**DOKUMENTASI ALGORITMA KRIPTOGRAFI**

**(UJIAN AKHIR SEMESTER)**

**MATAKULIAH KRIPTOGRAFI**



Disusun oleh :

|  |  |
| --- | --- |
| Wahyu adjie prasetyo | (2015.81.025) |

Dosen pengampu:

Tri ismardiko widyawan

**FAKULTAS ILMU KOMPUTER**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**UNIVERSITAS ESA UNGGUL**

**JAKARTA**

**2018**

**DAFTAR ISI**

**Halaman**

HALAMAN JUDUL i

DAFTAR ISI ii

KATA PENGANTAR v

**BAB I PENDAHULUAN 1**

1.1 Algoritma AES 1

1.2 Algoritma DES 6

1.3 Algoritma MD5 7

1.4 Algoritma TRIPLE DES 7

1.5 Algoritma RSA 7

1.4 Algoritma SH-1 8

**Bab V PENUTUP………………… 109**

5.1 Kesimpulan……. 109

**KATA PENGANTAR**

Puja dan puji syukur saya haturkan kepada Allah Subhanahu Wata’ala yang telah memberikan banyak nikmat, taufik dan hidayah. Sehingga saya dapat menyelesaikan makalah yang berjudul “Dokumentasi algoritma kriptografi” dengan baik tanpa ada halangan apapun.

Makalah ini saya selesaikan untuk memenuhi ujian akhir semester yang bersifat takehome yaitu matakuliah kriptografi, makalah ini berisi tentang pentingnya kriptografi dalam dunia keamanan digital yang perkembangannya semakin pesat dengan seiring kemajuan teknologi digital saat ini diluar itu, penulis sebagai manusia biasa menyadari sepenuhnya bahwa masih banyak kekurangan dalam penulisan makalah ini, baik dari segi tata bahasa, susunan kalimat maupun isi. Oleh sebab itu dengan segala kerendahan hati , saya selaku penyusun menerima segala kritik dan saran yang membangun dari pembaca.

Dengan makalah ini saya berharap dapat membantu menyelesaikan ujian akhir semester saya dengan sukses dan memberikan edukasi kepada pembaca bahwa keamanan data saat ini sudah menjadi hal yang perlu di perhatikan oleh masyarakat dengan seiringnya kemajuan teknologi digital yang semakin maju selain itu saya berharap makalah ini juga memberikan manfaat lebih bagi developer atau pengembang aplikasi untuk selalu memperhatikan keamanan data *user.*

Demikian yang bisa saya sampaikan, semoga makalah ini dapat menambah khazanah ilmu pengetahuan dan memberikan manfaat nyata untuk masyarakat luas.

Jakarta, 20 January 2018

Penulis,

**PENDAHULUAN**

1. **Latar belakang**

Kriptografi merupakan cabang ilmu yang memperlajari tentang seni pengamanan data *(enkripsi)* yang merubah teks *(plaintext)* dengan sebuah kunci *(key)* menjadi karakter yang tidak dapat baca oleh manusia *(user)* dengan seiringnya kemajuan teknologi digital seperti saat ini kriptografi merupakan hal yang wajib di perhatikan oleh pelaku bisnis yang mengedepankan teknologi dalam pengembangan bisnisnya misalnya seperti e-commerce ,bank dan lain sebagianya. Kriptografi saat ini telah di kembangkan menjadi kriptografi modern, kriptografi modern di dasari pada teori pada teori matematis dan aplikasi komputer; algoritme kriptografi didesain pada asumsi ketahanan komputasional, membuat algoritme ini sangat sulit dipecahkan oleh musuh. Secara teoretis, sangat sulit memecahkan sistem kriptografi, namun tidak layak melakukannya dengan cara-cara praktis. Skema ini oleh karena itu disebut sangat aman secara komputasional; kemajuan teoretis dapat meningkatkan algoritme faktorisasi integer, dan meningkatkan teknologi komputasi yang membutuhkan solusi ini untuk diadaptasi terus-menerus. Terdapat skema keamanan informasi yang benar-benar tidak boleh dapat ditembuh bahkan dengan komputasi yang tak terbatas namun skema ini sangat sulit diimplementasikan. Saat ini telah banya jenis kriptografi yang bermunculan untuk mengamankan data pengguna*(user)* salah satunya seperti DES, RC5, IDEA, RSA dan masih banyak lagi algoritma kriptografi lainnya.

1. **ALGORITMA AES**

Algoritma kriptografi bernama Rijndael yang didesain oleh oleh Vincent Rijmen dan John Daemen asal Belgia keluar sebagai pemenang kontes algoritma kriptografi pengganti DES yang diadakan oleh NIST (National Institutes of Standards and Technology) milik pemerintah Amerika Serikat pada 26 November 2001. Algoritma Rijndael inilah yang kemudian dikenal dengan Advanced Encryption Standard (AES). Setelah mengalami beberapa proses standardisasi oleh NIST, Rijndael kemudian diadopsi menjadi standard algoritma kriptografi secara resmi pada 22 Mei 2002. Pada 2006, AES merupakan salah satu algoritma terpopuler yang digunakan dalam kriptografi kunci simetrik.

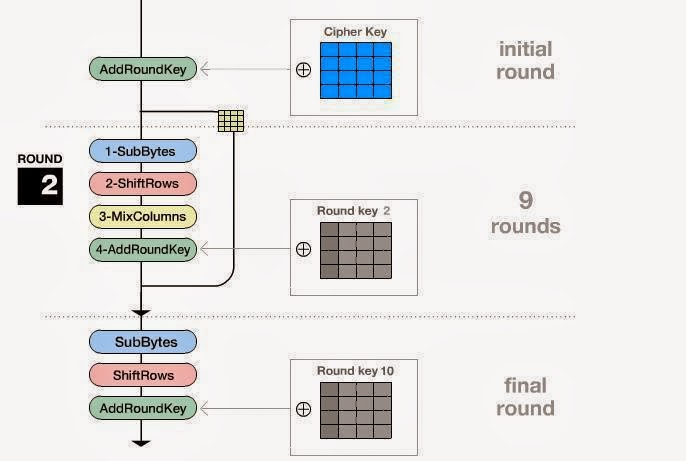
AES ini merupakan algoritma block cipher dengan menggunakan sistem permutasi dan substitusi (P-Box dan S-Box) bukan dengan jaringan Feistel sebagaiman block cipher pada umumnya. Jenis AES terbagi 3, yaitu :

1. **AES-128**
2. **AES-192**
3. **AES-256**

Pengelompokkan jenis AES ini adalah berdasarkan panjang kunci yang digunakan. Angka-angka di belakang kata AES menggambarkan panjang kunci yang digunakan pada tipa-tiap AES. Selain itu, hal yang membedakan dari masing-masing AES ini adalah banyaknya round yang dipakai. AES-128 menggunakan 10 round, AES-192 sebanyak 12 round, dan AES-256 sebanyak 14 round.

