STUDI DAN ANALISIS PERBANDINGAN KEAMANAN PGP ALGORITMA IDEA-RSA DENGAN PGP ALGORITMA CAST-DH

Pocut Viqarunnisa – NIM : 13503106

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if13106@students.if.itb.ac.id

Abstrak

Pretty good privacy (PGP) adalah program kriptografi yang dikembangkan oleh Phil R. Zimmermann dan menjadi standar untuk melindungi surat elektronik (*email*). PGP versi awal menggunakan algoritma *International Data Encryption Algorithm* (IDEA) sebagai algoritma simetri dan RSA sebagai algoritma kunci publik. Namun pada PGP versi akhir disediakan banyak pilihan algoritma seperti AES, CAST, TripleDES, IDES, dan twofish sebagai algoritma enkripsi kunci simetri, RSA dan Diffie Helman (DH) untuk algoritma kunci publik dan SHA, RIPEDMD dan MD-5 untuk membangkitkan fungsi hash. Akan tetapi algoritma yang pupuler digunakan adalah pasangan algoritma IDEA dan RSA dan pasangan algoritma DH dan CAST.

Sebagai alat pengamanan *email* tentu saja tingkat keamanan PGP perlu diperhatikan. Keamanan PGP bergantung pada algoritma yang digunakan. Algoritma IDEA dan RSA yang pertama kali digunakan dalam program PGP tentu saja memiliki keunggulan tetapi algoritma CAST dan agoritma DH juga memiliki kelebihan. Dengan alasan inilah maka meskipun secara umum PGP saat ini menggunakan algoritma CAST dan DH, PGP masih menyediakan pilihan algoritma IDEA dan RSA untuk mengenkripsi dan memberi tanda tangan digital. Oleh karena itu pada paper ini akan dibahas mengenai keamanan masing-masing algoritma dan membandingkan algoritma IDEA-RSA dengan algoritma CAST-DH dari sisi keamanan algoritmanya untuk mengenkripsi pesan maupun untuk memberi tanda tangan digital dengan menggunakan fungsi hash MD-5.

Kata kunci: *Pretty good privacy* (PGP), algoritma kunci publik, algoritma simetri, *International Data Encryption Algorithm* (IDEA), RSA, CAST, Diffie-Hellman (DH), AES, *TripleDES*, IDES, *twofish*, SHA, RIPEDMD dan MD-5.

1. Pendahuluan

Pengiriman dan penyimpanan pesan melalui media elektronik sudah banyak dilakuakan. Terkadang pengiriman dan penyimpanan pesan melalui media elektronik perlu dirahasiakan untuk menjamin keamanan dan keutuhan data yang dikirimkan. Oleh sebab itu maka dibutuhkan sebuah metode penyandian pesan. Ilmu sekaligus seni untuk menjaga keamanan pesan disebut kriptografi.

Perangkat lunak yang digunakan untuk menyandikan pesan baik yang bersifat komersial maupun yang dikembangkan untuk kepentingan pribadi saat ini sudah banyak digunakan. Meskipun perangkat ini seringkali dilarang oleh pihak-pihak tertentu karena dianggap membahayakan keamanan nasional.

Salah satu perangkat lunak untuk menjaga keamanan pesan surat elektronik yang biasa digunakan adalah *Pretty Good Privacy* (PGP). Perangkat lunak yang semula dikembangkan untuk mengamankan pengiriman email ini, sekarang digunakan pula untuk mengamankan semua jenis file program dan data. Program ini juga dapat membuat tanda tangan digital yang biasa digunakan untuk menandai pesan.

Sebagai program kriptografi, keamanan perangkat lunak PGP tentu saja berhubungan dengan algoritma-algoritma kriptografi yang digunakannya. Pada awal perkembangannya algoritma yang digunakan pada perangkat lunak PGP ini adalah RSA dan IDEA. Namun dalam perkembangannya pada PGP versi akhir disediakan banyak pilihan algoritma seperti

AES, CAST, TripleDES, IDES, dan twofish sebagai algoritma enkripsi kunci simetri, RSA dan Diffie Helman (DH) untuk algoritma kunci publik dan SHA, RIPEDMD dan MD-5 untuk membangkitkan fungsi hash.

Pasangan algoritma kriptografi yang sering digunakan untuk mengenkripsi pesan dengan menggunakan PGP adalah algoritma IDEA dan RSA juga pasangan algoritma CAST dan DH. Karena itu pada makalah kali ini akan dibandingkan tingkat keamanan program PGP yang menggunakan pasangan algoritma IDEA sebagai algoritma kunci simetri dan RSA sebagai algoritma kunci publik dengan CAST sebagai algoritma kunci simetri dan DH sebagai algoritma kunci publik. Yang akan dibandingkan adalah hasil enkripsi pesan dan tandatangan digital yang dihasilkan dengan menggunakan fungsi hash MD-5.

2. *Pretty good Privacy* (PGP)

Pretty Good Privacy atau lebih dikenal dengan sebutan PGP adalah paket perangkat lunak kriptografi yang dipublikasikan pada tahun 1991 oleh penemunya Phil R. Zimmermann dan secara de facto telah menjadi standar untuk mengenkripsi e-mail hingga sekarang[6].

PGP merupakan program yang populer digunakan untuk menenkripsi dan mendekripsi e-mail melalui internet[5]. Program ini juga dapat digunakan untuk mengirimkan tanda tangan digital yang sudah terenkripsi sehingga penerima pesan dapat melakukan verifikasi terhadap identitas pengirim dan dapat memastikan pesan tidak diubah selama mengalami proses pengiriman. Saat ini PGP tidak hanya digunakan untuk pengamanan *email* tetapi juga digunakan untuk untuk keamanan berbagai file dan program komputer[1].

PGP tersedia dalam dua versi yaitu versi *freeware* maupun versi komersil yang dikembangkan agar dapat dioperasikan di berbagai sistem operasi baik DOS, Windows, UNIX maupun Mac. Perangkat lunak PGP ini dapat digunakan baik unruk kepentingan individual maupun untuk kepentingan korporat.

Saat membuat PGP, Zimmerman melakukan usaha-usaha berikut :

1. Memilih algoritma kriptografi terbaik yang ada sebagai komponen dasar pembentuk PGP[2].
2. Mengintegrasikan algoritma - algoritma tersebut kedalam sebuah aplikasi sebaguna yang independent terhadap system operasi dan prosesor dan dijalankan dengan sekumpulan kecil insdktuksi yang mudah digunakan[2].
3. Membuat paket dan dokumentasinya, mencakup kode sumber, dapat diakses secara gratis melalui internet, *bulletin board*, dab jaringan komersial semacam *compuserve*[2].
4. Melakukan perjanjian dengan perusahaan untuk memberikan kompatibilitas yang penuh, versi komersial PGP yang berharga murah[2].

Sejak awal kemunculannya pada awal tahun 90-an, PGP berkembang dengan cepat dan banyak digunakan oleh masyarakat di Amerika. Alasan pertumbuhan PGP ini adalah :

1. Tersedia gratis ke seluruh dunia dalam berbagai versi yang dapat berjalan dalam berbagai platform termasuk DOS/windows, UNIX, Machintosh, dan lain-lain. Versi komersial diperuntukkan untuk pengguna yang menginginkan produk dengan dukungan dari pihak penjual.
2. PGP disusun dari algoritma yang telah bertahan baik selama bertahun-tahun dari berbagai analisis cipher dan dianggap sebagai algoritma yang sangat aman. Paket dasar terdiri dari RSA untuk enkripsi kunci publilk, IDEA untuk enkripsi konvensional, dan MD5 untuk kode hash.
3. PGP mempunyai daerah aplikasi yang luas, mulai dari perusahaan yang ingin memilih dan menjalankan standardisasi untuk mengenkripsi file-file dan pesan hingga individu-individu yang ingin berkomunikasi secara aman dengan relasinya di seluruh dunia melalui internet maupun melalui jaringan lainnya.
4. PGP tidak dikembangkan ataupuun dikontrol oleh organisasi tertentu maupun pemerintah. Kecurigaan masyarakat terhadap lembaga pemerintah maupun swasta ini menyebabkan PGP menjadi lebih menarik.

Pada awal perkembangannya PGP memang mengalami beberapa masalah. Fimmerman sendiri, sebagai penemunya diinvestigasi

pemerintah Amerika selama 3 tahun karena menyediakan perangkat enkripsi yang sangat baik bagi umum yang mungkin akan merugikan pemerintah. Pemerintah Amerika sangat ketat membatasi eksper perangkat enkripsi.
Namun sejak tahun 1995 PGP sudah mulai dapat diperoleh diluar Amerika dengan mudah, sehingga peraturan kontrol ekspor US menjadi tidak relevan lagi. PGP versi gratis menggunakan algoritma yang dipatenkan di Amerika, seperti RSA yang masa patennya berlaku hingga tahun 2000, dan IDEA berlaku hingga tahun 2010. Ini menyebabkan keterbatasan penggunaan PGP versi gratis di Amerika. Oleh karena itu orang berusaha menghindari penggunaan PG versi gratis yang menggunakan algoritma yang masih dipatenkan

PGP menggunakan sistem kriptografi campuran (*hybrid chryptosystem*)[4], faktanya PGP menggunakan baik sistem simetri maupun sistem asimetri dengan alasan sebagai berikut :

1. Saat menggunakan sistem simetri (algortima kunci simetri) sering kali kita khawatir mengenai cara pertukaran kunci dengan penerima pesan, dengan demikian kita cenderung menggunakan sistem asimetri[6].
2. Sistem asimetrik membutuhkan waktu lama untuk mengenkripsi keseluruhan pesan (sekitar 4000 kali lebih lambat dari pada kunci simetri)[6].

Dengan demikian maka PGP terdapat dua macam kunci yaitu pasangan kunci publik dan kunci privat untuk algoritma kunci publik dan juga kunci sesi untuk algoritma kunci simetri.

Versi awal PGP (sebelum versi 5) digunakan algoritma RSA sebagai sistem asimetri dan algoritma IDEA sebagai sistem simetri. Pada versi terakhir digunakan algoritma DSS/Diffie-Hellman sebagai sistem asimetri dan CAST sebagai sistem simetri. Akan tetapi pada PGP versi internasional disediakan pilihan untuk memilih antara kedua pasangan algoritma tersebut.

