Integrasi Kriptografi Kunci Publik dan Kriptografi Kunci Simetri

Andrei Dharma Kusuma / 13508009
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
andreidkusuma@yahoo.com

***Abstrak*—Makalah ini akan membahas mengenai bagaimana penulis mencoba mengintegrasikan algoritma kriptografi kunci publik dan algoritma kriptografi kunci simetri. Pembahasan dimulai dengan implementasi algoritma kriptografi kunci publik, implementasi algoritma kunci simetri, dan dilanjutkan dengan perbandingan antara kedua algoritma tersebut. Penulis juga mencoba mengintegrasikan kedua algoritma tersebut menjadi algoritma baru yang diharapkan mampu mengurangi kelemahan dan menambah kelebihan yang dimiliki oleh kedua algoritma tersebut.**

I. PENDAHULUAN

Dewasa ini, komunikasi telah menjadi sebuah hal yang sangat vital bagi kehidupan manusia. Keamanan komunikasi menjadi salah satu aspek yang dipertimbangkan seseorang dalam melakukan komunikasi. Oleh karena itu, kriptografi yang merupakan ilmu keamanan pesan, semakin berkembang dan terus berkembang.

Ada dua tahap yang digunakan dalam kriptografi itu sendiri, yakni tahap enkripsi dan tahap dekrispi. Tahap enkripsi merupakan tahap dimana pengirim pesan merubah seluruh isi pesan dengan menggunakan sebuah kunci menjadi sebuah pesan baru yang isinya sangat berbeda dengan pesan awalnya. Sedangkan tahap dekripsi merupakan tahap dimana penerima pesan mengubah isi pesan yang tidak dapat dimengerti atau pesan yang sudah terenkripsi menjadi pesan aslinya dengan menggunakan sebuah kunci tertentu.

Berkembanglah dua jenis algoritma yang umum digunakan dalam ilmu kriptografi. Pertama ialah dengan algoritma kunci simetri dan kedua ialah dengan algoritma kunci publik. Kedua algoritma ini memiliki kelebihan dan kekurangannya masing-masing. Dimana kelebihan algoritma yang satu menjadi kelemahan di algoritma lainnya, dan kelemahan algoritma yang satu malah menjadi kelebihan algoritma lainnya.

Pada mulanya, kriptografi hanya berupa pertukaran atau substitusi sederhana, dimana pengirim pesan dan penerima pesan memiliki sebuah kunci yang digunakan untuk mendekripsi dan juga mengenkripsi pesan yang dikirimkan. Algoritma ini banyak berkembang dan melahirkan banyak sekali algoritma kunci simetri.

Kemudian kriptografi berkembang lagi dimana si pengirim pesan dan si penerima pesan memiliki sepasang kunci dimana sebuah kunci digunakan untuk mengenkripsi pesan, dan kunci lainnya digunakan untuk mendekripsi pesan.

Kedua algoritma tersebut memiliki kelemahan dan juga kelebihan. Oleh karena itu, penulis mendapatkan ide untuk menggabungkan kedua algoritma tersebut menjadi sebuah kesatuan yang memiliki kelebihan dari masing-masing algoritma di atas dan mengurang kelemahan yang dimiliki oleh masing-masing algoritma di atas.

II. ALGORITMA KUNCI SIMETRI

Algoritma Kunci Simetri merupakan algoritma kriptografi pertama yang berkembang di masyarakat. Pada algoritma ini, ada dua pihak yang terlibat, yaitu si pengirim pesan dan si penerima pesan. Pengirim pesan menggunakan sebuah kunci K untuk mengenkripsi pesan, dan pesan tersebut hanya dapat didekripsi dengan kunci yang serupa

yakni kunci K. Ada banyak sekali algoritma yang terkenal dalam pengimplementasian algoritma kunci simetri ini, yakni :

1. Algoritma Caesar Cipher
2. Algoritma Vigenere Cipher
3. Algoritma Playfair Cipher
4. Algoritma Block Cipher
5. dan lain lain

Dari beberapa contoh diatas, hanya akan dibahas dua algoritma yang akan digunakan nantinya pada makalah ini, yaitu algoritma caesar cipher dan vigenere cipher.