**Cara kerja AES**

AES memiliki ukuran block yang tetap sepanjang 128 bit dan ukuran kunci sepanjang 128, 192, atau 256 bit. Tidak seperti Rijndael yang block dan kuncinya dapat berukuran kelipatan 32 bit dengan ukuran minimum 128 bit dan maksimum 256 bit. Berdasarkan ukuran block yang tetap, AES bekerja pada matriks berukuran 4x4 di mana tiap-tiap sel matriks terdiri atas 1 byte (8 bit). Sedangkan Rijndael sendiri dapat mempunyai ukuran matriks yang lebih dari itu dengan menambahkan kolom sebanyak yang diperlukan. Blok chiper tersebut dalam pembahasan ini akan diasumsikan sebagai sebuah kotak. Setiap plainteks akan dikonversikan terlebih dahulu ke dalam blok-blok tersebut dalam bentuk heksadesimal. Barulah kemudian blok itu akan diproses dengan metode yang akan dijelaskan. Secara umum metode yang digunakan dalam pemrosesan enkripsi dalam algoritma ini dapa di lihat dari gambar di bawah ini



1. **ADD ROUND KEY**

Add Round Key pada dasarnya adalah mengkombinasikan chiper teks yang sudah ada dengan chiper key yang chiper key dengan hubungan XOR.

1. **SUB BYTES**

Prinsip dari Sub Bytes adalah menukar isi matriks/tabel yang ada dengan matriks/tabel lain yang disebut dengan Rijndael S-Box.

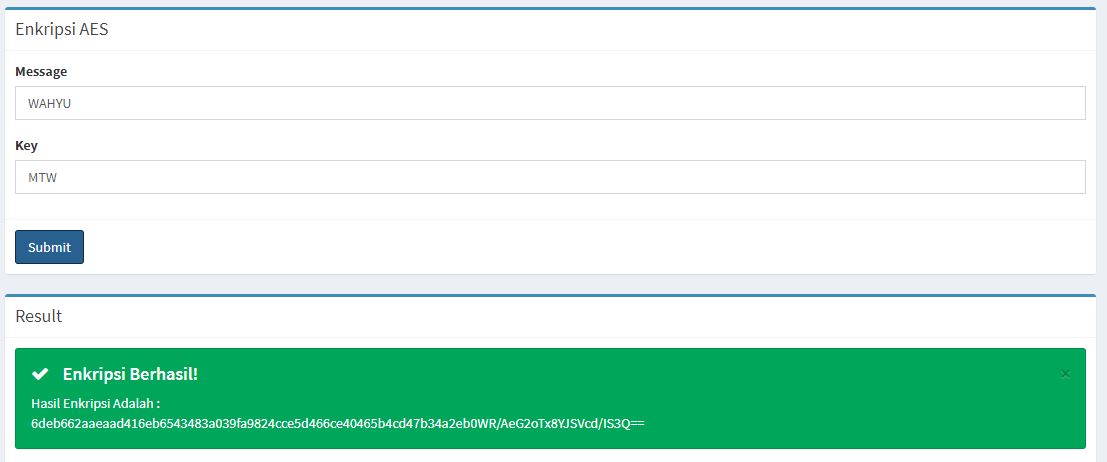
1. **SHIFT ROWS**

Shift Rows seperti namanya adalah sebuah proses yang melakukan shift atau pergeseran pada setiap elemen blok/tabel yang dilakukan per barisnya. Yaitu baris pertama tidak dilakukan pergeseran, baris kedua dilakukan pergeseran 1 byte, baris ketiga dilakukan pergeseran 2 byte, dan baris keempat dilakukan pergeseran 3 byte. Pergeseran tersebut terlihat dalam sebuah blok adalah sebuah pergeseran tiap elemen ke kiri tergantung berapa byte tergesernya, tiap pergeseran 1 byte berarti bergeser ke kiri sebanyak satu kali

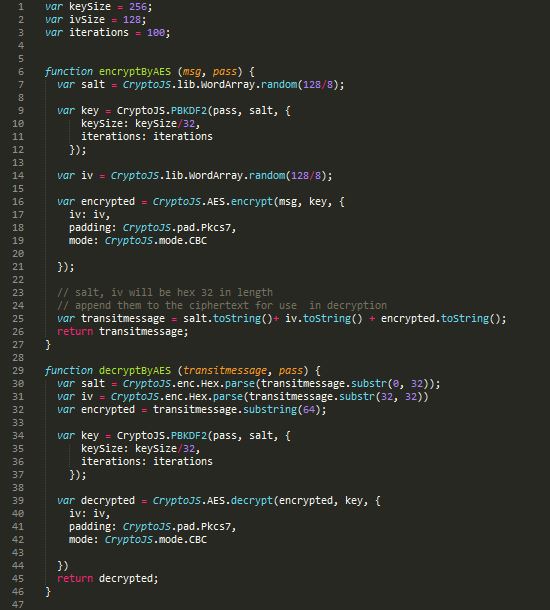
1. **MIX COLUMNS**

Yang terjadi saat Mix Column adalah mengalikan tiap elemen dari blok chiper dengan matriks yang ditunjukkan oleh Gambar 11. Tabel sudah ditentukan dan siap pakai. Pengalian dilakukan seperti perkalian matriks biasa yaitu menggunakan dot product lalu perkalian keduanya dimasukkan ke dalam sebuah blok chiper baru. Ilustrasi dalam gambar 12 akan menjelaskan mengenai bagaimana perkalian ini seharusnya dilakukan.

**Program enkripsi AES :**

****

**Source code AES:**

****

**Kelebihan AES**

1. Karena diimplementasikan pada perangkat keras dan perangkat lunak, protokol keamanannya paling kuat.
2. Ini menggunakan ukuran kunci yang lebih tinggi seperti 128, 192 dan 256 bit untuk enkripsi. Oleh karena itu algoritma AES lebih kuat untuk mengatasi hacking orang yang tidak bertanggung jawab.
3. Aes merupakan protokol keamanan yang paling umum digunakan untuk berbagai aplikasi seperti komunikasi nirkabel, transaksi keuangan, e-bisnis, penyimpanan data terenkripsi, dll.
4. Selain itu aes adalah salah satu solusi komersial dan open source yang paling banyak digunakan di seluruh dunia.

**Kelemahan AES**

1. Aes menggunakan struktur aljabar yang terlalu sederhana.
2. Setiap blok selalu dienkripsi dengan cara yang sama.
3. AES memerlukan mode counter rumit untuk diterapkan dalam perangkat lunak sehingga kinerja dan keamanan menjadi pertimbangan.
4. **ALGORITMA DES**