Untuk menggunakan PGP perlu diketahui 2 kunci yaitu kunci publik penerima pesan dan kunci privat pengirim pesan. Pada kenyataannya terdapat pasangan kunci yang saling berkaitan satu sama lain yaitu kunci publik penerima pesan dan kunci privat pengirim pesan. Sehingga saat mengenkripsi pesan pengirim pesan perlu mengetahui kunci publik penerima pesan dan ini pasti diketahui karena sifatnya yang publik dan seharusnya semua orang mengetahuinya. Saat penerima pesan menerima pesan yang terenkripsi, dia mendekripsi pesan dengan menggunakan kunci privat miliknya yang tidak diketahui orang lain. Sebagai tambahan mungkin penerima akan memberikan jawaban dari pesan yang dikirimkan maka setelah membaca pesan dia akan mengenkripsi pesan yang akan dikirimannya dengan kunci publik penerima pesan. Saat penerima pesan menerima pesan maka dia akan mendekripsikannya dengan kunci privat yang dimilikinya. Ini adalah sistem kriptografi asimetri. Dengan kata lain semua orang mengetahui kunci publik orang lain tetapi hanya orang yang dikirimi pesan yang mengetahui kunci privatnya dan yang dapat mendekripsi isi pesan.

Seperti yang diketahui PGP tidak menggunakan algoritma kunci publik untuk menenkripsi pesan. PGP mengenkripsi pesan dengan menggunakan algoritma kunci simetri. Tetapi untuk dapat mengenkripsi pesan kita tidak perlu mengetahui kunci privat.

Pada kenyataannya PGP membuat kunci rahasia secara acak hanya untuk pesan yang akan dikirim (*one-time only secret key*)[4] saat itu disebut juga kunci sesi. Jika mengenkripsi pesan yang sma pada waktu yang berbeda maka hasilnya pun akan berbeda karena PGP akan membuat kunci sesi baru lagi. Setelah membuat kunci sesi, PGP mengenkripsi pesan dengan menggunakan kunci sesi. Selanjutnya PGP mengenkripsi kunci sesi dengan menggunakan kunci publik penerima pesan. Dan mengirimkan pesan beserta kunci yang telah dienkripsi tadi.

Saat menerima pesan, penerima mendekripsi kunci yang dikirimkan dengan menggunakan kunci privat penerima. Perlu diketahui bahwa hanya penerima pesan yang dapat mendekripsi pesan karena menggunakan dekripsi hanya bisa dilakukan dengan kunci privat. Dan kemudian menggunakan kunci yang telah terdekripsi untuk mendekripsikan pesan sesungguhnya yang dikirimkan.

Pada PGP versi 8.0 *for windows* terdapat tiga program PGP yaitu : *PGPdisk, PGPkeys* (untuk pembangkitan dan manajemen kunci), dan *PGPmail* (enkripsi dan tandatangan digital untuk *file* maupun *e-mail*)[1]. Sedangkan pada PGP Desktop versi 9.5 terdapat beberapa sub program yaitu : *PGP keys,PGP Messaging, PGP Zip,*

PGP NetShare, *PGP Virtual Disk*, dan *PGP Whole Disk Encryption*. Namun untuk versi non komersial yang dapat digunakan hanya *PGP keys* untuk membangkitkan kunci dan *PGP ZIP* untuk mengenkripsi, mendekripsi dan menandatangani file.

3. Tanda Tangan Digital (*Digital Signature*)

Tanda tangan sering digunakan untuk membuktikan otentikasi dokumen kertas karena memiliki beberapa karakteristik yaitu merupakan bukti otentik, tidak mudah dilupakan, tidak dapat dipindah untuk digunakan ulang, dokumen yang ditandatangani tidak dapat diubah dan tidak dapat disangkal. Fungsi tanda tangan pada dokumen kertas juga diterapkan untuk otentikasi pada data digital.

Tanda tangan pada data digital ini disebut tanda tangan digital (*digital signature*). Yang dimaksud dengan tanda tangan adalah suatu nilai kriptografis yang bergantung pada pesan dan pengirim pesan. Dengan tanda tangan digital, maka integritas data dapat dijamin, disamping itu ia juga digunakan untuk membuktikan asal pesan (keabsahan pengirim dan anti-penyanggahan).

Tandatangan digital dapat memenuhi aspek keamanan kriptografi. Aspek yang mempengaruhi keamanan kriptografi adalah :

1. Kerahasiaan (*confidentiality*) adalah layanan yang digunakan untuk menjaga isi pesan dari siapapun yang tidak berhak untuk membaca pesan tersebut. Dalam kriptografi, biasanya pesan dienkripsi menjadi bentuk yang tidak dapat dimengerti.

2. Integritas data (*data integrity*), adalah layanan yang menjamin bahwa pesan masih asli atau belum pernah dimanipulasi selama pengrimina pesan. Untuk menjaga integritas data, sistem harus memiliki kemampuan untuk mendeteksi terjadinya manipulasi pesan oleh pihak-pihak yang tidak berhak, antara lain penyisipan, penghapusan, atau substitusi data lain ke dalam pesan yang sebenarnya.

3. Otentikasi (*authentication*), adalah layanan yang berhubungan dengan identifikasi, baik mengidentifikasi kebenaran pihak-pihak yang berkomunikasi (*user authentication* atau *entity authentication*), maupun mengidentifikasi kebenaran sumber pesan (*data origin authentication*). Dua pihak yang saling berkomunikasi juga harus diotentikasi asalnya. Otentikasi sumber pesan secara implisit juga memberikan kepastian integritas data, sebab jika pesan telah dimodifikasi berarti sumber pesan sudah tidak benar. Oleh karena itu, layanan integritas data selalu dikombinasikan dengan layanan otentikasi sumber pesan.

4. Nirpenyangkalan (*non-repudiation*), adalah layanan untuk mencegah entitas yang berkomunikasi melakukan penyangkalan, yaitu pengirim pesan menyangkal melakukan pengiriman atau penerima pesan menyangal telah menerima pesan.

Tanda tangan digital mampu memenuhi tiga dari empat aspek keamanan kriptografi, yaitu aspek integritas data, otentikasi, dan nirpenyangkalan. Proses pembuatan tanda tangan digital dapat dilakukan dengan dua cara, yaitu:

1. Enkripsi pesan
   Dengan mengenkripsi pesan maka secara otomatis telah dilakukan proses otentikasi. Pesan yang telah terenkripsi dapat dikatakan telah ditanda tangani.

2. Tanda tangan digital dengan fungsi *hash*
   Tanda tangan digital dibangkitkan dari *hash* terhadap pesan. Nilai *hash* adalah kode ringkas dari pesan. Tanda tangan digital kemudian ditambahkan pada pesan.

3.1 Penandatanganan dengan mengenkripsi pesan

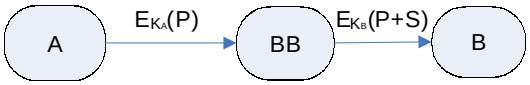
Proses pengenkripsian pesan sebagai tanda tangan digital dapat dilakukan dengan dua cara yaitu dengan algoritma simetri ataupun dengan algoritma asimetri

Penandatanganan algoritma kriptografi kunci simetri

Sebenarnya, algoritma kriptografi simetri sudah memberikan solusi untuk masalah keamanan pertama, kedua, dan ketiga, karena kunci simetri hanya diketahui oleh pengirim dan penerima. Jadi, jika *B* menerima pesan dari *A*, maka ia percaya pesan itu dari *A* dan isinya tidak mengalami perubahan, karena tidak ada orang lain yang mengetahui kunci selain mereka berdua.

Namun, algoritma kriptografi simetri tidak dapat menyediakan cara untuk mengatasi masalah keamanan yang ketiga, yaitu jika salah satu dari dua pihak, *A* dan *B*, membantah isi pesan atau telah mengirim pesan.

Agar dapat melakukan nirpenyangkalan dengan kriptografi kunci simetri, maka diperlukan pihak ketigga yang dipercaya oleh pengirim dan penerima. Pihak ketiga ini disebut arbitrase. Misalkan BB (*Big brothers*) adalah otoritas atbitrase yang dipercaya oleh A dan B.



Gambar skema penandatanganan pesan dengan bantuan arbitrase

Jika A menyangkal telah mengirim pesan tersebut, maka pernyataan BB pada pesan yang diterima B dapat digunakan untuk menolak penyangkalan A. BB tau pesan berasal dari A dan bukan dari C karena hanya BB dan A yang mengetahui kunci rahasia, maka hanya A yang dapat mengenkripsi pesan tersebut.

Penandatanganan algoritma kriptografi kunci publik

Jika kriptografi kunci publik digunakan, maka enkripsi pesan dengan kunci publik tidak dapat digunakan untuk otentikasi, karena setiap orang potensial untuk mengetahui kunci publik. Tetapi jika enkripsi pesan menggunakan kunci privat pengirim dan dekripsi pesan menggunakan kunci public si pengirim, maka kerahasiaan pesan (*secrecy*) dan otentikasi keduanya dicapai sekaligus, Ide ini ditemukan okeh Diffie-Hellman.

Beberapa algoritma kunci publik dapat digunakan untuk menandatangani pesan dengan cara mengenkripsinya, asal algoritma tersebut memenuhi sifat $D_{PK}(E_{SK}(M)) = M$ dan $D_{SK}(E_{PK}(M)) = M$ dengan *PK* = kunci public, *SK*= kunci privat

Contoh algoritma yang memenusi sifat tersebut adalah algoritma kunci publik RSA. Algoritma kunci-publik tersebut dapat digunakan untuk membuat sidik dijital.

Misalkan *M* adalah pesan yang akan dikirim. Tanda tangan digital *S* untuk pesan *M* diperoleh dengan mengenkripsikan *M* dengan menggunakan kunci rahasia (*SK*) pengirim,

$$S = E_{SK}(M)$$

yang dalam hal ini, *E* adalah fungsi enkripsi dari algoritma kunci-publik. Selanjutnya, S dikirim melalui saluran komunikasi.

Di tempat penerima, pesan dibuktikan kebenaran sidik-dijitalnya dengan menggunakan kunci publik (*PK*) pengirim,

$$M = D_{PK}(S)$$

yang dalam hal ini, *D* adalah fungsi enkripsi dari algoritma kunci-publik. Tanda tangan digital *S* dikatakan absah apabila pesan M yang dihasilkan merupakan pesan yang mempunyai makna.