Caesar Cipher

Caesar cipher merupakan sebuah algoritma kriptografi klasik. Operasi yang dilakukan dalam algoritma ini adalah operasi substitusi. Secara matematis, algoritma caesar cipher ialah sebagai berikut :

Enkripsi: $ci = E(pi) = (pi + 3) \bmod 26$
Dekripsi: $pi = D(ci) = (ci - 3) \bmod 26$

Fungsi diatas merupakan fungsi caesar cipher dari 26 karakter alfabet dengan pergeseran 3 untuk setiap karakter aslinya. Contohnya adalah sebagai berikut :

Plainteks:

AWASI ASTERIX DAN TEMANNYA OBELIX

Cipherteks:

DZDVL DVWHULA GDQ WHPDQQBA REHOLA

Karakter hasil enkripsi yang dihasilkan sangat berbeda dari karakter awal. Namun algoritma ini dapat dengan mudah dipecahkan dengan menggunakan cara brute force. Yakni mencoba semua kemungkinan yang ada, karena perhitungan yang dilakukan sangat sederhana.

Vigenere Cipher

Vigenere cipher dapat dibilang merupakan perbaikan dari algoritma caesar cipher. Perbedaan algoritma ini dengan algoritma caesar cipher adalah bahwa pergeseran tiap karakter pada pesan asli merupakan implementasi algoritma caesar cipher dengan key yang berbeda. Secara matematis, algoritma vigenere cipher ialah sebagai berikut :

Kunci: $K = k1k2 \ldots km$

ki untuk $1 \le i \le m$ menyatakan jumlah pergeseran pada huruf ke-i.

$$ci(p) = (p + ki) \bmod 26$$

Dimana untuk setiap karakter ke i pada pesan akan ditambahkan sebanyak kunci ke i sehingga menghasilkan karakter yang baru.

Jika panjang kunci yang ada lebih pendek daripada jumlah plainteks. Maka kunci diulang secara periodic. Misalkan panjang kunci = 20, maka 20 karakter pertama dienkripsi dengan persamaan diatas, dan setiap karakter ke-i menggunakan kunci ki. Hal yang sama berlaku dengan 20 karakter selanjutnya dimana 20 karakter selanjutnya dienkripsi menggunakan kunci yang sama dengan 20 karakter pertama. Karakter k-21 dienkripsi dengan kunci ke-1, dan selanjutnya.

Contohnya adalah sebagai berikut :

Kunci : sony

Plainteks : THIS PLAINTEXT
Kunci : sony sonysonys
Cipherteks : **LVVQ HZNGFHRVL**

Terlihat bahwa untuk sebuah karakter pesan awal dapat memiliki kemungkinan karakter cipherteksnya. Namun serangan masih dapat dilakukan terhadap algoritma ini dengan melakukan analisis frekuensi. Misalkan untuk bahasa inggris, kata THE sangat sering ditemui, dan dengan analisis pola tersebutlah algoritma ini masih dapat terpecahkan.

III. ALGORITMA KUNCI PUBLIK

Sampai akhir tahun 1970, hanya ada sistem kriptografi kunci-simetri. Satu masalah besar dalam sistem kriptografi:

Bagaimana mengirimkan kunci rahasia kepada penerima?

Mengirim kunci rahasia pada saluran publik (telepon, internet, pos) sangat tidak aman. Oleh karena itu, kunci harus dikirim melalui saluran kedua yang benar-benar aman. Saluran kedua tersebut umumnya lambat dan mahal.

Kriptografi kunci-nirsimetri disebut juga

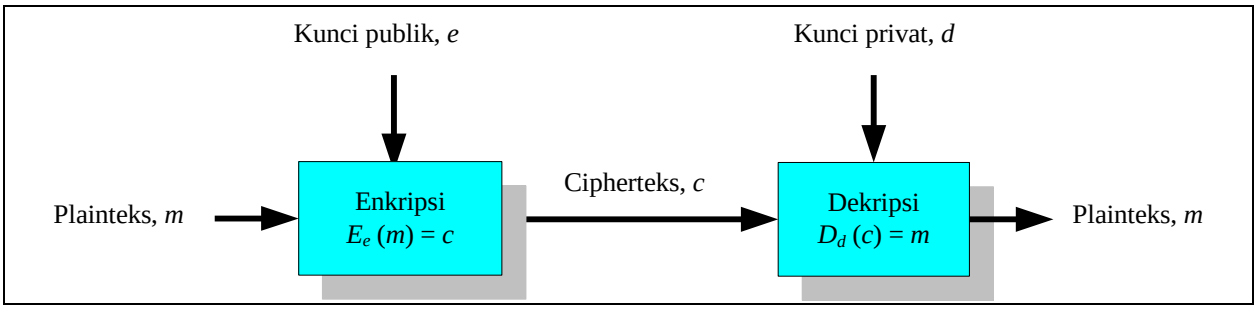
kriptografi kunci-publik. Pada kriptografi kunci-publik, masing-masing pengirim dan penerima mempunyai sepasang kunci:

1. Kunci publik: untuk mengenkripsi pesan
2. Kunci privat: untuk mendekripsi pesan.

Fungsi enkripsi dan dekripsinya ialah sebagai berikut :

$Ee(m) = c$
$Dd(c) = m$

Gambar dibawah ini menggambarkan proses yang terjadi dalam algoritma kunci publik.



