ANALISIS DAN STUDI ALGORITMA PERTUKARAN KUNCI DIFFIE-HELLMAN

Tommy Gunardi (13507109)

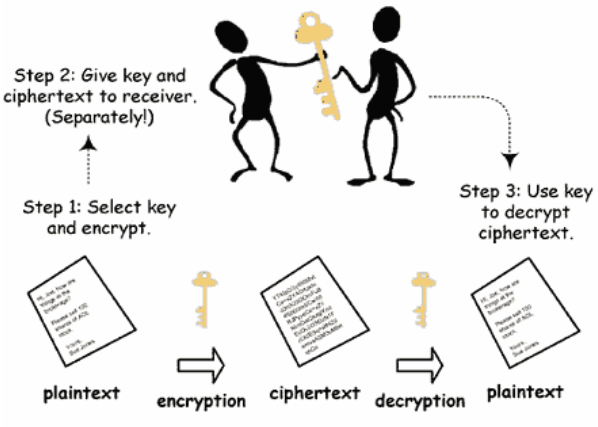
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
if17109@students.if.itb.ac.id

***Abstract*—Dewasa ini, teknik kriptografi sangat berkembang pesat. Semakin banyak teknik kriptografi yang dirancang demi terciptanya suatu teknik yang sempurna tanpa bisa dipecahkan pihak ketiga. Untuk memenuhi hal tersebut, dirancanglah dua macam teknik kunci kriptografi, yaitu teknik kriptografi kunci simetrik dan asimetrik. Kedua algoritma ini memiliki kelebihan dan kekurangannya masing-masing. Perbedaan antara keduanya akan dijelaskan secara gamblang pada isi paper ini. Algoritma kunci simetrik menggunakan sebuah kunci privat rahasia untuk enkripsi dan dekripsi. Sedangkan algoritma kunci asimetrik digunakan untuk membuat pasangan kunci yang berhubungan secara matematis, kunci publik yang diumumkan dan kunci privat rahasia. Penggunaan kunci ini memroteksi otentikasi dari sebuah pesan dengan membuat tanda tangan digital pada pesan tersebut. Salah satu teknik kriptografi kunci asimetrik yang akan dibahas pada paper ini adalah algoritma pertukaran kunci Diffie-Hellman. Algoritma ini memperbolehkan dua pihak yang tidak saling mengenal satu sama lain dapat menukarkan kunci publik masing-masing dalam saluran yang tidak aman untuk menghasilkan sebuah kunci yang dapat dipakai bersama-sama. Kunci inilah yang nantinya dapat digunakan sebagai kunci enkripsi.**

***Index Terms*—*kriptografi, pertukaran kunci, Diffie-Hellman.*

I. INTRODUCTION

Seperti yang telah disebutkan pada abstrak, ada dua macam pendekatan kriptografi yaitu kriptografi kunci simetrik dan asimetrik. Kedua kriptografi ini berbeda dalam hal penggunaan kunci. Kriptografi kunci simetrik mengacu pada metode enkripsi dimana kunci enkripsi dan dekripsi sama. Pengirim dan penerima pesan berbagi kunci yang sama. Studi modern kunci simetrik biasanya digunakan pada aplikasi *block cipher* dan *stream cipher*. *Block cipher* adalah algoritma kriptografi yang beroperasi pada plainteks / cipherteks dalam bentuk blok bit, dalam hal ini rangkaian bit dibagi menjadi blok-blok bit yang panjangnya sudah ditentukan sebelumnya. Misalkan panjang blok adalah 64 bit, berarti algoritma enkripsi memperlakukan 8 karakter setiap kali penyandian(1 karakter = 8 bit dalam pengodean ASCII). *Stream cipher* adalah algoritma kriptografi yang beroperasi pada plainteks/cipherteks dalam bentuk bit tunggal, yang dalam hal ini rangkaian bit dienkripsikan/didekripsikan per bit. *Data Encription Standard* (DES) dan *Advanced Encryption Standard* (AES) adalah suatu bentuk desain cipher block yang telah ditetapkan sebagai standar kriptografi oleh pemerintah Amerika Serikat. Tipe DES masih cukup populer digunakan di berbagai aplikasi, dari enkripsi ATM untuk privasi email dan akses jarak jauh.



