Penerapan Algoritma *Blowfish* dengan bahasa *java* untuk keamanan pengiriman data dari android ke *web server*

Ahmad Faiz Sahupala / 13516065
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
faiz.sahupala29@gmail.com

***Abstract*—pengiriman data dari android ke web server merupakan hal yang lazim dilakukan oleh para pengembang aplikasi berbasis android. Data yang dikirim merupakan data data privasi pengguna aplikasi android. Tetapi, pengiriman data tersebut sangatlah rawan dikarenakan data-data yang dikirimkan dapat diintip oleh orang yang ingin mengetahui data yang dikirimkan dengan cara yang tidak baik. Oleh karena itu, untuk menjaga keamanan pengiriman data dari android ke *web server*, diperlukan satu teknik kriptografi yang dapat membantu mengatasi masalah tersebut. Oleh karena itu, penulis akan membahas tentang Algoritma *Blowfish* karena algoritma ini merupakan salah satu teknik kriptografi yang memiliki tingkat keamanan yang tinggi.**

***Keywords*—Algoritma *Blowfish*, Kriptografi, android, *web server*, pengiriman data**

1. PENDAHULUAN

Pengiriman data adalah suatu kegiatan dimana ada pengirim dan data ada penerima data. Data yang ingin disampaikan oleh pengirim dan akan diterima oleh penerima data. Pengiriman data dilakukan untuk menghubungkan pengirim dan penerima data.

Dalam hal ini, aplikasi berbasis android juga melakukan pengiriman data kepada *web server*. *Web server* bertindak sebagai penerima data. Aplikasi android pada umumnya mengirimkan data data privasi dari pengguna seperti nama, PIN, password, nomor kartu kredit, dll. Data-data tersebut adalah data data privasi pengguna harus dilindungi oleh pengembang aplikasi. Karena, dalam penjelasan Pasal 26 UU ITE ayat 2, Setiap orang yang dilanggar haknya sebagaimana dimaksud pada ayat 1 dapat mengajukan gugatan atas kerugian yang ditimbulkan berdasarkan undang-undang ini. Pasal dibuat untuk membuat para pengembang aplikasi lebih waspada dengan serangan para peretas diluar sana yang berniat jahat. Pengembang aplikasi juga harus membuat pengguna yakin akan kerahasiaan data yang telah pengguna berikan kepada pengembang aplikasi. Untuk memberikan kemanan yang membuat pengguna yakin, pengembang aplikasi menggunakan teknik kriptografi untuk meng-enkripsi dan mendeskripsikan data-data yang dikirimkan dari android ke web server. Tetapi, pengembang juga memerlukan teknik kriptografi yang memiliki tingkat keamanan yang tinggi agar tidak dapat di baca oleh para peretas yang berniat jahat. Karena, semakin sulit data yang di enkripsi oleh pengirim data, maka para pertas akan kesusahan untuk melakukan pencurian data. Oleh karena itu, penulis akan membahas tentang teknik kriptografi dengan menggunakan algoritma *blowfish* untuk mengenkripsi dan mendekripsi data yang akan dikirimkan. Algoritma *blowfish* merupakan salah satu alogitma yang memiliki kemanan sangat tinggi, belum ada *cryptoanalist* ataupun peretas yang dapat memecahkan kode yang di enkripsi oleh algoritma *blowfish* . maka dari itu, penerapan algoritma *blowfish* untuk pengiriman data dari android ke *web server* sangatlah cocok untuk mengatasi permasalahan keamanan aplikasi yang dibuat oleh pengembang aplikasi.