Ada beberapa macam algoritma kunci public yang ada sekarang ini, yakni :

1. Algoritma RSA
2. Algoritma ElGamal
3. Algoritma Diffie-Hellman
4. Algoritma DSA
5. dan lain lain

Namun pada makalah ini akan digunakan dua jenis algoritma saja, yaitu algoritma Diffie-Hellman dan algoritma RSA.

Diffie-Hellman

Algoritma Diffie-Hellman merupakan algoritma yang berguna untuk berbagi kunci enkripsi simetri yang sama antara dua orang atau lebih. Keamanan algoritma ini ditentukan oleh sulitnya menghitung logaritma diskrit.

Parameter yang digunakan dalam algoritma ini ialah sebagai berikut :

- Misalkan dua orang yang berkomunikasi: Alice dan Bob.
- Mula-mula Alice dan Bob menyepakati bilangan prima yang besar, n dan g, sedemikian sehingga $g < n$.
- Bilangan n dan g tidak perlu rahasia. Bahkan, Alice dan Bob dapat membicarakannya melalui saluran yang tidak aman sekalipun.

Sedangkan algoritma yang digunakan ialah sebagai berikut :

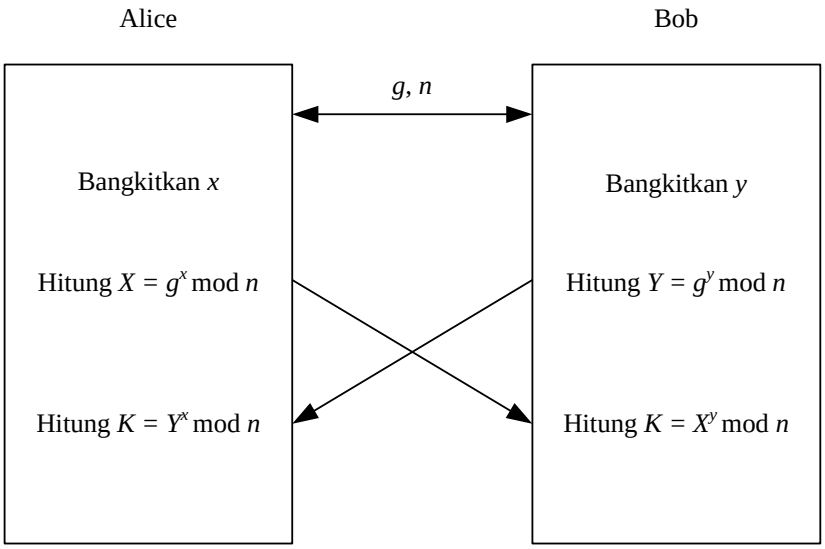
1. Alice membangkitan bilangan bulat acak yang besar x dan mengirim hasil perhitungan berikut kepada Bob:
   $X = gx \bmod n$
2. Bob membangkitkan bilangan bulat acak yang besar y dan mengirim hasil perhitungan berikut kepada Alice:
   $Y = gy \bmod n$
3. Alice menghitung
   $K = Yx \bmod n$
4. Bob menghitung
   $K' = Xy \bmod n$

Jika perhitungan dilakukan dengan benar, maka didapatkan

$$K = K'.$$

Baik K dan K' sama dengan $gxy \bmod n$. Eve yang menyadap pembicaraan antara Alice dan Bob tidak dapat menghitung K. Ia hanya memiliki informasi n, g, X dan Y, tetapi ia tidak mempunyai informasi nilai x dan y.

Untuk mengetahui x atau y, ia perlu melakukan perhitungan logaritma diskrit, yang mana sangat sulit dikerjakan.



Algoritma RSA

Algoritma kunci-publik yang paling terkenal dan paling banyak aplikasinya. Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima.