Gambar 1 Algoritma kunci simetrik

Beberapa kelebihan dan kekurangan kriptografi simetri adalah sebagai berikut

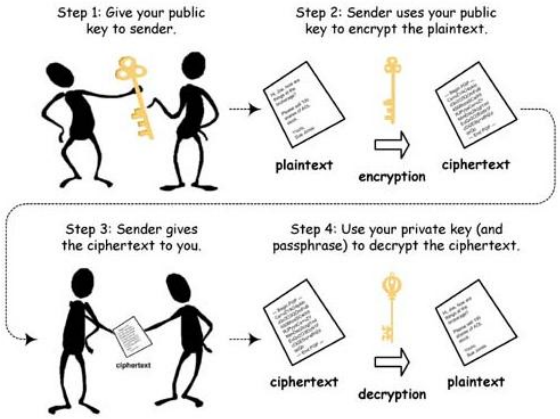
Kelebihan :

- Algoritma kriptografi simetri dirancang sehingga proses enkripsi/dekripsi membutuhkan waktu yang singkat
- Ukuran kunci simetri relatif pendek
- Algoritma kriptografi simetri dapat digunakan untuk membangkitkan bilangan acak
- Algoritma kriptografi simetri dapat disusun untuk menghasilkan cipher yang lebih kuat
- Otentikasi pengirim pesan langsung diketahui dari cipherteks yang diterima, karena kunci hanya diketahui pengirim dan penerima pesan saja

Kekurangan:

- Kunci simetri harus dikirim melalui saluran yang aman. Kedua entitas yang berkomunikasi harus menjaga kerahasiaan kunci ini.
- Kunci harus sering diubah, mungkin pada setiap sesi komunikasi untuk menjaga keamanan pesan.

Pada sebuah makalah yang inovatif pada tahun 1976, Whitfield Diffie dan Martin Hellman mengusulkan suatu gagasan mengenai kunci asimetrik. Ide dari algoritma kunci asimetrik ini adalah menggunakan dua macam kunci yang berhubungan secara matematis namun berbeda. Kedua kunci tersebut disebut kunci publik dan kunci privat. Dalam penggunaannya terdapat macam-macam cara untuk mengenkripsi pesan. Kadang untuk kasus yang berbeda kita dapat menggunakan kunci publik atau kunci privat untuk enkripsi. Untuk RSA contohnya, kita menggunakan kunci publik untuk mengenkripsi dan kunci privat untuk mendekripsi. Untuk kasus lainnya, contohnya tanda tangan digital, kita menggunakan kunci privat untuk mengenkripsi, dan kunci publik untuk mendekripsi. Pemanfaatan kunci publik dan kunci privat tersebut bisa bermacam-macam. Salah satunya dipakai dalam metoda Diffie-Hellman untuk menghasilkan suatu kunci baru lagi hasil integrasi dari beberapa kunci.



Gambar 2 Algoritma Kunci Asimetrik

Beberapa kelebihan dan kekurangan algoritma kunci asimetrik adalah sebagai berikut

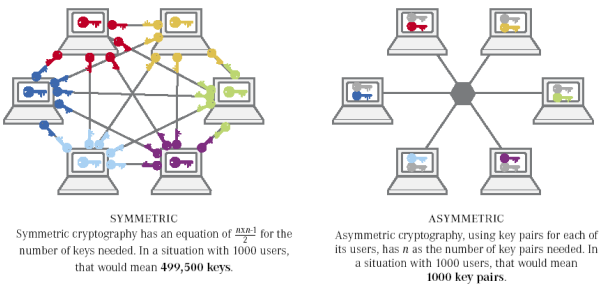
Kelebihan:

- Hanya kunci privat yang perlu dijaga kerahasiaannya oleh setiap entitas yang berkomunikasi. Namun tentu saja otentikasi terhadap kunci publik harus tetap terjamin.
- Pasangan kunci publik / privat tidak perlu diubah, bahkan dalam periode waktu yang panjang.
- Dapat digunakan untuk mengamankan pengiriman kunci simetri.
- Beberapa algoritma kunci publik dapat digunakan untuk memberi tanda tangan digital pada pesan.