3.2 Penandatanganan dengan fungsi hash

Penandatanganan pesan dengan cara mengenkripsinya selalu memberikan dua fungsi berbeda: kerahasiaan pesan dan otentikasi, Pada beberapa kasus, seringkali otentikasi yang diperlukan, tetapi kerahasiaan pesan tifa. Maksudnya, pesan tidak perlu dienkripsikan, sebab yang dibutuhkan hannya otentikasi saja.

Hanya sistem kriptografi kunci-publik yang cocok dan alami untuk pemberian tanda tangan digital dengan menggunakan fungsi *hash*. Hal ini disebabkan karena skema tanda tangan digital berbasis sistem kunci publik dapat menyelesaikan masalah *non-repudiation* (baik penerima dan pengirim pesan mempunyai pasangan kunci masing-masing)

Proses pemberian tanda tangan digital (*Signing*)

Pesan yang hendak dikirim diubah terlebih dahulu menjadi bentuk yang ringkas yang disebut *message digest*. *Message digest* (*MD*) diperoleh dengan mentransformasikan pesan *M* dengan menggunakan fungsi *hash* satu-arah (*one-way*) *H*,

$$MD = H(M)$$

Pesan yang sudah diubah menjadi *message digest* oleh fungsi *hash* tidak dapat dikembalikan lagi menjadi bentuk semula walaupun digunakan algoritma dan kunci yang sama (itulah sebabnya dinamakan fungsi *hash* satu-arah). Sembarang pesan yang berukuran apapun diubah oleh fungsi

hash menjadi *message digest* yang berukuran tetap (umumnya 128 bit).

Selanjutnya, *message digest* MD dienkripsikan dengan algoritma kunci-publik menggunakan kunci rahasia (SK) pengirim menjadi tanda tangan digital *S*,

$$S = E_{SK}(MD)$$

Pesan *M* disambung (*append*) dengan tanda tangan digital *S*, lalu keduanya dikirim melalui saluran komunikasi. Dalam hal ini, kita katakan bahwa pesan *M* sudah ditandatangani oleh pengirim dengan tanda tangan digital *S*.

Di tempat penerima, pesan diverifikasi untuk dibuktikan keotentikannya dengan cara berikut:

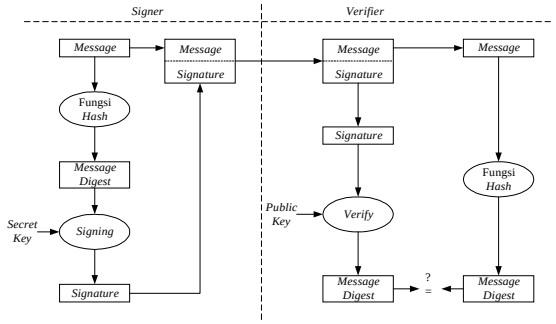
1. Tanda tangan digital *S* didekripsi dengan menggunakan kunci publik (*PK*) pengirim pesan, menghasilkan *message digest* semula, *MD*, sebagai berikut:

$$MD = D_{PK}(S)$$

2. Pengirim kemudian mengubah pesan *M* menjadi *message digest MD'* menggunakan fungsi *hash* satu-arah yang sama dengan fungsi *hash* yang digunakan oleh pengirim.

3. Jika *MD' = MD*, berarti pesan yang diterima otentik dan berasal dari pengirim yang benar.

Skema otentikasi dengan tanda tangan digital menggunakan fungsi *hash* ditunjukkan pada Gambar.



Gambar Otentikasi tanda tangan digital menggunakan fungsi *hash* satu-arah

Otentikasi pesan dijelaskan sebagai berikut:

a. Apabila pesan *M* yang diterima sudah berubah, maka *MD'* yang dihasilkan dari fungsi *hash* berbeda dengan *MD* semula. Ini berarti pesan tidak asli lagi.

b. Apabila pesan *M* tidak berasal dari orang yang sebenarnya, maka *message digest MD* yang dihasilkan dari persamaan 3 berbeda dengan *message digest MD'* yang dihasilkan pada proses verifikasi (hal ini karena kunci publik yang digunakan oleh penerima pesan tidak berkoresponden dengan kunci rahasia pengirim).

c. Bila *MD = MD'*, ini berarti pesan yang diterima adalah pesan yang asli (*message authentication*) dan orang yang mengirim adalah orang yang sebenarnya (*user authentication*).

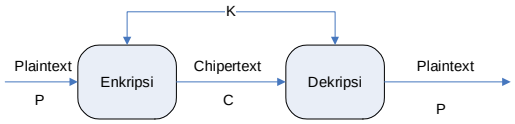
Pengirim pesan tidak dapat menyangkal pesan yang ia kirim, sebab tanda tangan digital dapat digunakan untuk melakukan nirpenyangkalan. Andaikan pengirim berbohong telah mengirim pesan. Sangkalan dari pengirim dapat dibantah dengan cara sebagai berikut : jika ia tidak mengirim pesan, berarti ia tidak mengenkripsi *message digest* dari pesan dengan kunci privatnya. Faktanya, kunci public yang berkoresponden dengan kunci privat pengirim menghasilkan *MD=MD'* Ini berarti *message digest* memang benar dienkripsi oleh pengirim sebab hanya pengirim yang mengetahui kunci privatnya sendiri.

4. Algoritma Simetri

Berdasarkan jenis kuncinya terdapat dua jenis algoritma kriptografi yaitu algoritma simetri (konvensional) dan algoritma asimetri (kunci-publik).

Algoritma simetri adalah algoritma yang menggunakan kunci enkripsi yang sama dengan kunci dekripsinya. Disebut juga algoritma konvensional karena algoritma ini biasa digunakan orang sejak berabad-abad yang lalu. Algoritma simetri sering juga disebut sebagai algoritma kunci rahasia, algoritma kunci tunggal, atau algoritma satu kunci, dam mengharuskan pengirim dan penerima menyetujui suatu kunci tertentu sebelum mereka dapat berkomuikasi dengan aman. Keamanan algoritma simetri tergantung pada kunci, membocorkan kunci berarti bahwa orang lain dapat mengenkrip den mendekrip pesan. Agar komunikasi tetap aman, kunci harus tetap dirahasiakan. Yang termasuk algoritma kunci simetri adalah OTP, DES, RC2, RC4, RC5, IDEA, Twofish, magenta, FEAL,

SAFER, LOKI, CAST, Rijndael (AES), Blowfish, GOST, A5, Kasumi dan lain-lain.



Gambar kriptografi kunci simetri

Pada gambar dapat dilihat bahwa kunci yang digunakan untuk mengenkripsi dan mendekripsi sama karena itu maka disebut kriptografi kunci simetri. Pesan asli yang belum dikodekan disebut plainteks. Plainteks tidak harus berupa teks, namun dapat berupa file gambar, file biner, file audio dan sebagainya. File yang telah disandikan disebut cipherteks. Enkripsi adalah proses peribahan pesan asal menjadi karakter yang tidak dapat dibaca. Sedangkan dekripsi adalah proses pengubahan karakter yang tidak dapat dibaca menjadi pesan asal. Cipherteks inilah yang biasanya dikirimkan melakui saluran internet sehingga rawan penyadapan.

Untuk mempermudah penulisan dan analisis dalam kriptografi seringkali digunakan notasi matematika. Notasi matematika untuk proses enkripsi kunci simetri digambarkan sebagai berikut :

$$C = E_k(P)$$

Dengan C menyatakan kode rahasia, P menyatakan pesan asal, K menyatakan kunci dan fungsi E menyatakan fungsi enkripsi yang dioperasikan terhadap masukan P dengan kunci K.

Sedangkan proses dekripsi digambarkan dengan notasi sebagai berikut :

$$P = D_k(C)$$

Dengan fungsi D menyatakan fungsi dekripsi yang dioperasikan terhadap masukan P dengan kunci K.

Secara umum algoritma simetri dapat dibagi dalam dua kategori yaitu algoritma aliran (*stream cipher*) dan algoritma blok (*block cipher*).

Cipher aliran adalah algoritma kriptografi yang beroperasi pada plainteks/cipherteks dalam bentuk bit tunggal. Rangkaian bit dienkripsi atau didekripsi bit per bit.

Sedangkan *Cipher* blok adalah algoritma kriptografi yang beroperasi pada plainteks dan cipherteks dalam bentuk blok bit. Rangkaian bit dibagi menjadi blok-blok bit yang panjangnya sudah ditentukan sebelumnya. Prosen enkripsi dan dekripsi dilakukan terhadap masing-masing blok. Baik algoritma IDEA maupun algoritma CAST yang akan dibahas pada paper ini menggunakan kunci simetri dengan kategori *cipher* blok.

4.1 Algoritma IDEA

International data encryption algorithm (IDEA) dibuat pada tahun 1992 oleh Xuejia Lain dan James Massey. Dirancang untuk menggantikan DES. Algoritma ini dibuat tanpa menggunakan jaringan Feistel. Seperti talah dijelaskan sebelumnya agoritma ini merupakan algoritma kunci simetri dengan kategori *cipher* blok yang beroperasi pada blok plainteks 64-bit. Memiliki panjang kunci 128 bit dan menggunakan algoritma yang sama untuk proses ekripsi maupun dekripsi.

Seperti algoritma yang lain algoritma ini menggunakan prinsip *shanon confusion* dan *diffusion*.

Kekuatan dari algoritma ini adalah melakukan operasi pengacakan (*mixing*) dengan memanfaatkan beberapa operasi aljabar. Secara umum ada tiga operasi aljabar yang digunakan yaitu :

- XOR
- Penambahan modulo 2^{16}
- Perkalian modulo $2^{16} + 1$ (operasi yang menggantikan kotak S)

4.1.1 Pembangkitan subkunci

Algoritma ini membutuhkan enam kunci untuk setiap tahapan dari 8 tahapan, dan 4 kunci untuk transformasi keluaran. Sehingga jumlah kunci yang dibutuhkan adalah 6 X 8 + 4 = 52 buah. Setiap sub kunci berukuran 16 bit.