Berikut ialah property dalam algoritma RSA:

1.	p dan q bilangan prima	Rahasia
2.	n = p . q	Tidak rahasia
3.	$\phi(n) = (p-1)(q-1)$	Rahasia
4.	e (kunci enkripsi) Syarat: PBB(e, $\phi(n)$) = 1	Tidak rahasia
5.	d (kunci dekripsi) d dihitung dari $d \equiv e\text{-}1 \bmod (\phi(n))$	Rahasia
6.	m (plainteks)	Rahasia
7.	c (cipherteks)	Tidak Rahasia

Algoritma yang digunakan ialah sebagai berikut, untuk proses enkripsi :

1. Nyatakan pesan menjadi blok-blok plainteks: $m1$, $m2$, $m3$, … (syarat: $0 < mi < n-1$)

2. Hitung blok cipherteks ci untuk blok plainteks pi dengan persamaan
   $ci = mie$ **mod** n
   yang dalam hal ini, e adalah kunci publik.

Dan untuk proses dekripsi sebagia berikut :

1. Proses dekripsi dilakukan dengan menggunakan persamaan

   $mi = cid$ **mod** n,

   yang dalam hal ini, d adalah kunci privat.

IV. INTEGRASI ALGORITMA KUNCI PUBLIK DAN SIMETRI

1. Diffie-Hellman –Kunci Simetri

Integrasi

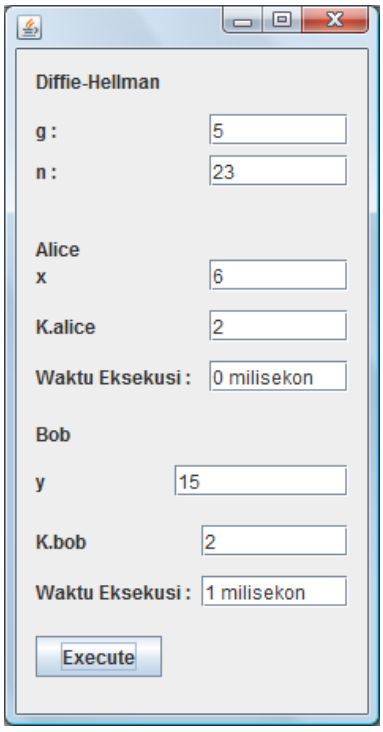
Integrasi algoritma Diffie-Hellman dan algoritma kunci simetri apapun sebenarnya dapat terlihat dengan jelas. Dalam kasus ini kita menggunakan salah satu algoritma kunci simetri yaitu algoritma caesar cipher sederhana. Algoritma dalam integrasi algoritma Diffie-Hellman – Caesar Cipher ialah sebagai berikut :

1. Pengirim pesan dan penerima pesan memilih dua bilangan prima yang besar dimana g < n
2. Kemudian masing-masing dari pengirim dan penerima pesan memilih 1 bilangan prima yang besar.
3. Kemudian dilakukan komputasi dengan menggunakan algoritma Diffie-Hellman.
4. Setelah komputasi berjalan dengan benar, pengirim dan penerima masing-masing menerima kunci simetri yang sama.
5. Dengan kunci tersebut, pesan dienkripsi dengan menggunakan algoritma caesar cipher dengan kunci hasil pertukaran dengan algoritma Diffie-Hellman

Implementasi

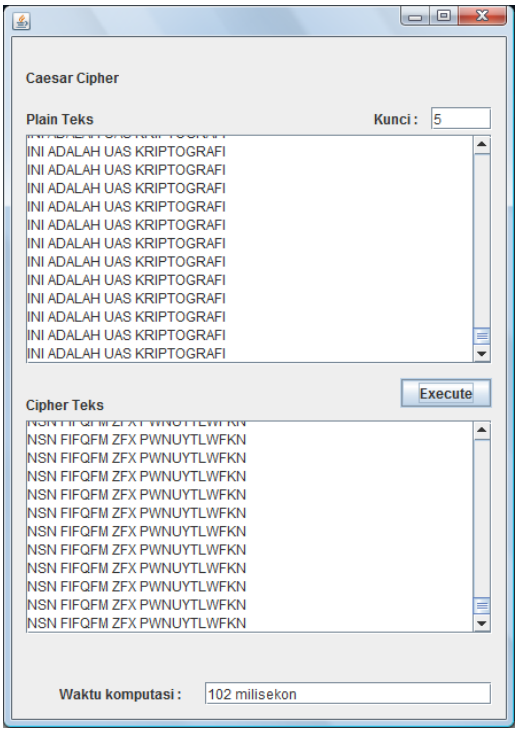
Berikut merupakan implementasi algoritma dan perbandingan kecepatan komputasi dengan dan tanpa algoritma diffie-hellman :

```
public int Diffie_Hellman_pub (String n, String g, int z){
// untuk membuat nilai X dari x

        BigInteger N = new BigInteger(n);
        BigInteger G = new BigInteger(g);

        BigInteger Result = G.pow(z);
        Result = Result.mod(N);
        int temp = Result.intValue();
        return temp;
    }

    public int Diffie_Hellman_key (String n, String O, int z){
// untuk menghasilkan kunci

        int temp;

        BigInteger o = new BigInteger(O);
        BigInteger N = new BigInteger(n);

        BigInteger Result = o.pow(z);
        Result = Result.mod(N);
        temp = Result.intValue();
        return temp;
    }
```