Kekurangan:

- Enkripsi dan dekripsi data umumnya lebih lambat dari sistem simetri, karena enkripsi dan dekripsi menggunakan bilangan yang besar dan melibatkan operasi perpangkatan yang besar.
- Ukuran cipherteks lebih besar daripada plainteks (dua sampai emat kail).
- Karena kunci publik diketahui secara luas dan dapat digunakan setiap orang, maka cipherteks tidak memberikan informasi mengenai otentikasi pengirim.
- Tidak ada algoritma kunci publik yang terbukti aman (sama seperti block cipher). Kebanyakan algoritma mendasarkan keamanannya pada sulitnya memecahkan persoalan-persoalan asimetriks yang menjadi dasar pembangkitan kunci.

Karena algoritma kunci simetrik menggunakan kunci yang sama untuk mengenkripsi dan dekripsi pesan, kita memerlukan manajemen kunci yang baik agar dapat menggunakannya dngan aman. Kedua belah pihak harus berkomunikasi setiap cipherteks yang berbeda. Jumlah kunci yang diperlukan meningkat seiring bertambahnya jumlah anggota jaringan yang membutuhkan manajemen kunci yang kompleks dan sangat cepat untuk menjaga mereka semua tetap rahasia. Kesulitan lain saat kita akan memutuskan menggunakan suatu kunci, namun saluran komunikasi antara mereka sangat tidak aman. Hal-hal tersebut menjadi kendala bagi pengguna kriptografi di dunia nyata.



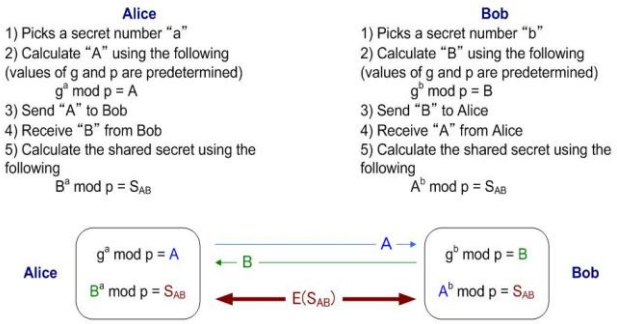
Gambar 3 Pembagian kunci simetrik dan asimetrik

Algoritma kunci asimetrik dianggap dapat memenuhi kebutuhan akan hal tersebut. Kita hanya perlu memiliki dua buah kunci yaitu kunci privat masing-masing dan kunci publik untuk di share. Algoritma kunci asimetrik ini memiiliki beberapa contoh aplikasi seperti Diffie-Hellman, Digital Signature, dan RSA. Salah satu algoritma yang akan dieksplorasi lebih jauh adalah algoritma pertukaran kunci Diffie-Hellman. Kita akan melihat lebih jauh bagaimana sistem kerja algoritma ini dan bagaimana kita menghasilkan kunci enkripsi dari pertukaran kunci antara dua orang atau lebih. Penjelasan tentang algoritma pertukaran kunci Diffie-Hellman akan dijelaskan lebih lanjut pada dasar teori.

II. METODA PERTUKARAN KUNCI DIFFIE-HELLMAN

A. DASAR TEORI

Algoritma Diffie-Hellman adalah sebuah protokol kriptografi yang memungkinkan dua pihak yang tidak saling mengenal dapat untuk menggabungkan kunci mereka menjadi sebuah kunci untuk dipakai dalam proses enkripsi dekripsi. Algoritma ini diciptakan oleh Whitfield Diffie dan Martin Hellman pada tahun 1976. Algoritma ini diciptakan sebagai metoda praktek pertama untuk menetapkan suatu *shared secret* melalui saluran komunikasi yang tidak dilindungi.



Gambar 4 Skema Algoritma Diffie-Hellman

Langkah-langkah pertukaran kunci Diffie-Hellman adalah sebagai berikut :

1. Pertama-tama Alice dan Bob menyetujui untuk menggunakan dua buah angka "g" dan "n" dimana nilai "g" harus lebih kecil dari "n" dan keduanya merupakan angka prima.
2. Alice lalu memilih suatu angka rahasia "a" atau kita sebut dengan secret key Alice. Bob juga memilih suatu angka rahasia "b" atau kita sebut dengan secret key Bob.
3. Alice menghitung nilai A yang dihasilkan dengan rumus A = g^a mod n. Bob juga mengkalkulasi nilai B yang dihasilkan dengan rumus B = g^b mod n. Nilai A dan B inilah yang merupakan kunci publik yang nantinya akan dipertukarkan.
4. Alice memberikan nilai A kepada Bob dan sebaliknya Bob memberikan nilai B kepada Alice.
5. Selanjutnya Alice menghitung nilai kunci enkripsi K dimana K = B^a mod n. Sebegitu pula dengan Bob menghitung nilai K` dimana K` = A^b mod n.