2. TEORI DASAR

2.1 Keamanan jaringan

Sistem keamanan jaringan adalah proses untuk mencegah dan mengindentifikasi pengguna yang tidak sah dari jaringan komputer. Tujuannya adalah untuk mengantisipasi resiko jaringan komputer yang dapat berupa ancaman fisik maupun logik. Yang dimaksud ancaman fisik adalah yang merusak bagian fisik komputer atau hardware komputer sedangkan ancaman logik yaitu berupa pencurian data atau penyusup yang membobol akun seseorang. Keamanan jaringan di bagi menjadi 4 bagian yaitu :

- Autentikasi
  Proses pengenalan peralatan, sistem operasi, aplikasi dan identitas user yang terhubung dengan jaringan komputer. Misalnya, user memasukan username dan password pada saat login ke jaringan.
- Enkripsi
  Teknik pengkodean data yang dapat berguna untuk menjaga data.
- VPN (*Virtual Private Network*)
  Jaringan komunikasi lokal yang dapat terhubung melalui media jaringan. Fungsi dari VPN tersendiri untuk memperoleh komunikasi yang aman melalui internet.

- DMZ(De-Militerized Zone)
  Melindungi sistem internal dari serangan hacker.

Setelah itu, keamanan jaringan memiliki banyak gangguan eksternal yang dilakukan oleh *programmer* jahat. Contoh gangguan pada keamanan jaringan adalah sebagai berikut :

- Carding
  Pencurian data terhadap identitas perbankan seseorang. Contoh, pencurian nomor kartu kredit untuk melakukan belanja online
- Physing
  Pemalsuan terhadap data resmi
- Deface
  Perubahan terhadap bentuk atau tampilan website
- Hacking
  Perusakan pada infrastuktur jaringan komputer yang sudah ada

Pada makalah kali ini, penulis akan membahas banyak tentang enkripsi data, karena algoritma *blowfish* merupakan aloritma untuk mengenkripsi data. Algoritma *blowfish* akan banyak membantu dalam mengatasi gangguan keamanan yang dilakukan oleh para *programmer* jahat.

2.2 Kriptografi

Kriptografi merupakan ilmu sekaligus seni untuk menjaga keamanan pesan. Secara etimologi kata kriptografi (*Cryptography)* berasal dari bahasa yunani, yaitu kriptos yang artinya yang tersembunyi dan graphien yang artinya tulisan. Awal mula kriptografi dipahami sebagai ilmu tentang menyembunyikan pesan,tetapi seiring perkembangan zaman hingga saat ini pengertian kriptografi berkembang menjadi ilmu tentang teknik matematis yang digunakan untuk menyelesaikan persoalan keamanan berupa privasi dan otentikasi. Keamanan pesan diperoleh dengan menyandikannya menjadi pesan yang tidak memiliki makna. Zaman sekarang inikerahasiaan informasi menjadi suatu yang penting. Informasi yang rahasian perlu disembunyikan agar tidak diketahui oleh orang yang tidak berhak.

- Sejarah kriptorafi
  Menurut sejarahnya, kriptografi sudah lama digunakan oleh tentara sparta di yunani pada permulaan tahun 400 SM. Mereka menggunakan alat yang disebut *scytale.* Alat ini terdiri dari sebuah pita panjang dari daun papyrus yang dililitkan pada sebatang silinder. Pesan yang akan dikirimkan ditulis horizontal. Bila peta dilepaskan, maka huruf-huruf di dalamna telah tersusun membentuk pesan rahasian. Untuk membaca pesan, penerima melilitkan kembali silinder yang diameternya sama dengan diameter silinder pengirim. Teknik kriptografi seperti ini dikenal dengan nama transposisi cipher, yang merupakan metode enkripsi tua.

Gambar 1. *Scytale*
Sumber :
https://upload.wikimedia.org/wikipedia/commons/thumb/5/51/Skytale.png/1200px-Skytale.png