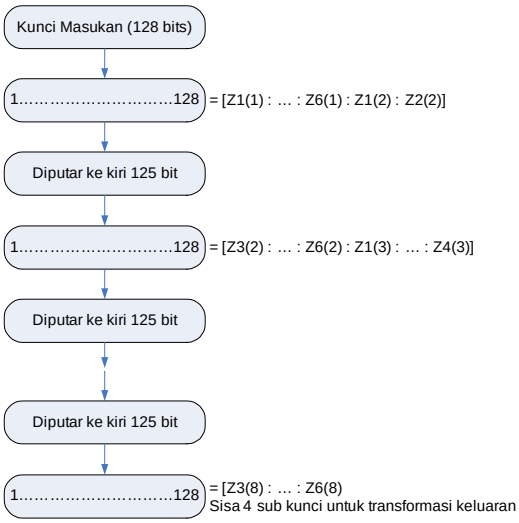
Pembangkitan subkunci dilakukan dengan cara sebagai berikut :

- Membagi kunci 128-bit menjadi 8 subkunci yang masing-masing berukuran 16 bit.
- Ke 128 digit kunci dirotasikan kekiri sebanyak 25 bit untuk membuat kunci baru

yang kemudian akan dibagi menjadi 8 subkunci 16-bit selanjutnya.

Proses ini dilakuan terus menerus hingga didapatkan 52 subkunci.

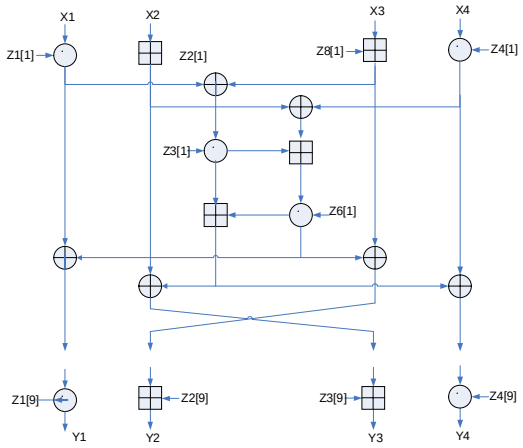
Proses pembangkitan kunci dapat dilihat pada gambar berikut.



Gambar Pembangkitan Kunci dengan IDEA

4.1.2 Enkripsi

Proses enkripsi dilakukan dengan membagi blok data 64 bit menjadi empat subblok 16 bit X_1, X_2, X_3, X_4. Empat subblok ini menjadi masukan bagi iterasi tahap pertama algoritma. Iterasi akan dilakukan sebanyakt 8 iterasi. Pada setiap iterasi, 4 sub-blok akan di-X-OR kan, ditambah, dan dikalika satu sama lain dengan 6 sub-kunci 16-bit. Diantara satiap iterasi sub-blok kedua dan ketiga saling dipertukarkan. Akhirnya 4 sub-blok dikombinasikan dengan 4 sub-kunci dalam transformasi keluaran. Berikut adalah gambaran tahapan pada setiap iterasi. Dengan Z sebagai subkunci dan X sebagai subblok.



Gambar skema enkripsi dengan IDEA

Pada setiap tahapan dilakukan urutan berikut :
1. Kalikan X_1 dengan sub-kunci pertama Z1(1)
2. Tambahkan X_2 dengan sub-kunci kedua
3. Tambahkan X_3 dengan sub-kunci ketiga
4. Kalikan X_4 dengan sub-kunci keempat
5. XOR hasil langkah (1) dengan (3)
6. XOR hasil langkah (2) dengan (4)
7. Kalikan hasil langkah (5) dengan subkunci kelima
8. Tambahkan hasil langkah (6) dan (7)
9. Kalikan hasil langkah (8) dengan subjkunci ke-6
10. Tambahkan hasil langkah (7) dan (9)
11. XOR hasil langkah (1) dengan (9)
12. XOR hasil langkah (3) dengan (9)
13. XOR hasil langkah (2) dengan (10)
14. XOR hasil langkah (4) dengan (10)

Keluaran setiap tahapan adalah 4 sub-blok yang merupakan hasil langkah (1), (12), (13), dan (14). Pertukarkan 2 subblok dalam (blok 2 dan 3) kecuali pada tahapan terakhir. Hasil itu merupakan input untuk tahapan berikutnya.

Setelah melakuakn tahapan tersebut sebanyak 8 iterasi lakukan :
1. Kalikan X_1 dengan sisa sub-kunci pertama
2. Tambahkan X_2 dengan sisa sub-kunci kedua
3. Tambahkan X_3 dengan sisa sub-kunci ketiga
4. Kalikan X_4 dengan sisa sub-kunci keempat

Kemudian gabungkan keempat sub-blok untuk menghasilkan cipherteks.

4.1.3 Dekripsi

Proses dekripsi dilakukan dengan cara yang hampir sama dengan proses enkripsinya kecuali pada bagian pembentukan sub-kunci. Proses

dekripsi ini menerima masukan 64-bit blok *cipher* dan menghasilkan keluaran plainteks. Empat subkunci pertama untuk dekripsi adalah :

$K_d(1) = 1 / K(49)$ (inversi perkalian mod $2^{16}+1$)
$K_d(2) = -K(50)$ (inversi penjumlahan mod 2^{16})
$K_d(3) = -1K(51)$
$K_d(4) = 1 / K(52)$

Pembentukan kunci fekripsi berikut berulang 8 kali, dengan menambahkan 6 ke indeks kunci dekripsi dan mengurangi 6 dari setiap indeks kunci enkripsi.

$K_d(5) = K(47)$
$K_d(6) = K(48)$

$K_d(7) = 1 / K(43)$
$K_d(8) = -K(45)$
$K_d(9) = -1K(44)$
$K_d(10) = 1 / K(46)$

IDEA terbukti menjadi salah satu algoritma enkripsi terbaik di dunia, setelah 12 tahun tidak ditemukan cara pemecahan kode rahasianya meskipun telah dianalisis oleh ratusan ahli kriptanalis terbaik di dunia. Tentu saja yang dimaksud adalah tidak dijumpainya publikasi yang menyatakan keberhasilan meng-*attack* IDEA. Jadi tidak menutup kemungkinan adanya keberhasilan semacam itu, namun tidak dipublikasikan. Sebab seringkali, keberhasilan semacam itu justru harus dirahasiakan oleh pihak-pihak tertentu seperti NSA, dan juga saat sekutu berhasil memecahkan sandi ENIGMA Jerman dalam perang dunia kedua.

4.2 Algoritma CAST

CAST adalah algoritma kriptografi kunci simetri yang konstruksinya secara umum mirip dengan DES Panjang kuncinya bervariasi tetapi dalam PGP digunakan panjang kunci maksimum yaitu 128 bit. Algoritma ini merupakan salah satu kandidat dalam kontes yang diadakan oleh NIST untuk mendapatkan AES (*Advanced Encryption Standatd*). Algoritma dasar CAST nya adalah sebagai berikut :

INPUT : plainteks m1...m64;key K = k1...k128.
OUTPUT : cipherteks c1...c64.

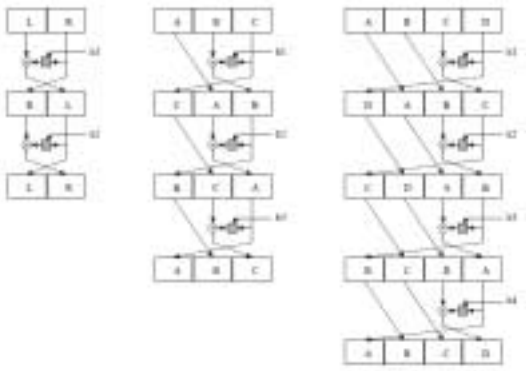
1. Penjadwalan kunci menghasilkan 16 pasang subkunci {Km_i, Kr_i} dari K. Kr_i yang berukuran 5 bit digunakan sebagai kunci "*rotasi*" untuk rotasi ke *i* sedangkan Km_i merupakan kunci "*masking*"untuk rotasi ke *i*[10].
2. (L0,R0) ← (m1...m64). Bagi painteks menjadi dua bagian kiri dan kanan. Sebelah kiri L0=mi…m32 sedangkan bagian kanan R0=m33…m64.
3. 16 putaran untuk i dari 1 sampai 16, hitung Li dan Ri dengan perhitungan sebagai berikut :

$$L_i = R_{i-1}$$

$$R_i = L_{i-1}^{f(R_{i-1},K_{mi},K_{ri})}$$

4. c1…c64 ← (R16,L16). Tukarkan blok fina1 L16, R16 dan gabungkan dengan cipherteks.

Secara umum algoritma CAST membentuk jaringan feistel seperti halnya Algoritma DES. Untuk memudahkan penggambaran mengenai jaringan feistel dapat dilihat pada gambar.



Gambar jaringan feistel

Diagram yang paling kiri adalah jaringan feistel tradisional. Digunakan untuk cipher dengan ukuran blok 64-bit 32 bit L dan 32 bit R. Diagram tengah menggambarkan perkembangan jaringan feistel untuk cipher yang memiliki ukuran blok 96-bit. Sedangkan diagram yang paling kanan menggambarkan struktur cipher dengan ukuran blok 128 bit seperti yang digunakan pada algoritma CAST ini.

Untuk masing-masing rotasi dilakukan fungsi matematis yang dibedakan menjadi tiga tipe yaitu :

Tipe 1:

$$I = ((k_{m_i} + D) \lrcorner k_{r_i})$$

$$O = ((S_1[I_a] \oplus S_2[I_b]) - S_3[I_c]) + S_4[I_d]$$

Tipe 2:

$$I = ((k_{m_i} \oplus D) \lhd k_{r_i})$$
$$O = ((S_1[I_a] - S_2[I_b]) + S_3[I_c]) \oplus S_4[I_d]$$

Tipe 3:

$$I = ((k_{m_i} - D) \lhd k_{r_i})$$
$$O = ((S_1[I_a] + S_2[I_b]) \oplus S_3[I_c]) - S_4[I_d]$$

Dimana *D* adalah data input untuk operasi, *Ia* - *Id* adalah byte yang paling signifikan dari *I*, *Si*adalah s-box ke *i* dan *O* adalah output dari operasi. Sedangkan operator + dan – menyatakan menambahan dan pengurangan dengan modulo 232, ⊕ adalah operasi *bitwise eXclusive-OR*, dan ↵ adalah operasi *left-shift*.

Untuk algoritma CAST-128 terdapat 4 macam S-box yang dibambarkan sebagai berikut. Dibaca dari kiri ke kanan atas ke bawah.