Setelah proses tersebut akan dihasilkan nilai K untuk Alice dan nilai K` untuk Bob. Jika hasil komputasi dan pertukaran nilai memenuhi ketentuan di atas, maka nilai K dan K` adalah sama.

B. SIMULASI DIFFIE-HELLMAN

Untuk mencoba menyimulasikan Algoritma Diffie-Hellman, penulis membuat program untuk pertukaran kunci dan penentuan kunci enkripsi. Program dibuat dalam bahasa C# dengan kakas Microsoft Visual Studio 2008. Source code program dilampirkan pada bagian akhir makalah.

Kasus eksekusi program adalah sebagai berikut :

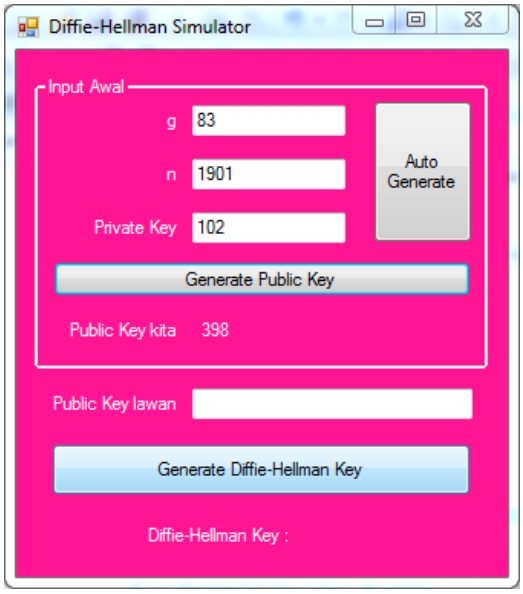
1. Untuk lebih mudah membedakan program simulasi milik Alice dan Bob, interface Diffie-Hellman Simulator milik mereka dibedakan warnanya. Alice dan Bob membuka program dan menyetujui bilangan prima g = 83 dan n = 1901. Keduanya lalu memilih kunci privat masing-masing, Kunci privat Alice = 102 dan kunci privat Bob = 91.

2. Lalu keduanya mengkalkulasi nilai kunci publik masing-masing dan kemudian memberikannya satu sama lain. Untuk kasus ini Kunci publik keduanya adalah :

Kunci Publik Alice

g = 83
n = 1901
kunci privat Alice = 102

kunci publik Alice
= g^a mod n
= 83^{102} mod 1901
= 55714138870550830693049304100719644252118268465539101561441961274937004935346912051106907241202439320582718094621764116343669631478325533447846570270447904093949747174128023677592957548172128488889 mod 1901
= 398



Gambar 5 interface Alice

Kunci Publik Bob

g = 83
n = 1901
kunci privat Bob = 91

kunci publik Bob
= g^b *mod* n
= 83^{91} *mod* 1901
= 43261981396444057909120141329404775803281629956388173023621630928911900262378068625483019134371742546551490144501647314962844045203631189367170363159108920510916232129438570671598