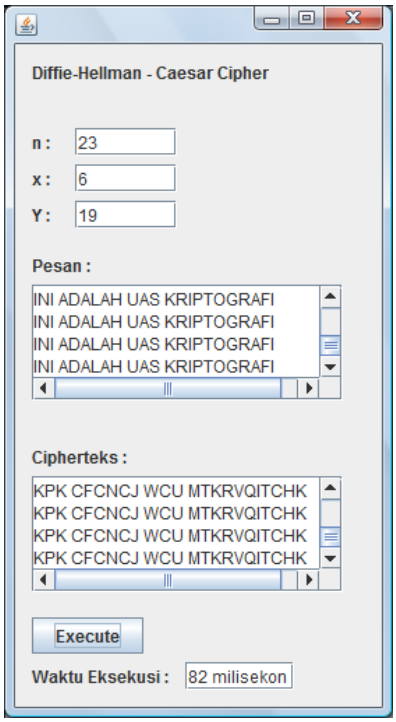
Berikut ialah lama komputasi 150 teks INI ADALAH UAS KRIPTOGRAFI dengan menggunakan caesar cipher dengan kunci 5.

```
public String Caesar_Cipher(String plainteks, int kunci){

        String cipherteks = "";
        String temp = plainteks.toUpperCase();
        int temp1;

        for (int i = 0; i< plainteks.length(); i++){
            temp1 = (int)temp.charAt(i);
            if(temp1 <= 90 && temp1>=65){
                temp1= temp1 + kunci;
                if(temp1 > 90){
                    temp1 = temp1 - 26;
                }
            }

            cipherteks = cipherteks + (char)temp1;
        }

        return cipherteks;
    }
```



Dan integrasi kedua algoritma tersebut ialah hanya dengan menggunakan fungsi dibawah ini, dan berikut implementasinya

```
Caesar_Cipher(plainteks,
Diffie_Hellman_key(n, X, y));
```



Analisis

Waktu komputasi yang dihasilkan ternyata lebih singkat dibandingkan dengan algoritma caesar cipher sederhana. Namun hal ini dikarenakan semata bahwa kunci yang dihasilkan merupakan bilangan yang sangatkecil yakni angka 2.

Namun algoritma ini akan sangat aman dari segi keamanan kunci karena algoritma diffie-hellman mustahil untuk dipecahkan karena sulitnya menghitung logaritma diskrit.

Walaupun demikian, masih ada satu lubang besar yang ada dalam algoritma ini yaitu masalah keamanan algoritma. Algoritma kunci simetri merupakan algoritma yang rentan terhadap serangan. Sudah banyak trik-trik yang digunakan dalam mendekripsi sebuah cipher teks, misalnya dengan analisis kemunculan huruf pada caesar dan vigenere cipher. Dan integrasi algoritma Diffie-Hellman dan algoritma kunci simetri tidak dapat mengurangi masalah keamanan tersebut. Berikut adalah tabel perbandingan ketiga algoritma

Algoritma	Diffie-Hellman	Kunci Simetri	Integrasi
Kekurangan	-	1. Pertukaran kunci rentan terhadap serangan 2. Algoritma mampu dipecahkan dengan beberapa cara	1. Algoritma tetap dapat dipecahkan tanpa mengetahui kunci
Kelebihan	1. Sangat aman 2. Waktu komputasi tidak terlalu lama	1. Waktu komputasi cepat	1. Waktu komputasi relatif cepat 2. Pertukaran kunci sangat aman

Tabel perbandingan Algoritma Diffie-Hellman, Kunci Simetri, dan integrasi keduanya

2. Algoritma 50%

Algoritma ini merupakan algoritma racikan penulis dimana tujuan dari algoritma ini membagi rata semua kekurangan yang dimiliki oleh algoritma kunci publik dan algoritma kunci simetri.