Ada beberapa istilah dalam kriptografi, yaitu sebagai berikut,

- Plainteks
  Pesan yang dirahasiakan, pesan yang akan dikirimkan
- Cipherteks
  Pesan hasil penyandian setelah plainteks di enkripsi
- Pengirim
  Pengirim adalah entitas yang mengirim pesan ke entitas lainnya
- Penerima
  Entitas yang menerima pesan
- Penyadap
  Orang yang mencoba menangkap pesan yang sedang ditransmisikan
- Kriptanalisis
  Ilmu dan seni untuk memecahkan cipherteks menjadi plainteks tanpa perlu menggunakan kunci yang dibuat oleh pengenkripsi pesan. Orang yang mempelajari dan melakukan kriptanalisis disebut kriptanalis
- Kriptologi
- Studi yang mempelajari dan membahas tentang kriptografi dan kriptanalisis
- Enkripsi
  Proses menyandikan plainteks menjadi chiperteks
- Deskripsi
  Proses membaca chiperteks agar dapat dibuat menjadi plainteks
- Key
  Parameter yang digunakan untuk mengenkripsi dan mengdeskripsi.

Teknik kriptografi untuk mengenkripsi data sangat banyak. Tetapi, hanya beberapa teknik yang memiliki tingkat keamanan tinggi, yang mana para kriptanalis masih sulit untuk mendeskripsikannya tanpa kunci.

2.3 Algoritma *blowfish*

Algoritma *blowfish* merupakan salah satu teknik kriptografi. Algoritma ini merupakan salah satu algoritma yang memiliki tingkat keamanan sangat tinggi. Bahkan, belum ada kriptanalis di dunia ini yang dapat memecahkan chiperteks tanpa memiliki kunci yang dibuat oleh pengenkripsi pesan. Keunggulan dari

algoritma *blowfish* adalah cepat, compact(dapat dijalankan pada memori kurang dari 5K), sederhana,algoritma *blowfish* hanya menggunakan operasi sederhana seperti : penambahan, XOR, dan lookup tabel , memiliki tingkat keamanan yang bervariasi.

Struktur algoritma *blowfish* , *blowfish* adl block cipher 64-bit dengan panjang kunci variabel. Algoritma ini terdiri dari dua bagian yaitu key expansion atau perluasan kunci dan enkripsi data. Key expension berfungsi merubah kunci jadi beberapa array subkunci dengan total 4168 byte. Kunci-kunci ini dibangkitkan dengan menggunakan subkunci yang harus dihitung terlebih dahulu sebelum enkripsi atau deskripsi data. Sub-sub kunci yang digunakan terdiri dari : P-array yang terdiri dari 18 buah 32-bit subkunci. Sedangkan enkripsi data terdiri dari iterasi fungsi sederhana sebanyak 16 kali putaran. Setiap putaran terdiri dari permutasi dan substitusi kunci kunci dan data dependent. Semua operasi mrp penambahan dan XOR pada variabel 32-bit.

Algoritma *blowfish* memiliki keunggulan karena keunikan alogirtma ini pada saat prose enkripsi dan deskripsi. Yaitu proses deskripsi engan urutan yang sama persis dengan proses enkripsi, hanya saja pada proses deskripsi P1,P2, . . ., P18 dipakai dalam urutan terbalik. Sampai saat ini algoritma *blowfish* belum ditemukan kelemahan yang berarti hanya ada *weak key* dimana dua entri S-box mempunyai nilai yang sama. Belum ada cara untuk mengecek *weak key* sebelum melakukan key expansion. Agar aman dalam pembongkaran pesan, algoritma ini harus dilakukan pemutaran sebanya 16 kali. Algoritma blowfish pun bisa digabungkan dengan teknik teknik kriptografi lainnnya. Sehingga alogritma ini akan sangat aman untuk dipakai mengirimkan data dari android ke *web server*.

2.4 Android

Gambar 2. Logo android
Sumber : http://dc942d419843af05523b-ff74ae13537a01be6cfec5927837dcfe.r14.cf1.rackcdn.com/wp-content/uploads/Android-logo.png