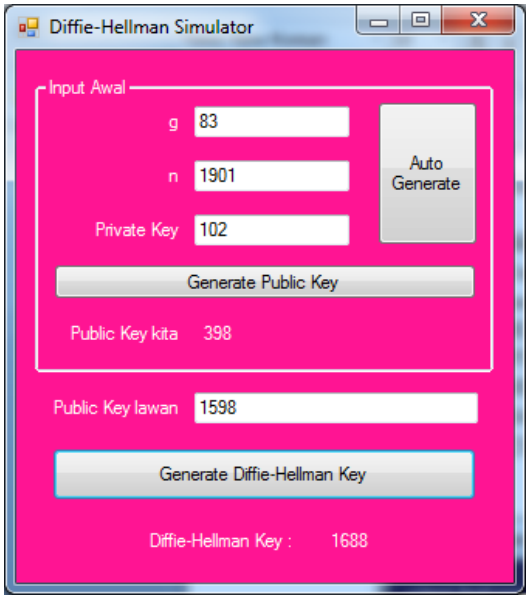
Gambar 6 interface Bob

3. Selanjutnya Alice dan Bob saling menukar kunci publik mereka. Sehingga Alice memiliki kunci publik Bob dan sebaliknya. Lalu Alice dan Bob kembali mengkalkulasi nilai kunci enkripsi yang akan mereka sepakati akan digunakan. Kunci enkripsinya adalah sebagai berikut :

Kunci Enkripsi Alice

g = 83
n = 1901
kunci privat Alice (a) = 102
kunci publik Alice (A) = 398

kunci publik Bob (B) = 1598
kunci enkripsi Alice (K)
= B^a *mod* n
= 1598^{102} *mod* 1901
= 58187680798869352435433146152844877846921272212576448816900168760041463805453708609356366269398575418896287961760349098066930992546346323360480191525573637703651080216302749369177138279238935633654007876877906799423231792625980666353757706715605665538745535524497361250411824530714216215001532311816089003581748779819246551048 *mod* 1901
= 1688



Gambar 7 hasil Diffie-Hellman Key Alice

Kunci Enkripsi Bob

g = 83
n = 1901
kunci privat Bob (b) = 91
kunci publik Bob (B) = 1598
kunci publik Alice (A) = 398

kunci enkripsi Bob (K`)
= A^b *mod* n
= 398^{91} *mod* 1901
= 38847185437695793073742204018633017082350642396563377059419096309178804834266023951026978271905966338198323449602022678725272316203260408052668923241594256619003953631267619394535445576188469463879702959538003161608123999129134878110515 2 *mod* 1901
= 1688

Gambar 8 hasil kunci Diffie-Hellman Bob

4. Ditemukan bahwa nilai K dan K` adalah 1688. Nilai inilah yang nantinya akan digunakan sebagai Kunci enkripsi.

C. ANALISIS PERCOBAAN

Dari percobaan, penulis sempat mengalami beberapa masalah dalam mengaplikasikan algoritma pertukaran kunci ini. Masalah-masalah tersebut antara lain:

1. Hasil pemangkatan nilai g dengan kunci privat a atau b ternyata hasilnya sangat besar untuk ditangani tipe dasar *int* pada kakas C #. Maka penulis mencoba mengganti tipe int menjadi *long* karena daya tampungnya yang lebih besar. Namun ternyata long pun tidak cukup besar untuk menampung hasil pangkat tersebut. Maka akhirnya penulis mencoba menggunakan library BigInteger sebagai tempat menampung hasil pemangkatan tersebut. Ternyata library BigInteger pun hanya dapat menangani pemangkatan sampai kira-kira pangkat 100. Angka integer g yang dipangkatkan pun bernilai paling besar 1999.
2. Begitu juga dengan kunci publik. Kunci publik yang dipangkatkan juga memiliki batas maksimal, yaitu n. Padahal semakin besar nilai nilai g, n dan kunci privat masing-masing semakin baik pula algoritma ini dan semakin sulit untuk memecahkannya.

D. SERANGAN TERHADAP DIFFIE-HELLMAN

Setelah melakukan percobaan dan menganalisis sistem algoritma pertukaran kunci Diffie-Hellman, penulis dapat melihat serangan yang dapat terjadi pada algoritma ini.

Berikut adalah tabel untuk membantu menyederhanakan informasi yang keluar dan masuk. (Eve ikut mendengarkan rahasia -ia melihat apa yang dikirim antara Alice dan Bob, tapi dia tidak mengubah isi komunikasi mereka.)

- Misalkan s = kunci rahasia bersama. s = 2
- Misalkan g = 5 , n = 23
- Misalkan kunci privat Alice = 6
- Misalkan A = kunci publik Alice. $A = g^a \bmod p = 8$
- Misalkan b = kunci privat Bob. b = 15
- Misalkan B = kunci publik Bob. $B = g^b \bmod p = 19$

Alice

knows	doesn't know
p = 23 base g = 5 a = 6 $A = 5^6 \bmod 23 = 8$ $B = 5^b \bmod 23 = 19$ $s = 19^6 \bmod 23 = 2$ $s = 8^b \bmod 23 = 2$ $s = 19^6 \bmod 23 = 8^b \bmod 23$ s = 2	b = ?