Gambar S-box tipe 1

CAST tahan terhadap baik diferensial maupun linear kriptanalisis dan tidak memiliki kunci lemah ataupun kumci semi lemah (*semi weak key*)[3]. Algoritma CAST biasanya digunakan dalam PGP.

5. Algoritma Kunci publik

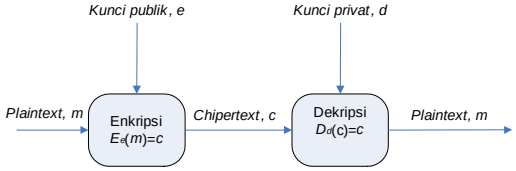
Kunci publik pertama kali diperkenalkan oleh Whitfield Diffie dan Martin Hellman pada tahun 1975 sebagai strategi untuk memecahkan masalah pendistribusian kunci pada kripografi kunci simetri[4]. Sebenarnya penemu kriptografi kunci publik yang pertama adalah James H Ekkis, Clifford Cocks, dan Malcolm Williamson di Inggris pada awal 1970 namun tidak pernah dipublikasikan hingga saat ini.

Algoritma asimetri atau disebut juga algoritma kunci publik merupakan algoritma yang didesain sedemikian rupa sehingga kunci yang digunakan untuk enkripsi disebut kunci publik berbeda dengan kunci yang digunakan untuk dekripsi disebut kunci privat[2]. Kedua kunci ini saling berkorespondensi satu sama lain. Meskipun demikian kunci privat dibuat sedemikian sehingga tidak dapat dihitung dari kunci publiknya. Algoritma ini disebut kunci publik karena kunci enkripsi dapat dibuat publik yang berarti semua orang boleh mengetahuinya. Semua orang dapat menggunakan kunci enkripsi untuk mengenkripsi pesan, namun hanya orang tertentu (penerima pesan sekaligus pemilik kunci dekripsi/pasangan kunci publik) yang dapat mendekripsikan pesan.

Setidaknya terdapat dua keuntungan penggunaan kriptografi kunci publik ini. Pertama, tidak ada kebutuhan untuk mendistribusikan kunci privat seperti pada kriptografi kunci simetri. Kunci publik dapat dikirim ke penerima melalui saluran yang sama dengan saluran yang digunakan untuk mengirim pesan yang biasanya tidak aman. Kedua, jumlah kunci dapat ditekan karena untuk berkomunikasi secara rahasia dengan banyak orang tidak perlu kunci rahasia sebanyak orang tersebut, cukup dua buah kunci yaitu kunci publik bagi para koresponden untuk mengenkripasi pesan dan kunci privat untuk mendekripsi pesan. Berbeda dengan kriptografi kunci simetri ydimana jumlah kunci yang dibuat adalah sebanyak jumlah pihak yang diajak berkorespondensi.[1]

Beberapa contoh algoritma kriptografi kunci publik adalah El-Gamal, RSA, Diffie-Helman, dan DSA (*Digital signature Algorithm*).

Konsep kriptografi kunci publik sederhana dan elegan tetapi mempunyai konsekuensi penggunaan yang hebat. Pada algoritma kunci publik, pengguna memiliki sepasang kunci satu kunci untuk enkripsi yang disebut kunci publik disimbolkan dengan e dan satu kunci untuk dekripsi yang bersifat rahasia dan disebut kunci privat disimbolkan dengan d. Prosess enkripsi dan dekripsi dengan kunci publik digambarkan pada gambar berikut



Gambar Kriptografi Kunci Publik

Misalkan E adalah fungsi enkripsi dan D adalah fungsi dekripsi. Misalkan (e,d) adalah pasangan kunci untuk enkripsi dan dekripsi sedemikian sehingga $E_e(m) = c$ dan $D_d(c) = m$ untuk suatu plaiteks m dan cipherteks c. Kedua persamaan ini menyiratkan bahwa dengan mengetahui e dan c maka secara komputasi hampir tidak mungkin menemukan m. Asumsi lainnya, dengan mengetahui e, secara komputasu hampir tidak mungkin menurunkan d. E_e digambarkan sebagai fungsi puntu kolong (*trapdoor*) satu arah dengan d adalah informasi *trapdoor* yang diperlukan untuk menghitung fungsi inversunya, D, yang dalam hal ini membuat proses dekripsi dapat dilakukan.

Kelemahan sistem kriptografimkunci publik terletak pada :

1. Enkripsi dan dekripsi data umumnya lebih lambat daripada sistem simetri, karena enkripsi dan dekripsi melibatkan operasi perpangkatan yang besar.
2. Ukuran cipherteks lebih besar daripada plainteks (bisa dua sampai empat kali ukuran plainteks).
3. Karena kunci publik diketahui secara luas dan dapat digunakan setiap orang, maka cipherteks tidak memberikan informasi mengenai otentikasi pengirim.

5.1 Algoritma RSA

Dari sekian banyak algoritma kriptografi kunci-publik yang pernah dibuat, algoritma yang paling populer adalah algoritma RSA. Algoritma RSA dibuat oleh 3 orang peneliti dari MIT (*Massachussets Institute of Technology*) pada tahun 1976, yaitu: Ron (R)ivest, Adi (S)hamir, dan Leonard (A)dleman.

Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima. Pemfaktoran dilakukan untuk memperoleh kunci privat. Selama pemfaktoran bilangan besar menjadi faktor-faktor prima belum ditemukan algoritma yang mangkus, maka selama itu pula keamanan algoritma RSA tetap terjamin.

Besaran-besaran yang digunakan pada algoritma RSA:

a. p dan q bilangan prima (rahasia)
b. 2. $r = p \cdot q$ (tidak rahasia)
c. $\phi(r) = (p-1)(q-1)$ (rahasia)
d. 4. PK (kunci enkripsi) (tidak rahasia)
e. 5. SK (kunci dekripsi) (rahasia)
f. 6. X (plainteks) (rahasia)
g. 7. Y (cipherteks) (tidak rahasia)

Prosedur Membuat Pasangan Kunci

a. Pilih dua buah bilangan prima sembarang, p dan q.
b. Hitung $r = p \cdot q$. Sebaiknya $p \neq q$, sebab jika $p = q$ maka $r = p^2$ sehingga p dapat diperoleh dengan menarik akar dari r.
c. Hitung $\phi(r) = (p-1)(q-1)$.
d. Pilih kunci publik, PK, yang relatif prima terhadap $\phi(r)$.
e. Bangkitkan kunci rahasia dengan menggunakan persamaan $SK \cdot PK \equiv 1 \pmod{\phi(r)}$.

SK juga dapat dihitung dengan

$$SK = \frac{1 + m\varphi(r)}{PK}$$

Akan terdapat bilangan bulat m yang menyebabkan memberikan bilangan bulat SK.

Enkripsi

Plainteks disusun menjadi blok-blok $x_1, x_2, \ldots,$ sedemikian sehingga setiap blok merepresentasikan nilai di dalam rentang 0 sampai $r-1$.

Setiap blok x_i dienkripsi menjadi blok y_i dengan rumus

$$y_i = x_i^{PK} \bmod r$$

Dekripsi

Setiap blok cipherteks y_i didekripsi kembali menjadi blok x_i dengan rumus

$$x_i = y_i^{SK} \bmod r$$

5.2 Algoritma DH (*Diffie-Hellman*)

Diffie Hellman (DH) dianggap merupakan algoritma kunci publik yang pertama kali ditemukan. Algoritma ini memperoleh keamanannya dari sulitnya menghitung logaritma diskrit pada bilangan yang sangat besar.

Algoritma DH cukup sederhana sehingga bahkan siswa sekolah menengah pun dapat memahaminya. Matematika fundamental yang digunakan meliputi ajabar eksponensial dan

modulus aritmatik[11]. Secara umum algoritma ini bekerja sebagai berikut. Pertama-tama, A dan B menetapkan dua bilangan prima yang sangat besar (misalnya berorde 2^{2048}) *n* dan *g* untuk digunakan bersama, sedemikian sehingga $g < n$ dan *g* adalah relatif prima terhadap *n*.

Kemudian proses berlanjut :

1. A membangkitkan bilangan acak $x < n$. Nilai x ini merupakan kunci privat bagi A. Tak ada orang lain yang boleh mengetahuinya. Kemudian A menghitung

   $$X = g^x \bmod n$$

   dan mengirimkan nilai X ke B. X merupakan kunci publik bagi A.
2. B membangkitkan bilangan acak $y < n$. Nilai y ini merupakan kunci privat bagi B. Kemudian B menghitung

   $$Y = g^y \bmod n$$

   dan mengirimkan nilai Y ke A. Y merupakan kunci publik bagi B.
3. A membangkitkan kunci simetri yang akan digunakan sebagai kunci IDEA (pada PGP)

   $$K = Y^x \bmod n$$

   Perhatikan bahwa A menerima kunci publik B, Y dan memiliki kunci privat x
4. B juga membangkitkan kunci simetri yang akan digunakan sebagai kunci IDEA

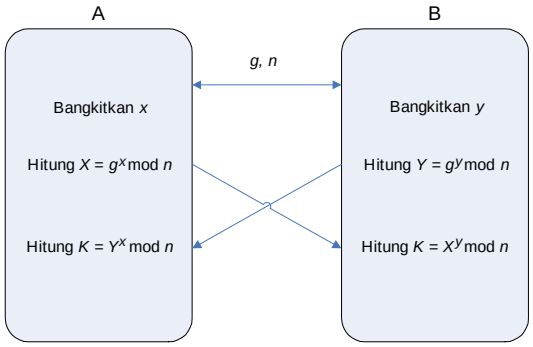
   $$K' = X^y \bmod n$$

Jika perhitungan benar maka K dan K' akan bernilai sama.

$$\begin{aligned} K &= Y^x \bmod n \\ &= (g^y \bmod n)^x \bmod n = (g^y)^x \bmod n \\ &= g^{yx} \bmod n = (g^x)^y \bmod n \\ &= (g^x \bmod n)^y \bmod n \\ &= X^y \bmod n \\ &= K' \end{aligned}$$

Lawan dapat menemukan *n*, *g*, X dan Y dengan mudah karena memang tidak diamankan, Namun untuk mendapatkan K yang akan dijadikan kunci penyandi data, diperlukan pemecahan x dari persamaan $X = g^x \bmod n$, dan menemukan nilai y dari persamaan $Y = g^x \bmod n$. Masalah ini disebut algoritma diskret yang dipercaya sangat sulit dilakukan bila nilai X, Y, *g*, dan *n* sangat besar misalnya berorde 2^{2048}.