Algoritma DES merupakan algoritma enkripsi blok simetris. DES dikatakan enkripsi blok karena pemrosesan data baik enkripsi maupun dekripsi, diimplementasikan per blok (dalam hal ini 8 byte). DES dikatakan enkripsi simetris karena algoritma yang digunakan untuk enkripsi relatif atau bahkan sama persis dengan algoritma yang digunakan dalam proses dekripsi. Proses enkripsi dapat didefinisikan secara sederhana sebagai proses penterjemahan data “asli” yang “jelas” dan “kasat mata” yang dapat dipahami maknanya. secara langsung menjadi data lain yang terlihat “buram” atau “acak” sehingga tidak dapat dipahami secara langsung, sedemikian rupa sehingga makna informasi yang disembunyikan tidak lagi dapat diketahui secara langsung kecuali dengan mengembalikan informasi tersebut ke bentuk aslinya. Sedangkan proses dekripsi dapat didefinisikan secara sederhana sebagai proses pengembalian bentuk data, dari data “buram” atau “acak” menjadi data “asli” yang “jelas” dan “kasat mata” yang dapat dipahami maknanya. Algoritma enkripsi umumnya dilengkapi semacam kata sandi (password), untuk memvariasikan fungsi enkripsi tersebut. Data yang sama, kunci yang sama dan algoritma yang sama akan menghasilkan data enkripsi yang sama. Dalam algoritma penyandian DES, kunci yang digunakan dalam proses enkripsi dan dekripsi haruslah sama, supaya data dapat dikembalikan ke bentuk aslinya. Bisa jadi, karena “kesamaan” kunci inilah DES juga dinamakan algoritma enkripsi simetris. Inti dari proses enkripsi adalah penyembunyian data dengan mengaburkan data “asli” dan mengurangi keteraturan informasi, sehingga data tersebut tidak dapat “dibaca” kecuali oleh pihak yang berhak. Berbagai algoritma enkripsi sengaja dibuat untuk melindungi informasi dari penyadapan, karena ada kemungkinan terjadinya penyadapan saat data melewati media hantar (media hantar dapat berupa suara, surat, email, kabel, kertas, frekwensi radio atau apapun itu). Seandainya penyadap dapat menyadap semua informasi yang melalui media hantar, idealnya hasil sadapan tersebut hanya menghasilkan data “sampah” yang tidak berguna. Semua algoritma kriptografi diciptakan untuk mewujudkan kondisi ideal tersebut, tapi sayangya kondisi tersebut sangat sulit dicapai, karena selalu ada cara untuk membalikkan informasi sadapan ke bentuk aslinya.

**Fungsi DES**

Secara umum tujuan DES diciptakan adalah untuk merahasiakan suatu pesan agar tidak bisa dibaca oleh pihak yang tidak absah. berpedoman dari tujuan umum tersebut, sehinga pesan yang dikirimkan aman dan tidak dapat dibajak

**Cara Kerja DES**

Dalam DES, algoritma dekripsi tepatnya merupakan proses kebalikan (inverse) algoritma enkripsi. Dalam prakteknya proses pembalikan (proses dekripsi) ini diimplementasikan dengan membalikkan urutan sub kunci yang digunakan dalam proses enkripsi, selebihnya algoritma enkripsi dan dekripsi adalah sama. Algoritma enkripsi DES bekerja dengan mengolah blok data 8 byte (64 bit) dengan blok kunci 8 byte (64 bit). Proses penyandian dalam DES diawali dengan fungsi pengacakan bit yang dinamai IP (Initial Permutation) kemudian fungsi inti DES yang diulang sebanyak 16 kali dan terakhir ditutup dengan fungsi pengacakan bit lain yang dikenal denagn nama IP‐1 (Inverse Initial Permutation). Pada sisi lain algoritma penjadwalan sub kunci akan menghasilkan 16 sub kunci secara berurutan dari parameter kunci yang diberikan untuk digunakan pada setiap putaran fungsi inti DES. Sub kunci pertama untuk putaran pertama, sub kunci kedua untuk putaran kedua dan seterusnya hingga putaran ke 16.  
Perlu diingat, kendatipun slot kunci yang disediakan digunakan berukuran 8 byte (64 bit), ternyata pada faktanya ukuran kunci yang digunakan hanya sebanyak 56 bit saja, karena bit paling signifikan (MSB) dari setiap bit diabaikan. Jadi sebenarnya ukuran kunci DES adalah 56 bit. Adapun Ilustrasi penyandian DES dalam diagram blok dapat dilihat pada gambar di samping. Algoritma penjadwalan sub kunci dibentuk dari pengacakan bit dan pemutaran kiri ruas kanan dan kiri kunci. Pertama kali, bit-bit kunci diacak dengan Permutation Choice 1 dan dibagi dua menjadi ruas kiri dan ruas kanan. Kedua ruas tersebut kemudian diputar kiri dan diacak kembali dengan Permutation Choice 2 untuk menghasilkan sub kunci. Jumlah pemutaran ke kiri ditentukan secara spesifik untuk setiap sub kunci.

Rinciannya adalah sebagai berikut.

[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

Untuk lebih lanjut, mari terlebih dahulu kita bahas detail algoritma penjadwalan sub kunci kemudian algoritma enkripsi dan terakhir algoritma dekripsi. Inti dari semua proses permutasi dalam DES adalah pengacakan bit. Sebagai contoh, jika masukan permutasi sebanyak n bit, maka akan ada sebanyak 2n kemungkinan masukan permutasi dan ada 2n kemungkinan hasil permutasi. Setiap satu kemungkinan masukan akan berpasangan dengan satu kemungkinan keluaran.  
Sebelum proses penjadwalan kunci dimulai, kunci terlebih dahulu dipetakan menjadi matriks 8x8 dan diberi indeks. Dalam setiap byte, indeks paling kecil melambangkan LSB dan indeks paling besar melambangkan MSB. Sebagai contoh, indeks ke 1 melambangkan LSB byte pertama, index ke 8 melambangkan MSB byte pertama, indeks ke 9 melambangkan LSB byte kedua, indeks ke 16 melambangkan MSB byte kedua dan seterusnya hingga indeks ke 64 yang melambangkan MSB byte ke 8. Mari kita perhatikan contoh dibawah ini. Kunci = 0x01, 0x23, 0x45, 0x67, 0x89, 0xab, 0xcd, 0xef

[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

Setelah pemetaan, hasilnya adalah sebagai berikut.

[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

Matriks Indeks Hasil Pemetaan

Pengacakan bit Permutation Choice‐1 akan mengolah 8 byte blok kunci menjadi 56 bit sub kunci yang siap diproses lebih lanjut. Untuk lebih mudahnya, proses pengacakan bit dilambangkan dengan pengacakan indeks bit yang bersangkutan. Berikut ini adalah detail Permutation Choice 1.[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)Setelah kunci diacak dengan Permutation Choice 1, hasil pengacakan bit tersebut kemudian dibagi 2, yakni ruas kiri dan ruas kanan, masing masing berukuran 28 bit (ditandai dengan garis tebal pada hasil Permutation Choice 1). Selanjutnya kedua ruas tersebut kemudian mengalami pemutaran kiri sebanyak jumlah yang tertera pada tabel penjadwalan jumlah pemutaran yang telah kita bahas sebelumnya. Berikut ini adalah ilustrasi pemutaran ke kiri sebanyak 1 kali (untuk ruas kiri atau ruas kanan kunci yang panjangnya 28 bit). Untuk pemutaran ke kiri dengan jumlah yang lebih besar, cukup mengulangi proses ini sebanyak yang diinginkan.

[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

 Setelah ruas kiri dan ruas kanan diputar kiri dengan jumlah tertentu, selanjutnya hasil pemutaran tersebut digabungkan kembali menjadi 56 bit dan diacak dengan Permutation Choice‐2 untuk menghasilkan sub kunci. Rincian Permutation Choice 2 adalah sebagai berikut.