Android merupakan sistem Operasi berbasis linux yang banyak digunakan pada smartphone dengan sumber daya terbuka (*open source)* dan lisensi perizinan. Dengan menggunakan Android , memungkinkan perangkat lunak untuk dimodifikasi secara bebas dan didistribusikan oleh para pembuat perangkat, operator nirkabel, dan pengembang aplikasi. Ada banyak versi dari Android yaitu mulai dari Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwich, Jelly Bean, Lollipop, Marshmallow, Nougat, dan yang terbaru Oreo.
Smartphone yan akan kami gunakan sebagai acuan pada projek kali ini adalah smartphone yang memiliki OS Android versi Marshmallow karena berdasarkan statistik survei yang dilakukan oleh google pada 2 Oktober 2017, jumlah pengguna OS Android versi Marshmallow merupakan yang terbanyak dengan persentase 32%. Menurut data dari Google Developer Group Indonesia. Terdapat seratus juta lebih pengguna OS Android di Indonesia.

2.4 *web server*

Secara umum, web server dapat diartikan sebagai sebuah software/aplikasi yang berjalan pada sebuah komputer yang dapat diakses oleh perangkat lain melalui internet sehingga dapat memberikan layanan berbasis data baik input atau output dan berfungsi menerima permintaanperintah klien melalui browser atau aplikasi khusus. Data input dan output tersebut ada dalam beberapa bentuk seperti halaman web yang pada umumnya aka berbentuk dokumen HTML, kiriman perintah/data, database dan lain sebagainya.

2.5 *java language*

Java adalah bahasa pemrograman yang sudah ada dari tahun 1990-an. Bahasa pemrograman java kian berkembang dan mendominasi di berbagai bidang teknologi. Bahasa pemrograman java merupakan bahasa pemrograman berorientasi objek (*Object Oriented Programming).* Java pun akrab dengan dunia saintifik dan akademik. Cukup banyak akademisi di Indonesia yang menggunakan bahasa Java sebagai alat menyelesaikan skripsi atau tugas akhir dengan berbagai topik seperti kecerdasan buatan, *data mining*, *architecture*, dan yang penggunaannya paling terbesar adalah pembuatan aplikasi android native. Oleh karena itu, penulis ingin menggunakan bahasa java sebagai acuan untuk menerapkan algoritma *blowfish* untuk pengiriman data. Karena android sendiri dibentuk dengan bahasa pemograman Java sebagai bahasa utamanya.
Bahasa pemrograman java mudah dipelajari, karena hampir mirip dengan bahasa pemrograman lainnya seperti Free Pascal, C, C++, C#, dan lain lain. Yang membedakan adalah bahasa ini merupakan bahasa pemrograman beroirentasi objek. Oleh karena itu, dalam bahasa java kita akan sering mendengar istilah public, class, static, void, dan lain lain.

Berikut adalah contoh potongan program dalam bahasa pemrograman Java :

```
public class HelloWorld {

    public static void main(String[] args) {
        System.out.println("Hello World...");
    }
}
```

Gambar 3. “Hello World” dalam bahasa Java

3. ANALISIS

3.1 Pengertian Key Expansion dan Langkahnya

Key Expansion berfungsi untuk merubah kunci (Minimum 32-bit, Maksimum 448-bit) jadi beberapa array subkunci. Dengan total 4168 byte. Kunci tersebut lalu di taruh di dalam K-array :

K1, K2, . . . K_j, dimana 1 < j < 14.

Kunci kunci ini dibangkitkan dengan menggunakan subkunci yang harus dihitung terlebih dahulu sebelum enkripsi data. Sub sub kunci yang digunakan terdiri dari : P-array yang terdiri dari 18 buah 32-bit subkunci.

P1, P2, . . ., P18

S-box yang terdiri dari 4 buah 32-bit, masing masing memiliki 256 entri

S1, 0, S1, 1, . . ., S1, 255
S2, 0, S2, 1, . . ., S2, 255
S3, 0, S3, 1, . . ., S3, 255
S4, 0, S4, 1, . . ., S4, 255

Langkah-langkah untuk membangkitkan subkuci adalah sebagai berikut :

1. Insialisasi P-array yang pertama dan juga empat S-box, berurutan, dengan string yang telah pasti. String tersebut terdiri dari digit-digit heksadesimal dari phi, tidak termasuk angka tiga di awal. Contoh :

   P1 = 0x243f6a88
   P2 = 0x85a308d3
   P3 = 0x13198a2e
   P4 = 0x03707344

   Dan seterusnya sampai S-box yang terakhir.