Algoritma pertukaran kunci digambarkan sebagai berikut :



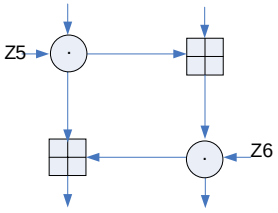
Gambar pertukaran kunci DH

6. Analisis keamanan algoritma IDEA, CAST, RSA dan DH

6.1 Analisis keamanan algoritma IDEA

Dalam melakuan perancangan algoritma IDEA diperhatikan beberapa prinsip sebagai berikut :

- Kunci cukup panjang : 128 bit
- *Confusion* : menggabungkan 3 kelompok operasi aljabar sederhana (xor, penjumlahan mod 2^{16} dan perkaliaan mod 2^{17})
- *Diffusion* : diberikan oleh kotak MA (*Multiply-Add*)



Gambar kotak MA (*Multiply-Add*)

dimana  adalah operator jumlah dan  adalah operator perkalian mod 2^{17}. Model ini merupakan jumlah komponen minimum yang diperlukan untuk menghasilkan difusi.

- Keserupaan antara enkripsi dan dekripsi, perbadaan hanya pada *key-schedule*
- *Scalable* : Versi mini (2 bir, 4 bit, atau symbol 8 bit) dapat dipelajari untuk analisis kelemahan
- Transparan : tidak tergantung pada table "misterius" kotak –S yang sering dicurigai berisi "jalan belakang"
- Cepat pada *software* dan *hardware*

- Sedapat mungkin dapat dibuktikan “aman”, misalnya “aman sempurna” dalam satu ronde dengan kunci sekali pakai.

IDEA kuat menghadapi kriptanalis diferensial maupun linear tetapi memiliki satu kelemahan utama yaitu banyak kunci lemah (2^{51}) tetapi karena ada 2^{128} kunci yang dapat dipilih maka kemungkinan memperoleh kunci lemah antara 2-77.
Sampai saat ini yang kemungkinan dapat membobol keamanan IDEA adalah *brute force attack* dan inipun saat ini mustahil dilakukan.

6.2 Analisis Keamanan algoritma CAST

Algoritma CAST memiliki tingkat keamanan yang tinggi dikarenakan hal-hal sebagai berikut[4] :

- Adanya komponen s-box yang dirancang dengan prosedur matematik yang menghasilkan kotak substitusi dengan komponen kriptografi yang penting seperti tingginya *nonlinearity*, rendahnnya distribusi perbedaan XOR, pengurutan yang baik dengan menggunakan kriteria bit yang terpisah.
- Penggunaan kunci “*masking*” dan kunci “*rotation*” menaikkan entropi kunci dibandingkan dengan entropi data pada setiap putaran. Hal ini menyebabkan sulit dilekukannya serangan statistic yang bersifat iteratif.
- Gabungan operasi dari kelompok aljabar yang berbeda (penambahan modulo 2 dan penambahan / pengurangan modulo 232) yang muncul mengakibatkan algoritma ini efektif tidak hanya dalam mengurangi probabilitas diferensial antar putaran tetapi juga mengurangi kemungkinan untuk serangan dengan deferensial tingkat tinggi
- Jaringan feistel yang digunakan dan variasi fungsi untuk tiap putaran mengakibatkan susahnya konstruksi yang bersifat iteratif.

6.3 Analisis keamanan algoritma RSA

Keamanan algoritma *RSA* terletak pada tingkat kesulitan dalam memfaktorkan bilangan non prima menjadi faktor primanya, yang dalam hal ini $r = p \times q$.

Sekali r berhasil difaktorkan menjadi p dan q, maka $\phi(r) = (p - 1)(q - 1)$ dapat dihitung. Selanjutnya, karena kunci enkrispi *PK* diumumkan (tidak rahasia), maka kunci dekripsi *SK* dapat dihitung dari persamaan $PK \cdot SK \equiv 1 \pmod{\phi(r)}$.

Penemu algoritma *RSA* menyarankan nilai p dan q panjangnya lebih dari 100 digit. Dengan demikian hasil kali $r = p \times q$ akan berukuran lebih dari 200 digit. Menurut Rivest dan kawan-kawan, usaha untuk mencari faktor bilangan 200 digit membutuhkan waktu komputasi selama 4 milyar tahun (dengan asumsi bahwa algoritma pemfaktoran yang digunakan adalah algoritma yang tercepat saat ini dan komputer yang dipakai mempunyai kecepatan 1 milidetik).
Algoritma yang paling mangkus untuk memfaktorkan bilangan yang besar belum ditemukan. Inilah yang membuat algoritma *RSA* tetap dipakai hingga saat ini. Selagi belum ditemukan algoritma yang mangkus untuk memfaktorkan bilangan bulat menjadi faktor primanya, maka algoritma *RSA* tetap direkomendasikan untuk menyandikan pesan.

6.3 Analisis Keamanan algoritma DH

Keamanan sistem Diffie-Hellman bergantung pada asumsi mudahnya menaikkan angka untuk perpangkatan tertentu tetapi sulit untuk menghitung pangkat yang digunakan jika diketahui angka dan hasilnya. Contohnya sangat mudah untuk menghitung 2^{10}=1024 namun sulit untuk menghitung bahwa 1024 adalah pangkat 10 dari 2.

Pengganggu mungkin saja mengetahui g^a dan g^b tetapi tanpa mengetahui a dan b akan sulit untuk menghitung kunci dalam waktu yang realistis. Untuk ,elakukannya diperlukan kalkulasi terhadap algoritma g dari g^ayang ditulis sebagai $\log_g(g^a)$ dan menghitung algoritma tersebut membutuhkan waktu yang sangat lama[12].

7. Perbandingan PGP algoritma RSA-IDEA dengan PGP algoritma DH-CAST

Pada PGP versi 9.5 tersedia banyak pilihan algoritma yang akan digunakan untuk membangkitkan kunci dan untuk mengenkripsi pesan. Algoritma yang didukung bukan hanya RSA, DH, CAST dan IDEA tetapi juga menyediakan algoritma AES, TripleDES dan Twofish.

Sedangkan pada PGP versi 6.5 pilihan algoritma yang digunakan untuk pembangkitan kunci hanya 2 yaitu algoritma RSA dengan IDEA

sebagai algoritma kunci simetri dan HD dengan CAST sebagai algoritma kunci simetrinya

Gambar tampilan menu pilihan algoritma pada PGP versi 9.5

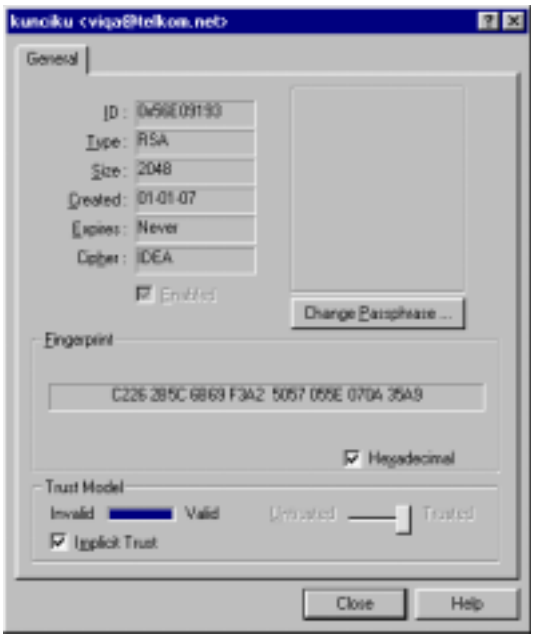
Gambar tampilan menu pilihan algoritma pada PGP versi 6.5

PGP versi 9.5 yang bersifat freeware tidak menyediakan lagi fitur pengenkripsian pesan dengan mode teks. Pengenkripsian pesan maupun pemberian tanda tangan digital dilakukan langsung dalam bentuk biner karena itu sulit untuk membandingkan hasil keluaran algoritma RSA dengan algoritma yanglain. Karena itu pada makalah kali ini dilakukan perbandingan dengan menggunakan PGP versi 6.5.

Sidik jari yang dihasilkan dengan algoritma IDEA-RSA :
C226 2B5C 6B69 F3A2 5057 055E 070A 35A9

Sedangkan dengan algoritma CAST-DH adalah

AB6E 8E14 088A 3769 3784 0EE2 8B7A 1DA8 817E 7031



Gambar tampilan properti kunci PGP versi 6.5

Kunci publik yang dihasilkan dengan menggunakan algoritma RSA maupun algoritma DH adalah sebagai berikut