[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

Karena jumlah keluaran Permutation Choice 48 bit sementara masukannya 56 bit, dengan demikian ada 8 bit yang “dihilangkan”. Bit-bit yang tidak muncul dalam keluaran Permutation Choice 2 diwarnai abuabu. Selanjutnya, mari kita perjelas algoritma enkripsinya. Setelah melihat diagram blok secara keseluruhan proses enkripsi,

ada tiga hal yang perlu digarisbawahi dan dibahas lebih lanjut yaitu, pertama IP (Initial Permutation), kedua detail fungsi F dan IP 1

(Inverse Initial Permutation).Selama proses enkripsi, pertama data dipetakan dan diberi indeks dengan prosedur sama persis seperti pemberian indeks pada penjadwalan kunci yang telah didiskusikan sebelumnya. Selanjutnya hasil pemetaan diacak dengan menggunakan Initial Permutation dengan rincian sebagai berikut

Masukan IP[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

Setelah data melewati proses Initial Permutation, data yang akan disandikan kemudian dibagi menjadi dua ruas, yaitu ruas kiri dan ruas kanan yang masIng-masing lebarnya 4 byte (32 bit). Pada setiap putaran, ruas kanan dan sub kunci yang bersangkutan diproses dalam fungsi F dan hasilnya di XOR dengan ruas kiri ruas kanan dan kiri dipertukarkan. Proses ini diulang sebanyak 16 kali.

Pada putaran terakhir ruas kiri dan ruas kanan dipertukarkan kembali untuk menghilangkan efek pertukaran pada putaran terakhir. Hasil ini kemudian diacak kembali dengan menggunakan IP 1 (Inverse Initial Permutation). XOR merupakan fungsi Boolean yang didefinisikan dengan table benaran berikut.

Initial Permutatrion dan Inverse Initial Permutation memiliki karakter saling menetralkan. Dalam notasi matematika, IP−1(IP(A))= A. Inverse Initial Permutation didefinisikan sebagai berikut. [](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

 Fungsi F merupakan fungsi inti kompleks yang terdiri dari beberapa proses. Fungsi F menerima dua parameter, yaitu sub kunci dan ruas kanan data yang akan dienkripsi. Berikut ini adalah diagram blok detail fungsi F. Pada fungsi F, ruas kanan (32 bit) diacak sekaligus diperluas dengan permutasi E menjadi 48 bit. Hasil pengacakan tersebut kemudian di XOR dengan sub kunci yang telah ditetapkan dengan putaran yang bersangkutan. Hasil XOR kemudian dipecah menjadi 8 unit yang masing‐masing lebarnya 6 bit. Setiap unit tersebut kemudian disubstitusikan dalam SBOX S1 hingga S8. 6 bit paling kiri disubstitusikan ke dalam S1 dan 6 bit paling kanan disubstitusikan ke dalam S8. Hasil setiap substitusi kemudian digabungkan menjadi data selebar 48 bit yang kemudian diacak dan diperpendek dengan permutasi P menjadi 32 bit. Hasil permutasi P kemudian dinyatakan sebagai keluaran fungsi F yang nantinya akan di XOR kan dengan ruas kiri data yang akan dienkripsi.

[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

Sekarang mari kita bahas detail fungsi F satu per satu. Pertama, permutasi E memetakan 32 bit masukan menjadi 48 bit keluaran. Karena lebar keluaran lebih besar dari lebar masukan, maka ada beberapa bit masukan yang digandakan untuk mengisi kekosongan. Permutasi E didefinisikan sebagai berikut

[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

Kedua, DES memiliki 8 buah SBOX (S1 hingga S8) yang memiliki masukan selebar 6 bit dan keluaran selebar 4 bit. Karena lebar keluaran SBOX lebih kecil daripada lebar masukannya, maka adakemungkinan beberapa kombinasi masukan yang berbeda akan menghasilkan keluaran yang sama. Seandainya masukan setiap SBOX adalah 1 2 3 4 5 6 x x x x x x maka S1 hingga hingga S8 didefinisikan sebagai berikut.

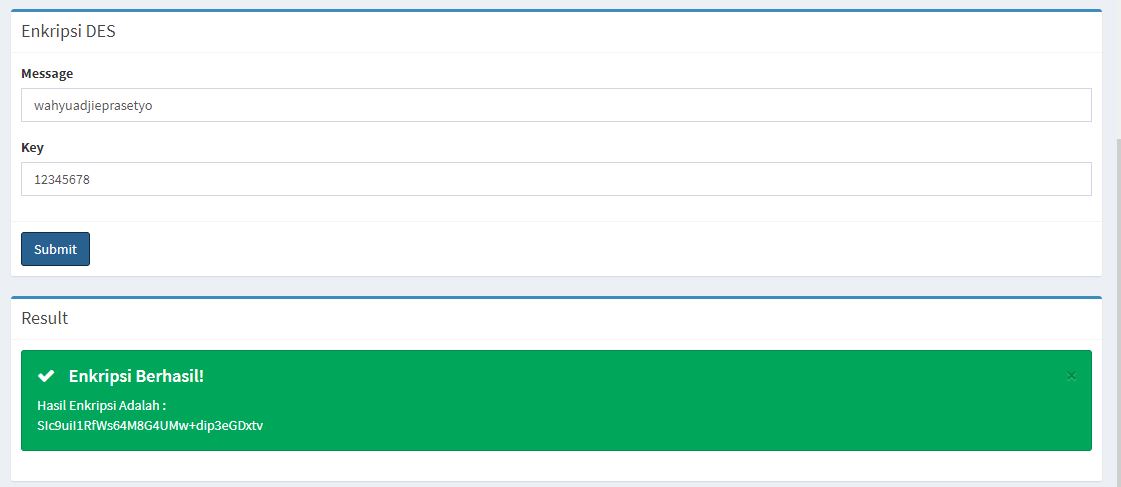
[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

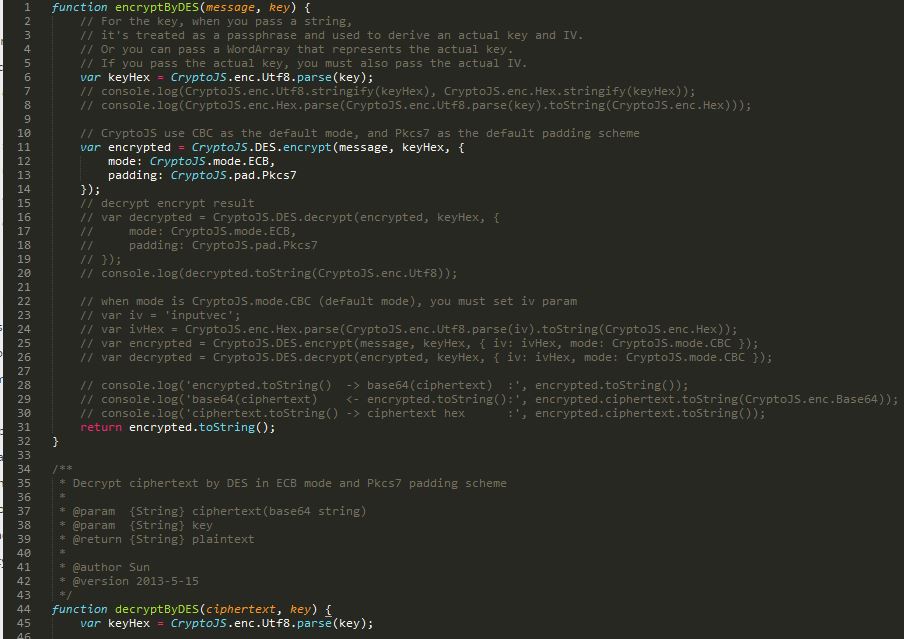
Ketiga, hasil substitusi SBOX kemudian digabungkan menjadi 32 bit dan diacak dengan permutasi P dan hasil permutasi P merupakan keluaran fungsi F yang nantinya di XOR dengan ruas kiri. Permutasi P didefinisikan sebagai berikut.

