Pada *squence diagram* profile menggambarkan tahapan untuk masuk kedalam sub menu profile. Pengguna masuk kedalam menu utama dan memilih bantuan. Setelah itu pengguna memilih profile penggunaan yang nantinya akan ditampilkan berisi profile dari pembuat aplikasi sebagai umpan balik dari aplikasi tersebut.

4. IMPLEMENTASI

4.1. Spesifikasi Perangkat Keras

Percobaan aplikasi tersebut diperlukan beberapa *handset* sebagai perangkat untuk melakukan test aplikasi tersebut, berikut ini adalah spesifikasi perangkat keras yang digunakan dalam pembuatan aplikasi dan juga dalam melakukan test aplikasi tersebut.

1. Sony Ericsson Neo V

   Handset yang pertama digunakan untuk melakukan test aplikasi adalah Sony Ericsson Neo V.

2. Samsung Galaxy Ace

   Handset kedua yang digunakan untuk melakukan test aplikasi adalah Samsung Galaxy Ace.

3. Andro Tab 7’ SmartFren

   Handset ketiga yang digunakan untuk melakukan test aplikasi adalah Andro Tab 7’ SmartFren.

4. Asus 43SD-3FVX (K43SD)

   Notebok Asus 43SD-3FVX (K43SD) digunakan untuk membuat, merancang serta melakukan simulasi dengan *emulator* apakah aplikasi tersebut dapat berjalan sebelum digunakan dengan *handset* yang sebenarnya.

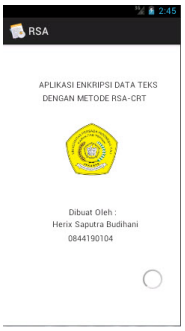
4.2. Pengujian

Pengujian dilakukan dalam dua tahap utama, yaitu pengujian *emulator and handset*. Pengujian emulator dilakukan dengan menguji atribut dan method yang ada pada kelas-kelas yang dibangun. Pengujian ini dilakukan pada proses pengembangan pada emulator.

Pengujian *handset* dilakukan dengan cara mengukur kecepatan dalam hal pembuatan kunci, enkripsi dan dekripsi yang bertujuan melihat apakah setiap *handset* mempunyai kecepatan yang dalam melakukan pembuatan kunci, enkripsi dan dekripsi. Berikut ini adalah hasil dari pengujian tersebut:

4.2.1. Pengujian Emulator

a. Main

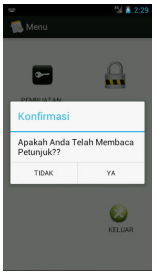


Gambar 17. Tampilan Main

b. Menu

Gambar 18. Tampilan Menu

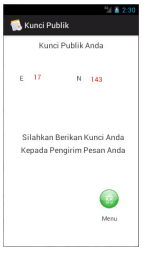
c. Pembuatan Kunci



Gambar 19. Tampilan Syarat Memulai Pembuatan kunci

Gambar 20. Tampilan Memasukkan P, Q, dan E

Gambar 21. Tampilan Kunci Privat



Gambar 22. Tampilan Kunci Publik

d. Enkripsi

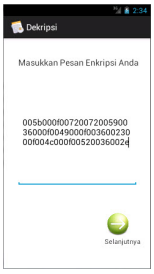
Gambar 23. Tampilan Memasukkan Kunci Publik

Gambar 24. Tampilan Memasukkan *Plaintext*

Gambar 25. Tampilan Hasil Enkripsi

e. Dekripsi

Gambar 26. Tampilan Memasukkan Kunci Privat

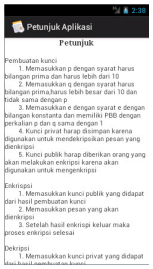


Gambar 27. Tampilan Memasukkan *Chipertext*

Gambar 28 .Tampilan Hasil Dekripsi

f. Bantuan

Gambar 29. Tampilan Bantuan



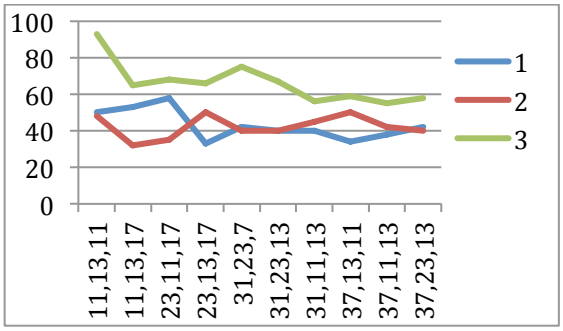
Gambar 30. Tampilan Petunjuk

Gambar 31. Tampilan Profile

4.2.2. Pengujian Kecepatan

a. Pengujian kecepatan membuat kunci

Pengujian kecepatan membuat kunci dilakukan dengan cara memasukkan p, q dan e ke dalam aplikasi dan setelah itu diukur kecepatan membuat kunci dengan menggunakan *stopwatch* secara manual. Hasil dapat berbeda ditiap *handset* walaupun dengan dengan merek dan tipe yang sama, karena dipengaruhi oleh besar kapasitas penyimpanan yang belum terpakai dan besar kapasitas RAM yang tidak terpakai.