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGP 6.5.3

mQENA0WY0dkAAAEIAMzupeGq0ZHQRcGd7PEjfZOgT5fz9E9nyWr7pMF7nihPHCJZ
7bV5K3BBT5cqjprj8QpAD+IubUKtuxXwKw/CbF3MNvke2kUiXns43a8BvmcM77AO
JBwE4J/GR2vDT7crFJkx9zcKOJOq35QziYnleXD4WI2WCa7AWath8Npi5Y1iCyEe
5scihBpljS34KeNMbPFPoNMlNGHfR7eJVD7jejJ2Uj1G5dO6AdJjduVwA95jqxQf
odqwpwrNAj2KPyU0gNNgCtvnxFQIBL9LKMTzNP+fwfs82HWXSPnSrIMIHJnRcH9K
ePpBKx5shfq5oxKGuYbqNGzxprKMu/J5SVbgkZMABRG0GWt1bmNpa3UgPHZpcWFA
dGVsa29tLm5ldD6JARUDBRBFmNHZu/J5SVbgkZMBAa0cCACDY2uDtGvSS5PagAaj
OzWBFh8G6aiPg6XmUzQxVoDwNhG4z/0HzFsQx6q0SC8fRSonq/QLuqrEeobUn1LU
0I6CYnfh9eh3TIJ3KAWKWliJn985arJ5GPIOENr4J6pvEy17vE/wEdRcU1CWHLng
1lwP5Hxg3bAX6QetZi+3e8DzvRQOX1hP9gi87+z5sXp4/aKK67U4TNQpMuuV3HVC
bfkkTGKUj+0f+RMaolxruCXzV9nBDSVVj0TVwbUQLhoLcdzAGqGhVRaQOxmUFj+0
iJ0qcWkES8VdqudxAkqGkNEbKUTiBCVvcxPrPg8NARQqkPQgLFMyHj2mRHtjF5sA
Ax6B
=qp+T
-----END PGP PUBLIC KEY BLOCK-----
```

Gambar kunci publik algoritma RSA

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGP 6.5.3

mQGiBEWY0VoRBADSrQe1/Q05shLggdyKGzPMhKU3F290rY40G7qvc+fyYdyuXfG2
pJLUrMLDACTcsWT9lc0k135u2yBjuTUZw/jWRDA7hSqlrAgk8PFLJeSrYpPso53+
8h6u5JuRqP10wIoWnE/F2h1aJfCTGuCv+grZuoJs3XTvQ5Sd/zXt7tmdTwCg/wkX
3jtZqAwBjQ1o/3Am6cIkDfcD/0OV3CjwJ73RFC2evIlIcWViRQr81jXz6kydAfS0
ocQ/RAd9P8lO8pt5Z44ybY3+6tez/PPwryOywXbjaRZu5ITZ0b+LlFwfNN+Mk+nh
Nvushkuc6ha1LVJs6rwzj6B0Q+w7zB8pZDpQ5RHSuziCSfjv3TDzl1K07fSa5w9z
diycA/9IpbHFtEkDDKz9uNkoDZkB+u3UjBSni7rr+NBxEgPrJ8e5IbtBMcHBIouc
HRb9qT2oN8nyc2wNO4rQbzdJagNa+BXldWedPXBslv2UiT2iFfqlPuoZ9WMvDzig
JzQBYDgO50wfVMHfZz1CXC18xuZcSZkECrAVJfvTioXSWpc1L7QZa3VuY2lrdSA8
dmlxYUB0ZWxrb20ubmV0PokATgQQEQIADgUCRZjRWgQLAwIBAhkBAAoJEIt6HaiB
fnAx6IUAoMJ5IShMyyA+yMxEwbuAjHm5MENSAKDCuPnKSHVvFbiFf6FV4A3qMEsZ
KrkCDQRFmNFaEAgA9kJXtwh/CBdyorrWqULzBej5UxE5T7bxbrlLOCDaAadWoxTp
j0BV89AHxstDqZSt90xkhkn4DIO9ZekX1KHTUPj1WV/cdlJPPT2N286Z4VeSWc39
uK50T8X8dryDxUcwYc58yWb/Ffm7/ZFexwGq01uejaClcjrUGvC/RgBYK+X0iP1Y
TknbzSC0neSRBzZrM2w4DUUdD3yIsxx8Wy2O9vPJI8BD8KVbGI2Ou1WMuF040zT9
fBdXQ6MdGGzeMyEstSr/POGxKUAYEY18hKcKctaGxAMZyAcpesqVDNmWn6vQClCb
```

```
AkbTCD1mpF1Bn5x8vYlLIhkmuquiXsNV6TILOwACAgf/dChBQK5yDLZEg5EhHSg4
90COI6pAlFpThrHjUjZ1jSXwwq78sIjUwG7B5eyL06wKaTUI2MC8dGEI2AhUDYFB
pfl7VB9sfQ4t0gmrWetuHE5cQp+xNQW/mTz6Bw03KdOh7CNM3daqQfjZDTgvpq9U
uKj1ANfeQGBOCsmpLCMuIwjtYb5AcrvUEKdKhuhBRzp+EvLbp8Tt+rDGY9wHtNRE
7NdBhNyHnvCWD0KCUY76Fv1fGdl6ix2uRNDpuEXLSJPKxwjzqCoTYuJwblUOx0bV
785SeoV6zuVi1pshN7L+LLmYE8wW6irO+Br7mLsIXW/5vZBMyQ4nPM7zhAHzay8q
mYkARgQYEQIABgUCRZjRWgAKCRCLeh2ogX5wMfjOAKDkbJnr2YldI+szjEbg95Cn
1p9MGACgnx/Ii2abxTT57hW7QoF2ZqvNn7I=
=jQQk
-----END PGP PUBLIC KEY BLOCK-----
```

Gambar kunci publik algoritma DH

Sidik jari dan kunci publik untuk kedua algoritma dapat digunakan berulang kali dan bersifat tetap akan tetapi pada proses pembangkitannya meskipun menggunakan nama dan email yang sama sidik jadi dan kunci publik yang dibangkitkan akan berbeda. Hal ini dikarenakan adanya kunci sesi yang dibangkitkan pada setiap kali membangkitkan kunci publik.

Berikut adalah gambaran tanda tangan digital dan gambaran hasil enkripsi pesan yang dihasilkan untuk menandatangani dokumen yang sama dengan menggunakan algoritma RSA dan DH.

```
-----BEGIN PGP SIGNATURE-----
Version: PGP 6.5.3

iQEVAwUARZjS/LvyeUlW4JGTAQFYPggAgODQkh2Es/Q4+UFEXtvhS1JsKvKqGlc6
VwqJJcA2rY/p0IfV0QwH0Ot0NkqO2PeaO++u+9VdUF+hRQkf6jUkdbLd2kbUOnXY
5h698TX2ZTydSLPRd/d46ZFWny+iq1Sq9v+KC3AKgoTRgD0OLyJ7bgS33pOiRSsl
uoraTTllH1NsVdmP6UQmLFo/ccEsh15fRrOGDbsApCiOvQFqv5OupQMS+L14mwTi
KbRu8X2iCr+VT6BfPmZXzqj+Yli5QPJV5P6zkdFMj4IH4WsxwNrVpzj1Zc35hEdK
QMA7vNnb8svUlbM4VIjxGtYck2Gt5bx3Ln3R4Bkmp6HIrhgqFdOsug==
=LzJW
-----END PGP SIGNATURE-----
```

Gambar tanda tangan algoritma RSA

```
-----BEGIN PGP SIGNATURE-----
Version: PGP 6.5.3

iQA/AwUARZjTdot6HaiBfnAxEQJZbgCg4fMElF0m0M3ynp99JFcG4Zegg6oAn3iV
YR+N2MPAHzZZU6RYjZ6r1pKf
=dRmP
-----END PGP SIGNATURE-----
```

Gambar tandatangan algoritma DH

```
Di sebuah dataran hijau yang luas hiduplah makhluk lucu yang
merupakan hasil persilangan antara alien dan manusia. Makhluk itu
dikenal dengan sebutan si Bong.

Si Bong merupakan makhluk hijau lucu yang memiliki sungut seperti
kupu-kupu. Makhluk yang tidak perna bersedih dan selalu
tersenyum. Wajah dan bentuk badannya nya seperti alien tetapi
memiliki otak dan perasaan seperti halnya manusia. Seperti halnya
smurf dalam komik smurf yang hanya dapat mengucapkan kata smurf
untuk menggantikan kata apapun, sibong hanya dapat mengucapkan
kata bong... bong... untuk menggantikan kata apapun.
```

Gambar plainteks

```
-----BEGIN PGP MESSAGE-----
Version: PGP 6.5.3

hQEMA7vyeUlW4JGTAQf/U+5Jcqjphej+7u200y2mzf7XKB/8YhfAOij6Me5k5IMi
tJqMeIxnfLc0CM2vUkAG6LjyTaoppooc8iQBjcEGd6vA9n1nw7bjF7X85+eoutk3
w3j7IJ/hf8DhuYWMqF5blMG3qx8998st3oVdorQtxbbX4o1am/FC666suRgEURzf
DuS+VZdds2NbTDWIhQmbbxzw+z86KQRsgnrfyokUoNJMeUdlKyTNG6h19EoOAirS
vfflXHcs5lhBFDEASTgq6u0+t5IJ63OgS5B+pFo77AGBnoyjBeb0WcL1mLNjZTty
9RKRdyLo257C5nQHykt0CApzX50pD2/4KzMSrMo1vIUCDgM4odfbtyS5nRAIAKv1
xBmB35MqQxGdCNlMwn9FRG9TnEPI4OmpfpVwjOEIRB1A+0xgXdxu+Z5iBx0uuBul
H0wDWy8EHa/P2Dnmrc3AdtCnxRwYEGGBSGW7ohizv/gXxJFvinplshgTaPxQp3ya
Bk/sWrirTgh6WIa1SuJh2fZRgXgrvgCykZkSEBil4s4Zm7qTTqzK/jpEddZIlE95
52tO9BHIxr1WHaPG7NF9rPHLL8pX8NTfhJwIjZld6raoq+SXTGJtit9urU1Zw+D+
lZKVmJi6EAGvzviNQO9TDad8P7E+LhFzN8VbcLbYO3cgonFfiWgZOuKnaPFAfuLx
WuJJ2G1oVtcWL/Qc59AIAO5XKs7tekLZZEyv4mAHi26F26Q9fGiOyqLcm2alGe6m
CIm5WL2YKaGaEGu+2GD9iZj9f/l0nzBmi/GYjnPxAQZlWHjoKcLYf8PNVDRXGf3v
7NDkMpiFtUe8pNz7HePDApm9KZ59ljs76SO1Q8ke6KhP3BFrrmfkyi48+T+Xl+Kg
lgb7MxSXKOLxGOHkBxZC8lSgLEOOxAUSYxAZH7Xelj7bp560bcq0FREig6o+dx+3
zHaxKGE4piJln6ESIgAN8lr05TyI0yQq97Dr6bw9ChtqxrVlWrWKff8AwlCkojxC
dUXh8Y5SF+rpcWMBLHKEBVeKtoPAnJAN6i3WnySvARulAqRgHq0McYxdiTM1iR0P
46QLa6fq4x1v1QJ2IPX7TUuat0L/AIGOMWe0qgDVUSDhlMRqCNO2xidnllg4zZlo
6TuMHcxPPh09HXKlYy62TSVFw8mDPAONWcoU1Ax1AmNE90LvfmfGJFVgns8/pT4/
oM9BuVHCMfPFB7wwomK5m4qTZsW94jpIhnJZqQtl7kp0V1pxb79FghWrknmqtj+p
rS+RfhY2P71CrFpLifqOFzxskrWYO4ODNBo0vpId/ROMOwicukLh/1ZxAYGt7A1U
HGoJ4NGj1WaqEFuY5lPvHT9uHWHPzcpbOvYaGrXeuCIu5YREeNqFwGicpLPERQm0
5KFAeo2mcL2MDQr/kei+RT0TIQtcGDlSfU5GzccnhWZrQbnUZoSxsPb/gEAvy+Oc
qiLFj9hK6TjiDMqBv9QdSzZtTNWr7YcQruupUwbV4y5UnFekxm5l9AmET8WbADh3
DAoWfTb6VVfDCS5beB9Be71hLVgWyC82jDZ3ZPYN+HVpZKm2vs+r0vhjgFh+brZP
TwOH7avmQ4g1QffyrA1cKvM2FkWYzJWjGKukUB7gEZPi844kVHjiQy0RJvRpysHF
7byHCkLUaesVtD7cPQcfy3d6nnH5rn741mbolkvLdYMm513VjURc4TJxzz0vI6bu
GAXZL0p7ev46RLxSWHDg9bz2DojUTZkRoGNY8+NPoja0DUwVc/vB3bgsD+MCJLKP
X84bjJdAapqQW0kJHPfnXkZVCCl/l2AgRF4eY2h8Wdmi1Hmq8+ayvLZOhnyTwvV4
kJlt17mkSVzGRHL01Czrodz2otk83r6olc1QRD9xJs6XHyYCiKeuRNRng3iZTvpg
pIa+hcfHInSTYwZ/oypOm0kUgMeFh3ScZWdOTSdhodXR/S0fzYGk
=RNdg
-----END PGP MESSAGE-----
```