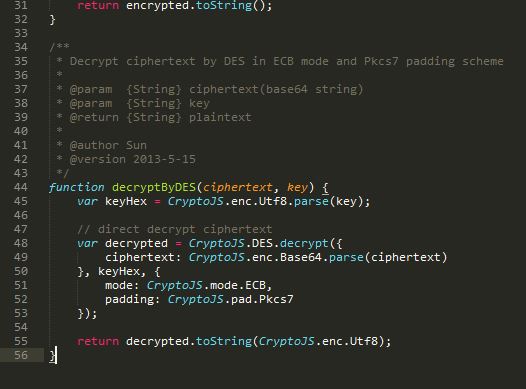
[](https://link.safelinkconverter.com/?id=aHR0cDovL3B1c2F0LW1ha2FsYWhrdS5ibG9nc3BvdC5jb20v&c=0&user=61204)

Dalam DES, algoritma yang digunakan dalam proses enkripsi sama persis dengan algoritma yang digunakan dalam proses dekripsi, hanya saja penggunaan sub kuncinya saja yang berbeda. Dalam proses dekripsi, urutan sub kunci yang digunakan merupakan kebalikan urutan sub kunci yang digunakan dalam proses enkripsi.

**Screenshoot :**

**Source kode :**

****

****

**Kekurangan DES**

Data Encryption Standard ( DES ) pun seperti sistem yang lainnya, memiliki kekurangan dan kelebihan. Salah satu kekurangan DES adalah proses yang lebih lama dalam melakukan proses dekripsi dan enkripsi.

**Kelebihan DES**

Kelebihan DES yaitu, sistem sandi lebih kompleks sehingga membutuhkan waktu yang tidak cepat untuk menembus enkripsi DES dan sulit untuk diketahui dari pihak luar. Tetapi setelah berkembangannya jaman DES tidak digunakan karena ukuran kunci yang terlalu kecil, sehingga mudah untuk ditembus

1. **MD5**

MD5 adalah salah satu dari serangkaian algortima(merupakan salah satu fungsi Hash) message digest yang didesain oleh Profesor Ronald Rivest dari MIT (Rivest, 1994). Saat kerja analitik menunjukkan bahwa pendahulu MD5, yaitu MD4 mulai tidak aman, MD5 kemudian didesain pada tahun 1991 sebagai pengganti dari MD4 (kelemahan MD4 ditemukan oleh Hans Dobbertin).pada tahun 1993, den Boer dan Bosselaers memberikan awal, bahkan terbatas, hasil dari penemuan pseudo-collision dari fungsi kompresi MD5. Dua vektor inisialisasi berbeda I dan J dengan beda 4-bit diantara keduanya.

**Fungsi Hash**

Hash adalah suatu teknik "klasik" dalam Ilmu Komputer yang banyak digunakan dalam praktek secara mendalam. Hash merupakan suatu metode yang secara langsung mengakses record-record dalam suatu tabel dengan melakukan transformasi aritmatik pada key yang menjadi alamat dalam tabel tersebut. Key merupakan suatu input dari pemakai di mana pada umumnya berupa nilai atau string karakter.

Pelacakan dengan menggunakan Hash terdiri dari dua langkah utama, yaitu:

1. Menghitung Fungsi Hash. Fungsi Hash adalah suatu fungsi yang mengubah key menjadi alamat dalam tabel. Fungsi Hash memetakan sebuah key ke suatu alamat dalam tabel. Idealnya, key-key yang berbeda seharusnya dipetakan ke alamat-alamat yang berbeda juga. Pada kenyataannya, tidak ada fungsi Hash yang sempurna. Kemungkinan besar yang terjadi adalah dua atau lebih key yang berbeda dipetakan ke alamat yang sama dalam tabel. Peristiwa ini disebut dengan collision (tabrakan). Karena itulah diperlukan langkah berikutnya, yaitu collision resolution (pemecahan tabrakan).
2. Collision Resolution. Collision resolution merupakan proses untuk menangani kejadian dua atau lebih key di-hash ke alamat yang sama. Cara yang dilakukan jika terjadi collision adalah mencari lokasi yang kosong dalam tabel Hash secara terurut. Cara lainnya adalah dengan menggunakan fungsi Hash yang lain untuk mencari lokasi kosong tersebut.

**Cara kerja MD5**

Algoritma MD5 adalah algoritma yang menggunakan fungsi hash satu arah yang diciptakan oleh Ron Rivest. Algoritma merupakan pengembangan dari algoritma-algoritma sebelumnya yaitu algoritma MD2 dan algoritma MD4 karena kedua algoritma ini berhasil diserang para *cryptanalist*.

Cara kerja kriptografi algoritma MD5 adalah menerima input berupa pesan dengan ukuran sembarang dan menghasilkan message diggest yang memiliki panjang 128 bit. Berikut ilustrasi gambar dari pembuatan message diggest pada kriptografi algoritma MD5 :



Gambar 3.1 Pembuatan message digest dengan algoritma MD5

Menilik dari gambar diatas, secara garis besar pembuatan message digest ditempuh melalui empat langkah, yaitu :

1. Penambahan bit bit pengganjal

Proses pertama yang dilakukan adalah menambahkan pesan dengan sejumlah bit pengganjal sedemikian sehingga panjang pesan (dalam satuan bit) kongruen dengan 448 modulo 512. Ini berarti setelah menambahkan bit-bit pengganjal, kini panjang pesan adalah 64 bit kurang dari kelipatan 512. Hal yang perlu diingat adalah angka 512 muncul karena algoritma MD5 memproses pesan dalam blok-blok yang berukuran 512.

Apabila terdapat pesan dengan panjang 448 bit, maka pesan tersebut akan tetap ditambahkan dengan bit-bit pengganjal. Pesan akan ditambahkan dengan 512 bit menjadi 96 bit. Jadi panjang bit-bit pengganjal adalah antara 1 sampai 512. Lalu satu hal lagi yang perlu diperhatikan adalah bahwasanya bit-bit pengganjal terdiri dari sebuah bit 1 diikuti dengan sisanya bit 0.

1. Penambahan nilai panjang pesan semula

kemudian proses berikutnya adalah pesan ditambah lagi dengan 64 bit yang menyatakan panjang pesan semula. Apabila panjang pesan lebih besar dari 264 maka yang diambil adalah panjangnya dalam modulo 264. dengan kata lain, jika pada awalnya panjang pesan sama dengan K bit, maka 64 bit yang ditambahkan menyatakan K modulo 264. sehingga setelah proses kedua ini selesai dilakukan maka panjang pesan sekarang adalah 512 bit.