Ada dua kelemahan yang menjadi dasar kunci simetri, pertama ialah bagaimana pertukaran kunci dilakukan dan yang kedua ialah bagaimana algoritma kunci simetri tidak dapat dipecahkan oleh penyerang. Masalah pertama merupakan masalah yang paling realistis untuk kita selesaikan sedangkan masalah kedua merupakan masalah yang sulit untuk dipecahkan dengan algoritma kunci simetri. Oleh karena itu, dalam algoritma 50% ini masalah pertukaran kunci dapat sepenuhnya terselesaikan dan kerapuhan algoritma kunci simetri akan diobati dengan algoritma kunci publik.

Algoritma kunci publik berguna untuk

meningkatkan keamanan dari pesan tersebut, seberapa lama waktu yang dibutuhkan oleh penyerang untuk mendekripsi pesan yang terenkripsi. Namun, kendala utama ialah dari proses perhitungan yang relatif rumit dan lama. Oleh karena itu, kita akan menggunakan kelebihan kunci simetri dimana perhitungan yang dilakukan tidaklah rumit dan dapat dengan cepat diselesaikan.

Pada pembahasan ini, algoritma yang akan dipakai ialah algoritma vigenere cipher untuk algoritma kunci simetri, dan RSA untuk algoritma kunci publik.

Integrasi

Algoritma dari algoritma 50% ialah sebagai berikut. Kita memiliki dua jenis metode enkripsi. Pertama ialah proses enkripsi dengan vigenere cipher, dan kedua ialah proses enkripsi dengan algoritma RSA. Oleh karena itu dibutuhkan beberapa kunci untuk melakukan proses enkripsi tersebut. Untuk proses enkripsi dengan vigenere cipher, kita membutuhkan satu buah kunci K. Sedangkan untuk proses enkripsi dengan algoritma RSA, kita membutuhkan dua kunci, yaitu N sebagai perkalian antara dua buah bilangan prima yang besar, dan E sebagai kunci publik dimana E relatif prima terhadap N. Ada 3 kunci yang dibutuhkan dalam proses enkripsi.

Dalam algoritma kunci publik, E dan N dapat disebarkan dengan bebas, dengan penerima pesan memiliki D, yakni kunci privat, yang digunakan untuk mendekripsi pesan. Sedangkan untuk kunci simetri K, kunci ini akan kita masukkan kedalam pesan yang akan dikirim nantinya, dengan K dienkripsi menggunakan algoritma RSA dan ditempelkan di depan pesan yang dikirimkan.

Tahap algoritmanya ialah sebagai berikut :

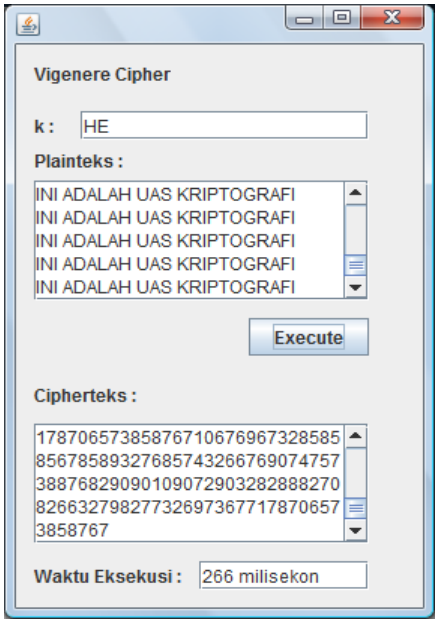
1. Asumsikan kita telah memiliki 4 nilai diatas, yakni K, E, D, dan N. Dan sebuah pesan P yang nantinya akan di enkripsi menjadi pesan C.
2. Pertama-tama kita bagi pesan tersebut kedalam blok-blok yang berukuran X bit, dimana X dapat dipilih oleh pengguna.
3. Kemudian untuk blok yang ganjil, dilakukan proses enkripsi menggunakan algoritma RSA, hal ini ditujukan agar blok pertama sebagai header pesan dapat terenkrispi dengan baik dan keamanan header dapat terjaga. Proses enkripsi dilakukan seperti biasa menggunakan kunci E dan N.
4. Kemudian untuk blok yang genap, dilakukan prosesn enkripsi menggunakan algoritma vigenere cipher. Proses enkripsi ini dilakukan dengan menggunakan kunci K.
5. Setelah semua proses enkripsi telah selesai dilakukan, kunci K dipersiapkan untuk di enkripsi menggunakan algoritma RSA, lebarkan kunci tersebut hingga memiliki panjang 1 blok. Pelebaran kunci tersebut dilakukan seperti halnya perulangan kunci pada vigenere cipher. Setelah 1 blok kunci tersebut di enkripsi, hasil enkripsi ditempelkan pada bagian awal pesan.
6. Kemudian gabungkan kembali blok-blok tersebut hingga membentuk urutan sesuai dengan pesan semula.
7. Pesan selesai di enkripsi.