2. XOR kan P1 dengan 32-bit awal kunci, XOR-kan P2 denan 32-bit berikutnya dari kunci, dan seterusnya untuk semua bit kunci. Ulang siklus seluruh bit kunci secara berurutan sampai seluruh P-array ter-XOR-kan dengan bit-bit kunci. Atau jika disimbolkan : P1 = P1 XOR K1, P2 = P2 XOR K2, dst.
3. Enkripsikan string yang seluruhnya nol dengan algoritma *blowfish*
4. Gantikan P1 dan P2 dengan keluaran dari langkah 3
5. Enkripsikan keluaran langkah 3 dengan algoritma *blowfish*
6. Gantikan P3 dan P4 dengan keluaran dari langkah 5
7. Lanjutkan langkah langkah diatas, gantkan seluruh elemen P-array dan kemudian keempat S-box secara berurutan, dengan hasil keluaran algoritma blowfish yang terus menerus berubah.

Setelah langkah langkah tersebut dijalankan, akan bangkit sebuah kunci yang hanya pengembang yang tau. Oleh karena itu, bagian ini merupakan bagia nuntuk pembuatan kunci itu tersendiri.

3.2 Enkripsi Data

Enkripsi data terdiri dari iterasi fungsi sederhana sebanyak 16 kali putaran. Setiap putaran terdiri dari permutasi & subtitusi kunci & data dependent. Untuk operasi, sama halnya seperti memubat key, hanya saja ada sedikit tambahan yaitu operasi empat penelusuran tabel array berindeks untuk setiap tujuan. Untuk alur algoritma enkripsi edngan metode blowfish, akan dijelaskan sebagai berikut.

1. Inisialisai array P sebanyak 18 buah yang masing masing bernilai 32-bit

   P1, P2, . . ., P18

2. Bentuk S-box sebanyak 4 buah
3. *Plainteks* yang akan dienkripsikan sebagai masukan.
4. Hasil pembagian tadi dibagi dua.
5. Selanjutnya lakukan operasi sebagai berikut

   XL = XL xor Pi & XR = F(X)jadi XL

6. Hasil operasi di atas, kalian ditukar, XL menjadi XR dan XR menjadi XL
7. Lakukan sebanyak 16x
8. Pada proses ke-17 lakukan operasi ini

   XR = XR XOR P17
   XL = XL XOR P18

9. Proses terakhir satukan XR dan XL sehingga menjadi 64 bit.

Algoritma *blowfish* memiliki keunikan karena cara untuk mengenkripsikan pesan dan mendeskripsikannya tidak jauh berbeda. Hanya berbeda pada saat pemakaina P1 – P18. Pada saat deskripsi, pendeskripsi hanya perlu memakai P1 – P18 dalam urutan terbalik.

3.3 Penerapan Algoritma *Blowfish* Dalam Bahasa Pemrograman Java

Dalam hal ini, penulis memakai *text editor* Intellij IDEA 2017.2.5 sebagai tempat untuk menerapkan algoritma *blowfish* dalam bahasa Java. Langkah pertama yang penulis lakukan adalah membuat sebuah template file untuk memulai menulis kode program.

```
public class Main {

    public static void main(String[] args)
{
   // write your code here
```

```
    }
}
```

Setelah itu, buat kelas Java baru yang diberi nama BlowfishAlgorithm.java . di dalam file java tersebut, kita membuat *method* enkripsi dan deskripsi yang dapat dipakai di file Main.