1. Inisialisasi penyangga MD5

Pada algoritma MD5 dibutuhkan empat buah penyangga atau buffer, secara berurut keempat nama penyangga diberi nama A, B, C dan D. Masing-masing penyangga memiliki panjang 32 bit. Sehingga panjang total :



Keempat penyangga ini menampung hasil antara dan hasil akhir. Setiap penyangga diinisialisasi dengan nilai-nilai (dalam notasi Hexadesimal) sebagai berikut :

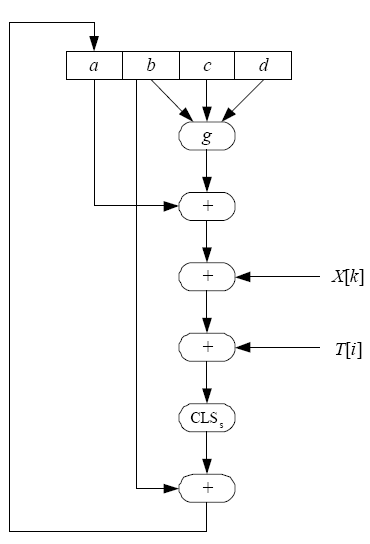


1. Pengolahan pesan dalam blok berukuran 512 bit

Proses berikutnya adalah pesan dibagi menjadi L buah blok yang masing-masing panjangnya 512 bit (Y0 sampai YL-1). Setelah itu setiap blok 512 bit diproses bersama dengan penyangga MD yang menghasilkan keluaran 128 bit, dan ini disebut HMD5. Berikut ini gambaran dari proses HMD5 :

Gambar 3.2 Pengolahan blok 512 bit (Proses HMD5)

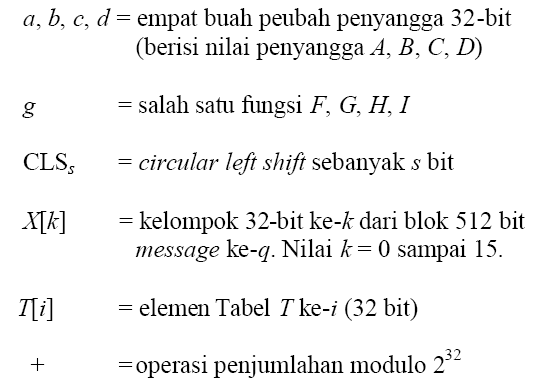
Dari gambar diatas dapat kita lihat bahwa proses HMD5 terdiri dari 4 buah putaran, dan masing-masing putaran melakukan opersi dasar MD5 sebanyak 16 kali. Dimana disetiap operasi dasar memakai sebuah elemen T. Sehingga setiap putaran memakai 16 elemen tabel T.

Pada gambar 3.2, Yq menyatakan blok 512 bit ke-q dari pesan yang telah ditambahkan dengan bit-bit pengganjal pada proses pertama dan tambahan 64 bit nilai panjang pesan semula pada proses kedua. MDq adalah nilai *message digest* 128 bit dari proses HMD5 ke-q. Pada awal proses , MDq berisi nilai inisialisasi penyangga MD. Kemudian fungsi fF, fG, fH, dan fI pada gambar, masing-masing berisi 16 kali operasi dasar terhadap input, setiap operasi dasar menggunakn elemen tabel T. Berikut ini ilustrasi gambar operasi dasar MD5 :

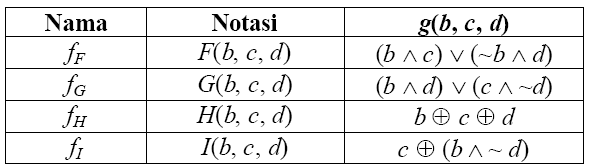
Gambar 3.3 Operasi dasar MD5

Operasi dasar MD5 yang diperlihatkan gambar diatas dapat dituliskan dengan persamaan berikut ini :

https://lh3.googleusercontent.com/yc1r6MUqLhvl51nGPUE-Rv5PUhO24HhD5F0jrYruf-7VTGyBPSu-ym-Qy8ogAS3LXakZf-AMHZ_rPlVJVf-lYQ4YnwGb6xeMpHmfJi_grcO8yi_Hknjyse1ArY7CFa4EDHaRF5Pxoeeup1mplg

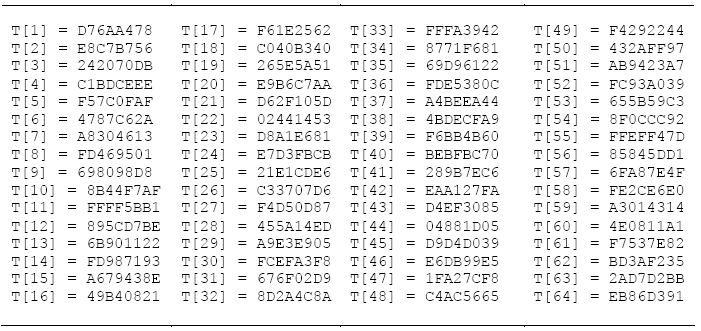
Dimana :

Fungsi fF, fG, fH, dan fI adalah fungsi untuk memanipulasi masukan a, b, c, dan d dengan ukuran 32-bit. Masing-masing fungsi dapat dilihat pada Tabel 1 dibawah ini :

Tabel 1 Fungsi-fungsi dasar MD5

NB: secara berurut peartor logika AND, OR, NOT dan XOR dilambangkan dengan ∧, ∨, ~, ⊕

Kemudian nilai T[i] dapat dilihat pada tabel dibawah ini. Tabel ini disusun oleh fungsi 232 x abs(sin(i)), i dalam radian.

Tabel 2. Nilai T[i]

Sebagaimana telah dijelaskan sebelumnya bahwa fungsi fF, fG, fH, dan fI melakukan 16 kali operasi dasar. Misalkan notasi berikut ini :

https://lh4.googleusercontent.com/KNYqQeuE7ee-uetp4ZzmTtRqVgftNTzZy211ck5GzAYW_1_4tMhQMI9wG3KR-igME2CAhzRDgnGkJACrAtM_w67L7xP8X40BzEdxfqcaGi4BiXtjXuQet5-ACWk8j80Q-7_FiJwRQmN_K8bmMg

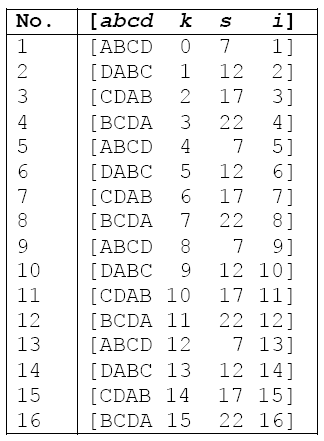
Meyatakan operasi

https://lh3.googleusercontent.com/3iFGid5GP_vunvEXdR1lGEOdFFKzIBDOUBO9X9Oi11mPpYIsFjjCwRqpfpBEpgLfYa-UIw6DqEe7YNzP7zQX1DER8SqeY6-Bk0yyuSynQ9_JmvaVvcDMFD01tvB1KfbefsmB4GqhcHsZdGHi8A

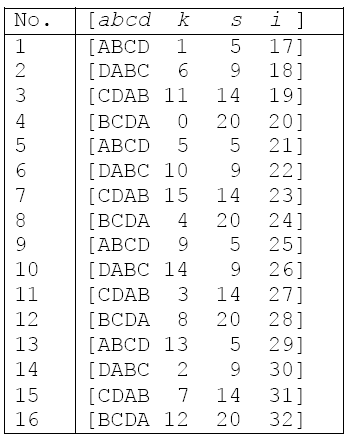
untuk operasi diatas, <<<s melambangkan opersi *circular left shift* 32 bit, maka operasi dasar pada masing-masing putaran dapat ditabulasikan sebagai berikut :

* putaran 1 : 16 kali operasi dasar dengan g(b, c, d) – F(b, c, d), dapat dilihat pada tabel berikut ini :