Implementasi

Berikut adalah 150 teks INI ADALAH UAS KRIPTOGRAFI yang dienkripsi menggunakan ketiga algoritma dan waktu eksekusi yang dibutuhkan oleh masing-masing algoritma. Hasil cipherteks yang ditampilkan ialah berupa bilangan ASCII karakter hasil enkripsi.

```
public String VC (String plainteks, String k){

        String cipherteks = "";

        String plain = plainteks.toUpperCase();

        String kunci= "";
        int j = 0;

        for(int i = 0; i<plainteks.length();i++){
           if(j>=k.length())
              j = j-k.length();

           kunci = kunci + k.toUpperCase().charAt(j);
           j++;
        }

        int temp;

        for(int i = 0; i< plain.length();i++){

           if((int)plain.charAt(i) <=90 && (int)plain.charAt(i)>=65){
```

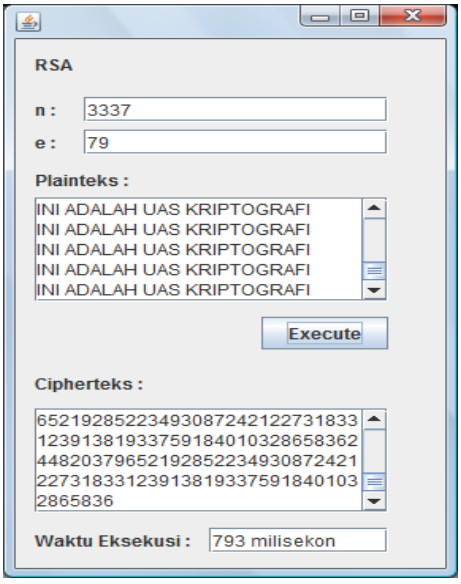
```
                temp = (int)plain.charAt(i) + (int)kunci.charAt(i);
                while(temp>90)
                    temp = temp - 26;
            }
            else{
                temp = (int)plain.charAt(i);
            }
                cipherteks = cipherteks + temp;
        }

        return cipherteks;
    }
```



Berikut ialah implementasi algoritma RSA :

```
    public String RSA (String plainteks, int p, int n){

        String cipherteks ="";
        BigInteger N = new BigInteger(""+n);

        String temp ="";
        for (int i = 0; i<plainteks.length();i++){
            temp = temp + (int)plainteks.charAt(i);
        }

        String[] blok = new String[temp.length()/3 + 1];
        int j = 0;
        for(int i = 0; i< (temp.length()/3);i++){
            blok[i] = "";
            if(j < temp.length())
                blok[i] = blok[i] + temp.charAt(j);
            j++;
            if(j < temp.length())
```

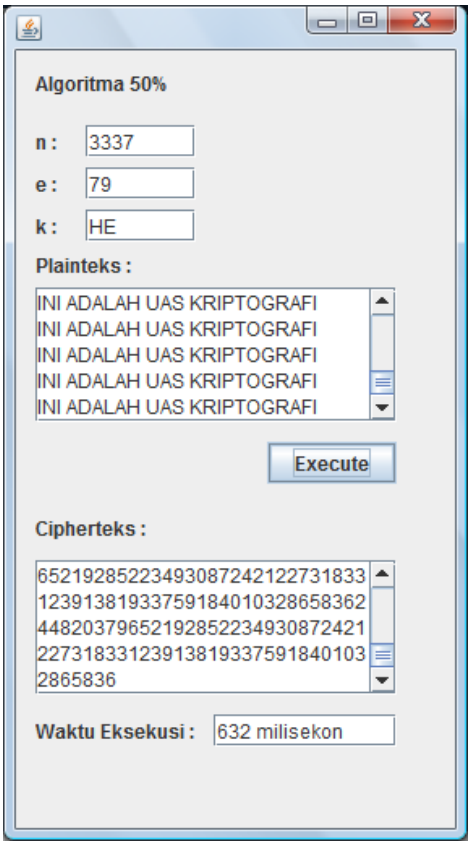
```
                blok[i] = blok[i] + temp.charAt(j);
            j++;
            if(j < temp.length())
                blok[i] = blok[i] + temp.charAt(j);
            j++;
            System.out.println(blok[i]);
        }
        BigInteger cipherblok = BigInteger.ZERO;
        BigInteger result = BigInteger.ZERO;
        for(int i = 0; i<blok.length && blok[i]!=null;i++){
            cipherblok = new BigInteger(blok[i]);
            result = cipherblok.pow(p);
            result = result.mod(N);
            cipherteks = cipherteks + result;
        }


        return cipherteks;
    }
```