Bob

knows	doesn't know
p = 23 base g = 5 b = 15 $B = 5^{15} \bmod 23 = 19$ $A = 5^a \bmod 23 = 8$ $s = 8^{15} \bmod 23 = 2$ $s = 19^a \bmod 23 = 2$ $s = 8^{15} \bmod 23 = 19^a \bmod 23$ s = 2	a = ?

Eve

knows	doesn't know
p = 23 base g = 5 $A = 5^a \bmod 23 = 8$ $B = 5^b \bmod 23 = 19$ $s = 19^a \bmod 23$ $s = 8^b \bmod 23$ $s = 19^a \bmod 23 = 8^b \bmod 23$	a = ? b = ? s = ?

Penjelasan : Akan sulit bagi Alice untuk memecahkan kunci privat Bob atau sebaliknya. Jika tidak sulit bagi Alice untuk memecahkan kunci privat Bob, atau sebaliknya, maka tidak sulit bagi Eve untuk mengganti pasangan kunci privat dan publiknya lalu memasang kunci publik Bob ke kunci privatnya, menghasilkan kunci rahasia bersama palsu dan memecahkan kunci privat Bob. (dan selanjutnya digunakan untuk memecahkan kunci rahasia bersama. Eve mungkin memilih pasangan kunci publik dan kunci privat yang memudahkannya untuk memecahkan kunci privat Bob.)

III. Conclusion

Setelah melakukan percobaan dan menganalisis sistem algoritma pertukaran kunci Diffie-Hellman, penulis dapat menarik beberapa kesimpulan sebagai berikut:

1. Algoritma Diffie-Hellman dapat menghasilkan sebuah kunci enkripsi dengan mengkalkulasi kunci publik dan kunci privat bagi dua orang atau lebih yang ingin membuat suatu shared secret tapi berada dalam saluran komunikasi yang tidak terjaga.
2. Hasil Kunci enkripsi dapat digunakan sebagai kunci algoritma yang lain. Kunci ini hanya sebagai masukan awal dari suatu algoritma enkripsi lainnya.
3. Tingkat keamanan algoritma ini berdasarkan nilai dari d, n, dan kunci privat masing-masing. Semakin besar nilai d, n dan kunci privat, semakin baik tingkat keamanannya. Namun tetap saja algoritma ini dapat dipecahkan. Karena cukup banyak input yang dishare dalam saluran yang tidak aman. Dan eavesdropper dapat mengetahui kunci apabila ia dapat melakukan perhitungan diskrit yang cukup luar biasa ini.

IV. Acknowledgment

Terima kasih kepada Tuhan Yang Maha Esa atas kemurahan-Nya dalam melindungi dan memberkati penulis selama pengerjaan makalah ini. Terima kasih kepada Pak Rinaldi atas kesempatan yang diberikan untuk mengerjakan paper ini. Dan terima kasih kepada semua orang yang sudah berjasa namun tidak dapat disebutkan satu-persatu. Terima kasih.

References

[1] http://en.wikipedia.org/wiki/Diffie-Hellman_key_exchange diakses pada tanggal 14 Mei pukul 22.00
[2] http://en.wikipedia.org/wiki/Public-key_cryptography diakses pada tanggal 14 Mei pukul 22.15
[3] http://www.quadibloc.com/crypto/pk0503.htm diakses pada tanggal 14 Mei pukul 22.24
[4] http://www.stanford.edu/~arnab/rdm-tgc07.pdf diakses pada tanggal 14 Mei pukul 23.00
[5] http://cr.yp.to/ecdh/curve25519-20051115.pdf diakses pada tanggal 14 Mei pukul 23.30