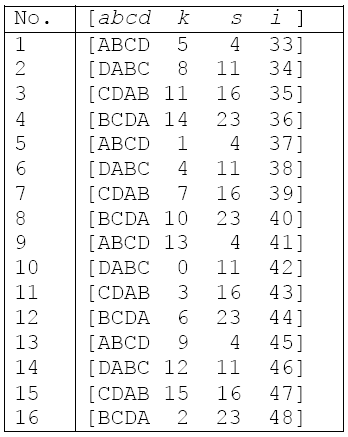
Tabel 3 Rincian operasi pada fungsi F(b, c, d)



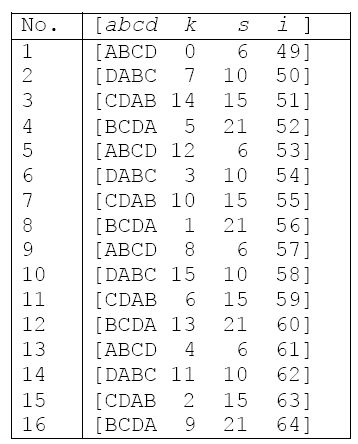
* putaran 2 : 16 kali operasi dasar dengan g(b, c, d) – G(b, c, d), dapat dilihat pada tabel berikut ini :

Tabel 4 Rincian operasi pada fungsi G(b, c, d)

* putaran 3 : 16 kali operasi dasar dengan g(b, c, d) – H(b, c, d), dapat dilihat pada tabel berikut ini

Tabel 5 Rincian operasi pada fungsi H(b, c, d)

* putaran 4 : 16 kali operasi dasar dengan g(b, c, d) – I(b, c, d), dapat dilihat pada tabel berikut ini :

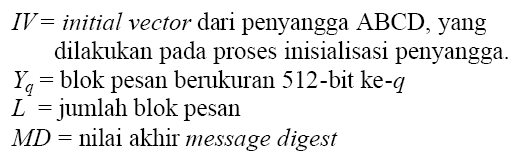
Tabel 6 Rincian operasi pada fungsi I(b, c, d)

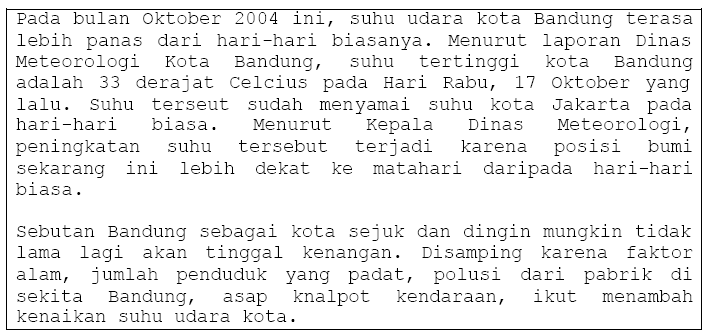
Setelah putaran keempat, a, b, c dan d di tambahkan ke A, B, C dan D yang selanjutnya algoritma akan memproses untuk blok data berikutnya (Yq+1). Output akhir dari algoritma MD5 adalah hasil penyambungan bit-bit di A, B, C dan D.

https://lh4.googleusercontent.com/TP3lYN8Ih19TPMiWo1LpuM_Y9NLRvsotnRzJKeryDgTUdhSQtGt6ZiKhK5KGCU5iaI3DkkRbsCpuOD3gUBtEiRHfqGajQctj8pySqPmmzdt6JZLF70Z_6_EvNeNnNfwwF9TcnsUPAUaDIN2fwQDari uraian diatas, secara umum fungsi hash MD5 dapat ditulis dalam persamaan matematis sebagai berikut :

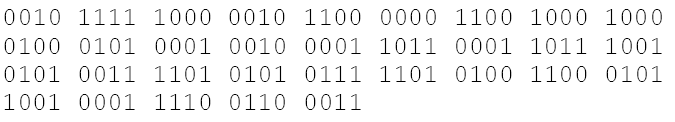
https://lh6.googleusercontent.com/fE3htJqzjmPU1ReBuwuV1-YmDqd378MA37gabrJWfSvtpbRsEDsdnjtF4q_LHGJxMQWUXfvrTVh-OyZD5-PSQHit2etBAhr1Ka9jyq0K7y0PY7I2xjErwA4bk91IwstIuf1LsKbpeL344VTvGg

https://lh6.googleusercontent.com/K7bVR59pKvc-YIIUU0oKrxmw0aF6KLlJiRVhDGuP_PA3NFYYgDo7EwZGj0gSEYhzuWx1m0-tRPOyubtNmuta7_1opt7KAqsrloHYBcWIdOdjBYUCg2ZwlaF_YkTz5mrmMCG4iBM5yYwsXJo3HQ

Dimana

Agar lebih mudah dipahami, berikut ini contoh penerapan kriptografi MD5 pada suatu pesan yang ingin dirahasiakan. Misalkan terdapat sebuah arsip dengan nama bandung.txt sebagai berikut :

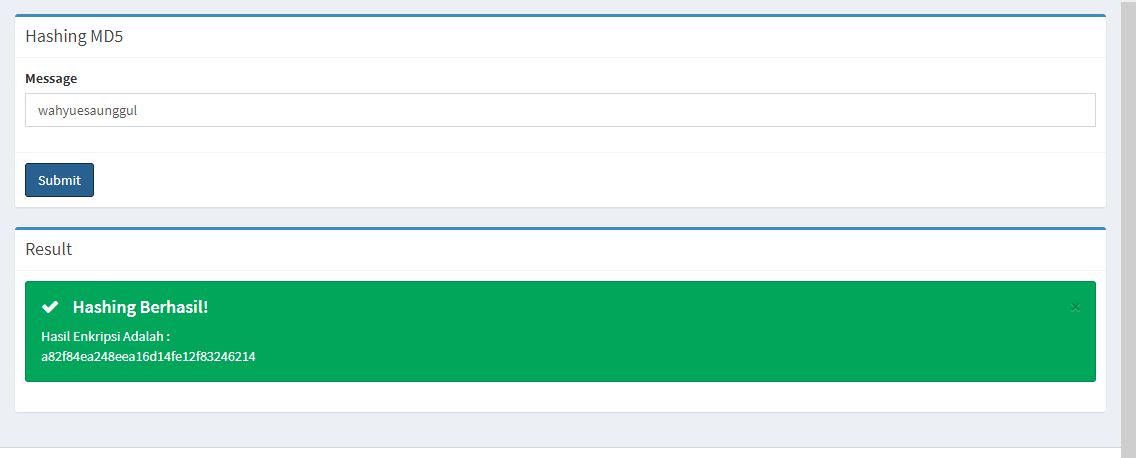
Message digest yang dihasilkan dari arsip diatas adalah untuk 128 bit :