Berikut ialah implementasi dari algoritma 50% :

```
    public String limapuluhpersen(String plainteks, String k, int p, int n){
        String cipherteks = "";

        String kunci = "";
        int j = 0;
        for(int i = 0; i < 12; i++){
            if(j >= k.length())
                j= j-k.length();
            kunci = kunci + k.toUpperCase().charAt(j);
            j++;
```

```
        }


        cipherteks = RSA(kunci, p, n) + " ";

        String[] blok = new
String[plainteks.length()/12];

        int i = 0;
        for(j = 0; j<plainteks.length()/12;j++){

            for(int l = 0; l<12;l++){
                blok[j] = blok[j] +
plainteks.toUpperCase().charAt(i);
                i++;
            }
        }

        j=0;
        for(i = 0; i<blok.length;i++){
            if(i%2 == 0 || i==0){
                cipherteks = cipherteks +
RSA(blok[i], p, n)+ " ";
            }
            else{
                cipherteks = cipherteks +
VC(blok[i], k)+" ";
            }
        }

        return cipherteks;
    }
```



Analisis

Terlihat bahwa waktu eksekusi yang dihasilkan jauh lebih cepat dari pada waktu ekseskusi dengan menggunakan algoritma RSA, dan tentunya dengan cara ini keamanan kunci dapat terjamin. Hal ini merupakan keunggulan yang diberikan oleh algoritma 50% ini.

Kunci simetri juga dapat secara aman dipertukarkan dan ditempelkan dalam pesan, selain itu algoritma ini akan menjadi sangat sulit untuk dipecahkan karena akan membutuhkan waktu lebih lama untuk memecahkan vigenere cipher dengan panjang text terbatas daripada vigenere cipher dengan panjang text yang panjang dan tentunya merupakan hal yang sulit untuk memecahkan algoritma kunci publik karena operasi perpangkatan yang sangat besar.

Berikut ialah tabel perbandingan ketiga algoritma.

Algoritma	Vigenere Cipher	RSA	Integrasi / 50%
Kekurangan	1. Lemah terhadap serangan 2. Pertukaran kunci tidak aman	1. Waktu komputasi sangat lama 2. Rumitnya komputasi yang harus dilakukan	1. Rumitnya waktu komputasi yang harus dilakukan
Kelebihan	1. Waktu komputasi cepat	1. Aman dan sulit untuk dipecahkan 2. Serangan terhadap kunci tidak perlu diperhatikan	1. Waktu komputasi menjadi lebih cepat 2. Pertukaran kunci simetri menjadi sangat aman 3. Algoritma kuat akan serangan karena sulitnya memecahkan vigenere cipher dengan panjang teks yang pendek

Tabel Perbandingan algoritma Vigenere Cipher, RSA, dan 50%

V. KESIMPULAN

Kesimpulan makalah ini ialah walaupun integrasi antara algoritma Diffie-Hellman dan algoritma kunci simetri sangat ampuh dalam mengatasi keamanan pertukaran kunci, namun integrasi tersebut masih rapuh terhadap serangan-serangan karena lemahnya algoritma yang dimiliki oleh kunci simetri. Sedangkan algoritma 50% yang penulis kembangkan dapat mengurangi hampir sebagian besar kelemahan yang dimiliki oleh kedua algoritma tersebut dengan hasil yang cukup optimal.

DAFTAR PUSTAKA

[1] http://informatika.org/~rinaldi/

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, 9 Mei 2011

Andrei Dharma Kusuma / 13508009