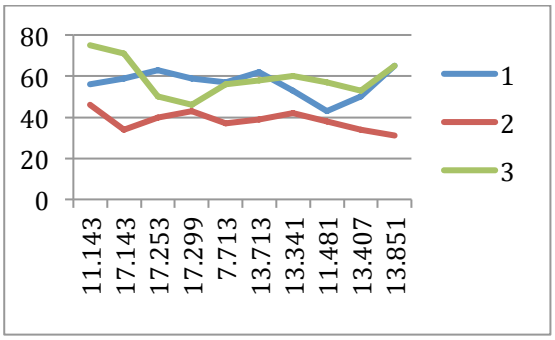
Gambar 32. Pengujian membuat kunci

Keterangan:

- *Handset* 1 = SonyEricsson Neo V
- *Handset* 2 = Samsung Galaxy Ace
- *Handset* 3 = Smartfren Andro Tab 7'

b. Pengujian kecepatan enkripsi

Pengujian kecepatan enkripsi dilakukan dengan cara memasukkan kunci publik ke dalam aplikasi setelah itu memasukkan pesan yang akan dienkripsi dengan kata “herix saputra budihani 0844190104” lalu diukur kecepatan enkripsi dengan menggunakan *stopwatch* secara manual. Hasil dapat berbeda ditiap *handset* walaupun dengan dengan merek dan tipe yang sama, karena dipengaruhi oleh besar kapasitas penyimpanan yang belum terpakai dan besar kapasitas RAM yang tidak terpakai. Hasil dari pengujian tersebut adalah sebagai berikut :



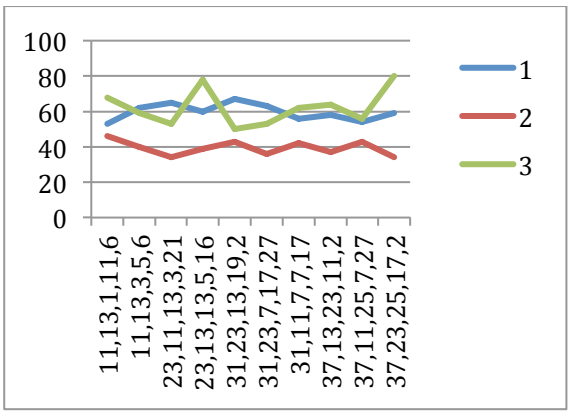
Gambar 33. Pengujian enkripsi

Keterangan:

- *Handset* 1 = SonyEricsson Neo V
- *Handset* 2 = Samsung Galaxy Ace
- *Handset* 3 = Smartfren Andro Tab 7'

c. Pengujian kecepatan dekripsi

Pengujian kecepatan dekripsi dilakukan dengan cara memasukkan kunci privat ke dalam aplikasi setelah itu dimasukkan hasil enkripsi dari proses enkripsi pada tahap sebelumnya lalu diukur kecepatan dekripsi dengan menggunakan *stopwatch* secara manual. Hasil dapat berbeda ditiap *handset* walaupun dengan dengan merek dan tipe yang sama, karena dipengaruhi oleh besar kapasitas penyimpanan yang belum terpakai dan besar kapasitas RAM yang tidak terpakai. Hasil dari pengujian tersebut adalah sebagai berikut:



Gambar 34. Pengujian dekripsi

Keterangan:

- *Handset* 1 = SonyEricsson Neo V
- *Handset* 2 = Samsung Galaxy Ace
- *Handset* 3 = Smartfren Andro Tab 7'

5. KESIMPULAN

Aplikasi enkripsi data teks dengan menggunakan metode RSA – CRT berbasis Android adalah Aplikasi enkripsi data teks dengan menggunakan metode RSA – CRT yang dapat digunakan *handset* dengan sistem operasi Android 2.2 keatas. Perangkat keras yang digunakan oleh pengguna berpengaruh terhadap kecepatan dalam membuat kunci, enkripsi dan dekripsi. Pengembangan yang dapat dilakukan oleh aplikasi enkripsi data teks dengan RSA – CRT adalah dengan menambahkan sistem pengiriman baik menggunakan *email* maupun lewat pesan SMS.

REFERENSI

[1] Munir, Rinaldi. *Kriptografi.* Bandung: Informatika, 2006.

[2] Sadikin, Rifki. *Kriptografi untuk Keamanan Jaringan* . Yogyakarta: ANDI, 2012.

[3] Safaat, Nazaruddin. *Pemrograman Aplikasi Mobile Smart Phone dan Tablet PC Berbasis Android.* Bandung: Informatika, 2012.

[4] Munawar. *Pemodelan Visual dengan UML.* Jakarta: Graha Ilmu, 2005.

[5] Suprianto, Dodit, dan Rini Agustina. *Pemrograman Aplikasi Android.* Yogyakarta: Mediakom, 2012.