Gambar cipherteks dengan algoritma RSA

```
-----BEGIN PGP MESSAGE-----
Version: PGP 6.5.3

hQEMA7vyeUlW4JGTAQgAhRN2xo5uI9n/EVKuXu9bmHMfcWI91WHM5e5Y/edm3Vv0
yDN7SstqKe8OaUe0dqatV/i/+W+FzCrcD7eoX6g0pOZb0kz7LjX/v5pIB64H2GcT
tVZDoEfgEUkW32vlex/oLz+pxPzs0i+At0pld81q3Zu+zYjPfbyJ/nIAolaoIwoH
8v7ywi6LTJAiwnZoXY77TW7vpMtZYopzDTqpeL9KxvM2tV+8+juy2hBprCqq941Z
ZyLMijfXzA8EUCyuamjVS+nETEhC7e8KL9AD8oGtJL+15XihqbcWsoZMv4Z1yXUi
z32p1vgSXwSvEVfHSk9c1mJT5G/LtV46Pr+UGP4C4IUCDgM4odfbtyS5nRAIAKcb
78yNl7XneHilc4lE/9Bt+JuRDmNTspZluS85tVU6XKKr0ebLROs2RyJDk2NRR6qp
NJ/pKg7EC2NH9H8M/fbBANpvwnCWBizayLYhP0rixIm5FZYvUNuesQJQHYKv4QMD
4gGVidaDis7jm5D6IdghoviDWRptQt4OZoZn7D5WDNI3nj+3lkaue3L+8100jKFr
klGozBN7rN1bCPY6PbBkByXjHbmfCmzg/R5kEeFDYFrTM2XgFVqDN499ArYXJ/f7
ZbP92H7jdidbPyRV1dJEh/BKq+3AOrJZTeYLrGupYBHzA/ZByZQbE3h2gw9mLoA/
8lxVxuH8/sF+CsWG0XoIAKQICRN1DGbb757alK0ehJwPrO3oXWf8PqxlCkvTmWS0
24lhFpfXS1svyAY8FOAHODUlRi9wyXUbDXdjofN6jRTbnNYzIG5cDQ6HXnqNaFoH
JctzMikaGXqq1AzoG7pULn3nEOag2fiT3xaEny/N/Kb3LhZ/+bs/HYahbiOphEc+
J3fh2YiBNvn8CPMYWXlgv7RpRDmVydpz8DS5LfMV+YjCTadpMXltznyoMI4IMAFa
ok491R5p7Z6krGQptiHO+BVR26lg9mhNiXSfthYgAuUFqbUrDO3sTY7W+Kb+WcGw
/0ZDdiDKujkIcNJg+xAPNju2DfyqdCVTRPU+Mrp/t+ClAaVYzZsfe1+32mjkbS0g
Uewtj8KIbN5QbOPX6nbOcjBr11p+a+jQfF6tjN96Gc7UKk/qkfMFPgMAHj8D/0B0
xLxLHDX66a/HNEWS6WfZlwYv7nMHBeaZQ7U7RT/r/lmQaYGrzhebGw0i7biedUNt
qx5L8W9rfipk+6sGwRHpB0ISg4iRMrTFupmQC9Y7SsaQ0s/TF1rbXpvoSZK/mJuT
Z9GLM5uvPiKMqpp5ZFl5MA33nFpERZP7JQm9ilJIRIzENly1v1psIokXu+YfiaCu
SjjScVwO7nWr/vKYdJzgOvgsHAPtv1MwU9vojqAv1xR4BuXp+3uVcon8YhNNfxPu
VqDSheuLSyVQ7JgUZViGoMtJiQxdunNthvuyXvyOjyZJWF77m4O3yLmODMusij8d
6RkkRzy1io1AWhH3p6P39T2ENVZ0a4jAR2HJSI/VjCFahicOpedxHzmCeofIDRUd
/ukCdg9yZPRxva7fxSewdo+OUWlYAzWJV7iUJy1iIjZ+lUGZNrfWHnobU2Svlr9L
cBKI5vMytLNfQENCH0lvAM7kU576JRCB
=fFGz
-----END PGP MESSAGE-----
```

Gambar cipherteks dengan algoritma DH

Seperti halnya kunci publik maupun sidik jari, hasil tanda tangan maupun hasil enkripsi selalu berubah meskipun digunakan pada file yang sama dan dengan kunci publik yang sama. Hal ini dikarenakan adanya kunci sesi yang dibangkitkan oleh PGP secara otomatis. Namun setelah dilakukan beberapa kali percobaan

diperoleh kesimpulan bahwa beberapa digit awal hasil enkripsi maupun tanda tangan untuk algoritma apapun tetap dan tidak berubah.

Dari gambaran di atas dapat dilihat bakwa panjang kunci maupun sidik jari yang dibangkitkan oleh algoritma DH lebih panjang dibandingkan panjang kunci yang dibangkitkan oleh algoritma RSA. Namun pada saat enkripsi maupun pembangkitan tandatangan algoritma RSA menghasilkan tandatangan dan cipherteks yang lebih panjang.

8. Kesimpulan

Kesimpulan yang dapat dimbil dari studi perbandingan antara PGP dengan algoritma RSA-IDEA dan PGP algoritma DH-CAST diantaranya adalah :

1. Algoritma IDEA kuat untuk menghadapi kriptanalis diferensial maupun linear juga tidak memiliki s-box sehingga tidak mungkin memiliki jalan belakang tetapi algoritma ini memiliki kelemahan yaitu memiliki kunci lemah.
2. Algoritma CAST memiliki keamanan yang tinggi karena menggunakan kunci *masking* dan kunci *rotation*, menggunakan gabungan operasi aljabar yang berbeda dan menggunakan jaringan feistel. Meskipun demikian algoritma ini dicurigai memiliki jalan elakang karena menggunakan s-box.
3. Keamanan algoritma RSA bergantung pada kesulitan memfaktorkan bilangan besar menjadi bilangan prima.
4. Keamanan algoritma Diffie-Helman hanya terletak pada sulitnya mencari pangkat jika diketahui akar dari suatu bilangan.
5. Algoritma PGP IDEA-RSA memiliki keunggulan karena kuatnya algoritma RSA dan dibandingkan DH. Sedangkan algoritma PGP CAST-DH memiliki keunggulan pada algoritma CAST yang lebih unggul dibandingkan algoritma IDEA.
6. Pada PGP dengan mengasumsikan bahwa keamanan algoritma dilihat dari panjangnya kunci maka algoritma CAST-DH dapat dianggap lebih unggul disamping karena alasan lebih efektif dalam mengenkripsikan pesan karena menghasilkan pesan dan tandatangan yang lebih pendek dibandingkan algoritma IDEA-RSA.
7. Karena secara umum algoritma kunci publik RSA dinilai lebih aman maka algoritma IDEA-RSA sebaiknya digunakan untuk membuat tanda tangan digital karena pada tanda tangan digital tidak diperlukan algoritma kunci simetri yang kuat. Sedangkan untuk mengenkripsikan pesan sebaiknya digunakan algoritma CAST-DH karena algoritma CAST merupakan algoritma kunci simetri yang kuat dan sebaiknya digunakan untuk menenkripsikan pesan.

DAFTAR PUSTAKA

[1] Munir, Rinaldi. (2004). Bahan Kuliah IF5054 Kriptografi. Departemen Teknik Informatika. Institut Teknologi Bandung.

[2] Kurniawan, Yusuf. (2004). Kriptografi Kemananan Internet dan Jaringan Komunikasi. Penerbit Informatika.

[3] http://www.davidyaw.com/crypto/PGP.pdf
Tanggal Akses : 14 Desember 2006 pukul 08.00

[4] http://www.pgpi.org/doc/pgpintro/
Tanggal Akses : 14 Desember 2006 pukul 08.00

[5] http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci214292,00.html
Tanggal Akses : 14 Desember 2006 pukul 08.00

[6] http://www.wowarea.com/english/help/swpgp.htm
Tanggal Akses : 14 Desember 2006 pukul 08.00

[7] http://jim.willingham.com/pgppg.htm
Tanggal Akses : 14 Desember 2006 pukul 08.00

[8] http://www.pgp.com/newsroom/mediareleases/2005/
Tanggal Akses : 14 Desember 2006 pukul 08.00

[9] http://www.finecrypt.net/idea.html
Tanggal Akses : 27 Desember 2006 pukul 20.00

[10] http://www.mirrors.wiretapped.net/security/cryptography/algorithms/aes-testing/cast/cast-256.pdf
Tanggal Akses : 27 Desember 2006 pukul 20.00

[11] http://www.sans.org/reading_room/whitepapers/vpns/
Tanggal Akses : 31 Desember 2006 pukul 20.00

[12] http://www.iusmentis.com/technology/encryption/diffie-hellman
Tanggal Akses : 31 Desember 2006 pukul 20.00