Hal pertama yang penulis lakukan adalah meng-*import* library library bawaan yang diberikan dari java untuk membantu membuat algoritma blowfish. Berikut adalah library library tersebut

```
import
com.sun.org.apache.xml.internal.security.exceptions.Base64DecodingException;
import
com.sun.org.apache.xml.internal.security.utils.Base64;
import javax.crypto.*;
import javax.crypto.spec.SecretKeySpec;
import java.security.InvalidKeyException;
import
java.security.NoSuchAlgorithmException;
```

Setelah itu mendeklarasikan variabel-variabel penting yang akan membantu berjalannya algoritma

```
KeyGenerator keyGenerator = null;
SecretKey secretKey = null;
Cipher cipher = null;
SecretKeySpec secretKeySpec = null;
```

Selanjutnya, membuat sebuah *Constructor* untuk kelas tersebut

```
public BlowfishAlgorithm() {
    try {
        /**
         * Create a Blowfish key
         */
        String key = "MyOwnKey";
        byte[] keyData = key.getBytes();
        secretKeySpec = new
SecretKeySpec(keyData, "Blowfish");

        /**
         * Create an instance of cipher
mentioning the name of algorithm
         *     - Blowfish
         */
        cipher =
Cipher.getInstance("Blowfish");
    } catch (NoSuchPaddingException ex) {
        System.out.println(ex);
    } catch (NoSuchAlgorithmException ex)
{
        System.out.println(ex);
    }
}
```

pada kode diatas, dapat diperhatikan pada variabel “key”. Variabel tersebut adalah kunci utama algoritma ini. Karena variabel tersebut yang dapat membedakan antara pemakai satu dengan pemakai lainnya. Variabel tersebut juga yang merupakan variabel yang sangat dirahasiakan agar tidak dapat diketahui oleh *programmer* selain tim pengembang.

Setelah itu membuat fungsi encrypt yang memiliki parameter input sebuah string yang ingin dijadikan chiperteks.

```
public String encrypt(String plainText) {
    String cipherText = null;
    byte[] cipherBytes
=encryptText(plainText);
    cipherText =
bytesToString(cipherBytes);
    return cipherText;
}
```

```
public byte[] encryptText(String
plainText) {
    byte[] cipherBytes = null;
    try {
        cipher.init(Cipher.ENCRYPT_MODE,
secretKeySpec);

        byte[] plainBytes =
plainText.getBytes();

        cipherBytes =
cipher.doFinal(plainBytes);
    } catch (IllegalBlockSizeException ex)
{
        System.out.println(ex);
    } catch (BadPaddingException ex) {
        System.out.println(ex);
    } catch (InvalidKeyException ex) {
        System.out.println(ex);
    }
    return cipherBytes;
}
```

```
private String bytesToString(byte[]
rawText) {
    String plainText = null;
    plainText = Base64.encode(rawText);
    return plainText;
}
```

fungsi encrypt akan memanggil dua fungsi lagi, yaitu fungsi encryptText dan byteToString. Karena pesan yang ingin di enkripsi pertama tama harus di jadikan sebuah *array of bytes* (dilakukan oleh fungsi encrytText) dan setelah itu baru di *convert* menjadi sebuah string di fungsi bytesToString.

Setelah membuat metode enkripsi, penulis juga membuat metode deskripsi yang diberi nama decrypt. Berikut adalah metode tersebut.

```
public String decrypt(String cipherText) {
    String plainText = null;
    byte[] cipherBytes =
stringToBytes(cipherText);
    plainText = decryptText(cipherBytes);
    return plainText;
}
```

```
public String decryptText(byte[]
cipherBytes) {
    String plainText = null;
    try {
        cipher.init(Cipher.DECRYPT_MODE,
secretKeySpec);

        byte[] plainBytes =
cipher.doFinal(cipherBytes);

        plainText = new
String(plainBytes);
    } catch (IllegalBlockSizeException ex)
{
        System.out.println(ex);
    } catch (BadPaddingException ex) {
        System.out.println(ex);
    } catch (InvalidKeyException ex) {
        System.out.println(ex);
    }
    return plainText;
}
```

```
private byte[] stringToBytes(String
plainText) {
    byte[] rawText = null;
    try {
        rawText =
Base64.decode(plainText);
    } catch (Base64DecodingException ex) {
        System.out.println(ex);
    }
    return rawText;
}
```

Sama halnya dengan metode enkripsi, metode deskripsi juga memanggil dua metode baru yaitu decryptText dan stringToBytes. Karena alur kerja deskripsi adalah membuat sebuah *array of bytes* terlebih dahulu dari chiperteks yang lalu dijadikan parameter input pada metode decryptText yang akan mengembalikan plainteks dalam bentuk String.