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, 29 April 2010

ttd

Tommy / 13507109

Kelas DH.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace DiffieHellman
{
    class DH
    {
        public readonly int[] primesBelow2000 = {
        2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73,
79, 83, 89, 97,
        101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179,
181, 191, 193, 197, 199,
        211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293,
        307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397,
        401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491,
499,
        503, 509, 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599,
        601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691,
        701, 709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797,
        809, 811, 821, 823, 827, 829, 839, 853, 857, 859, 863, 877, 881, 883, 887,
        907, 911, 919, 929, 937, 941, 947, 953, 967, 971, 977, 983, 991, 997,
        1009, 1013, 1019, 1021, 1031, 1033, 1039, 1049, 1051, 1061, 1063, 1069, 1087,
1091, 1093, 1097,
        1103, 1109, 1117, 1123, 1129, 1151, 1153, 1163, 1171, 1181, 1187, 1193,
        1201, 1213, 1217, 1223, 1229, 1231, 1237, 1249, 1259, 1277, 1279, 1283, 1289,
1291, 1297,
        1301, 1303, 1307, 1319, 1321, 1327, 1361, 1367, 1373, 1381, 1399,
        1409, 1423, 1427, 1429, 1433, 1439, 1447, 1451, 1453, 1459, 1471, 1481, 1483,
1487, 1489, 1493, 1499,
        1511, 1523, 1531, 1543, 1549, 1553, 1559, 1567, 1571, 1579, 1583, 1597,
        1601, 1607, 1609, 1613, 1619, 1621, 1627, 1637, 1657, 1663, 1667, 1669, 1693,
1697, 1699,
        1709, 1721, 1723, 1733, 1741, 1747, 1753, 1759, 1777, 1783, 1787, 1789,
        1801, 1811, 1823, 1831, 1847, 1861, 1867, 1871, 1873, 1877, 1879, 1889,
        1901, 1907, 1913, 1931, 1933, 1949, 1951, 1973, 1979, 1987, 1993, 1997, 1999 };

        public BigInteger g, n;         // g dan n harus prima dimana g < n
        public BigInteger X, Y;         // X dan Y adalah hasil fungsi Diffie-Hellman
        public int privateKey;  // privateKey dihasilkan secara random jika diinginkan
        public int K;           // K adalah kunci yang digunakan sebagai kunci enkripsi

        public int getRandomPrime()
        {
            //fungsi yang menggenerate angka prima berdasarkan waktu saat ini
            DateTime dt = DateTime.Now;
            int prime = dt.Millisecond % 303;
            return primesBelow2000[prime];
        }

        public int getSecondRandomPrime()
        {
            DateTime dt = DateTime.Now;
            int prime = (dt.Millisecond+dt.Second) % 303;
            return primesBelow2000[prime];
        }

        public int getThirdRandomPrime()
        {
            //fungsi yang menggenerate angka prima berdasarkan waktu saat ini
            Random r = new Random();
            int prime = r.Next() % 303;
            return primesBelow2000[prime];
        }
```

```
        public void AutoGenerateGNP()
        {
            g = getRandomPrime();

            while (g == 1999) g = getRandomPrime();

            if (g == 1997) n = 1999;

            n = getSecondRandomPrime();

            if (g > n)
            {
                n = g;
                g = getSecondRandomPrime();
            }
            else if (g == n)
            {
                if (getThirdRandomPrime() > g)
                {
                    n = getThirdRandomPrime();
                }
                else if (getThirdRandomPrime() < g)
                {
                    n = g;
                    g = getThirdRandomPrime();
                }
                else
                {
                    g = 349;
                    n = 1361;
                }
            }

            privateKey = getThirdRandomPrime() * getThirdRandomPrime();
        }

        public BigInteger Power(BigInteger n, int pow)
        {
            BigInteger a = 1;
            for (int i = 0; i < pow; i++)
            {
                a *= n;
            }
            return a;
        }

        public BigInteger GenerateXY(BigInteger gin, BigInteger nin, int xy)
        {
            Console.WriteLine("HasilPow = " + Power(gin, xy));
            X = Power(gin, xy) % nin;
            Console.WriteLine("Hasil = " + X);
            return X;
        }

        public BigInteger GenerateKey(int xory, BigInteger XorY, ulong nin)
        {
            // xory adalah kunci privat dari Penukar kunci 1 atau 2
            // XorY adalah hasil Diffie-Hellman
            // fungsi dibuat untuk (x,Y) atau (y,X)

            return Power(XorY,xory) % nin;
        }

    }
}
```