3.4 Masukan dan Keluaran

Setelah penulis berhasil membuat sebuah kelas Blowfish Algorithm, maka kita dapat mencobanya dalam program utama yang sudah penulis buat di awal. Langkah pertama adalah menginisiasi objek dari kelas BlowfishAlgorithm.

```
BlowfishAlgorithm blowfishAlgorithm = new
BlowfishAlgorithm();
```

Setelah objek tersebut di inisialisasi, penulis dapat secara mudah memanggil metode metode yang sudah dibuat di kelas tersebut. Berikut adalah contoh pemanggilan metode-metode tersebut

```
String textToEncrypt = "Saya suka belajar
Matematika Diskrit";
System.out.println("Text before
Encryption: " + textToEncrypt);
String cipherText =
blowfishAlgorithm.encrypt(textToEncrypt);
System.out.println("Cipher Text: " +
cipherText);
System.out.println("Text after Decryption:
" +
blowfishAlgorithm.decrypt(cipherText));
```

Dengan begitu akan diperoleh keluaran program sebagai berikut

```
Text before Encryption: Saya suka belajar Matematika Diskrit
Cipher Text: J1w5oNgQbSmBy584JaPoQ5GlCitW4q4JGPO7/YaDhl0ohfAIKjBG/A==
Text after Decryption: Saya suka belajar Matematika Diskrit

Process finished with exit code 0
```

4. KESIMPULAN

Dengan berkembangnya teknologi aplikasi berbasis android akan sangat banyak berkembang. Oleh karena itu, para pengembang aplikasi memiliki tantangan baru untuk dapat membuat pengguna aplikasi tersebut merasa aman ketika mengirimkan data yang diberikan kepada aplikasi. Karena, seiring perkembangannya teknologi juga membuat banyak orang yang berniat jahat untuk mencuri data data privasi seseorang. Oleh karena itu, Para pengembang aplikasi sudah harus belajar tentang teknik teknik mengenkripsikan data yang memiliki tingkat keamanan tinggi. Algoritma *blowfish* merupakan algoritma yang memiliki tingkat keamanan yang tinggi dan juga sangat mudah diterapkan dalam bahasa pemrograman Java. Yang mana bahasa pemrograman Java adalah bahasa utama yang dipakai untuk mengembangkan aplikasi berbasis android.

REFERENCES

[1] Munir, Rinaldi. 2006. *Matematika Diskrit*. Bandung:Penerbit Informatika

[2] http://www.hukumonline.com/klinik/detail/lt4f235fec78736/dasar-hukum-perlindungan-data-pribadi-pengguna-internet , diakses 25 Nov 2017 pukul 11:41 WIB

[3] https://microcyber2.com/sistem-keamanan-jaringan-komputer/ , diakses pada 25 Nov 2017 pukul 12:29 WIB

[4] http://www.kajianpustaka.com/2014/01/pengertian-sejarah-dan-jenis-kriptografi.html , diakses pada 25 Nov pukul 12:56 WIB

[5] http://www.metode-algoritma.com/2013/02/kriptografi-algoritma-blowfish.html , diakses pada 24 Nov pukul 23:42 WIB

[6] https://www.codepolitan.com/mengenal-dan-memulai-pemrograman-java-belajar-java , diakses pada 27 Nov pukul 23:52 WIB

[7] http://www.theinsanetechie.in/2013/08/java-blowfish-encryption-algorithm.html , diakses pada 28 Nov pukul 12:32 WIB

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 28 Desember 2017

Ahmad Faiz Sahupala
13516065