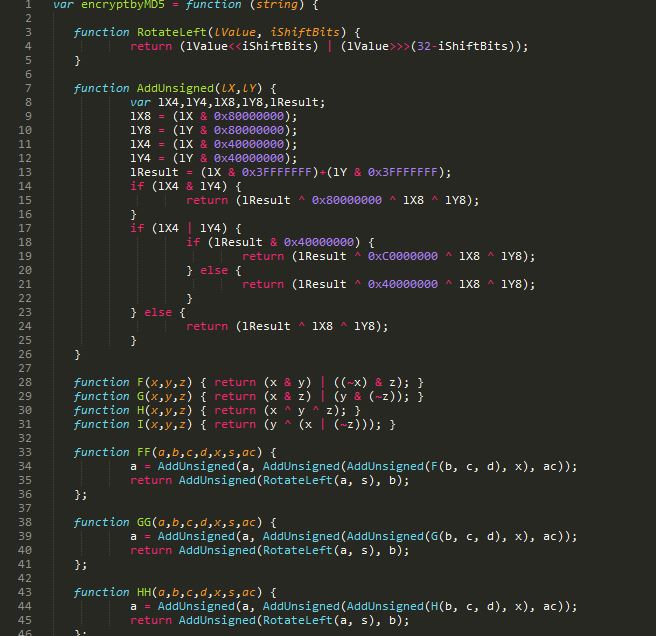


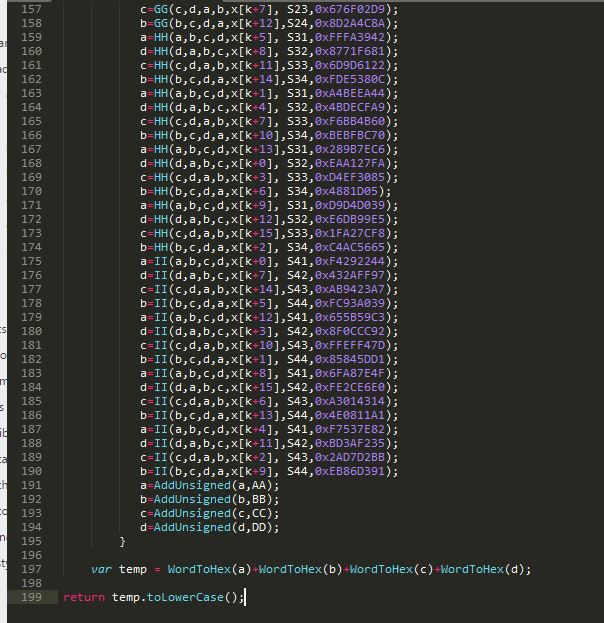
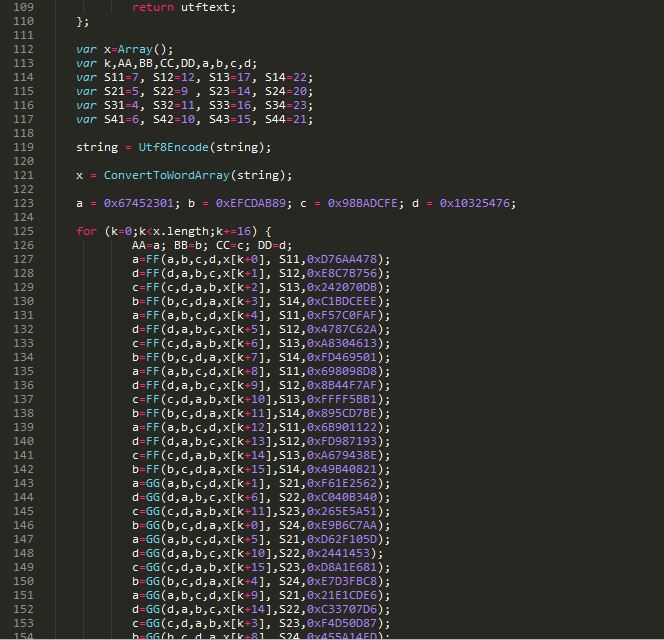
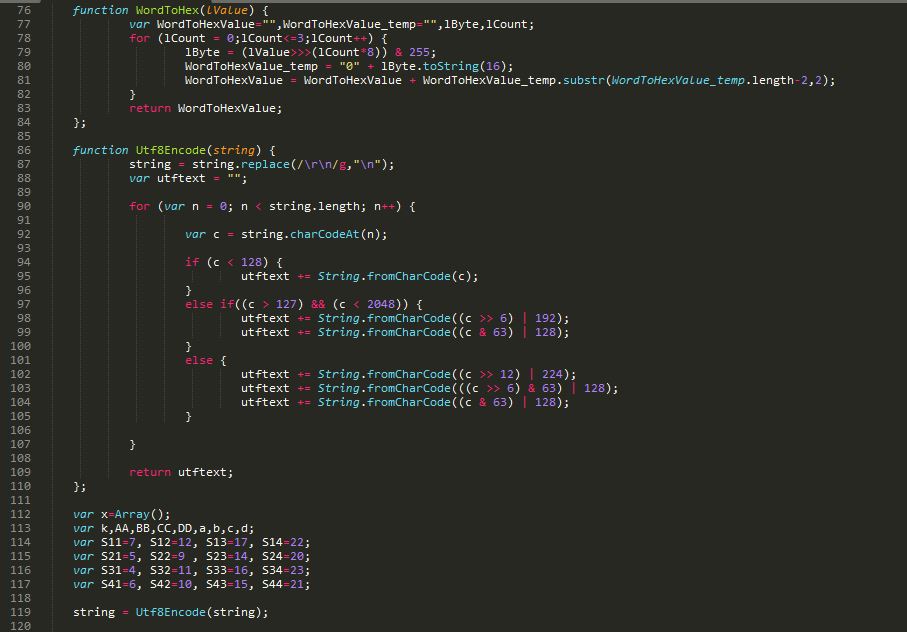
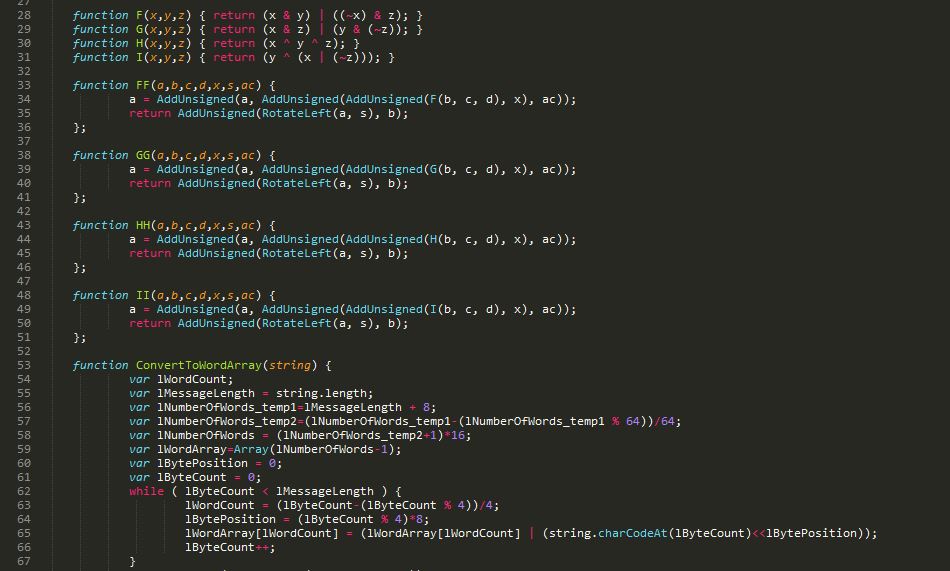
Lalu untuk notasi hexadesimal :

https://lh6.googleusercontent.com/3fEIyvxnVCkaeGo5n1x7sWPueH3dJkoI6ZvcBwu6TE1dRB2Wjdy-ow_WSVIwOUSQIycmeHlYf5TiwlAn37CDHP4n2Lumw9ufwNL-QUXPRDAEFQEksRFHkEBPBHqtbxanXVzeRyS_4r6LbGdSuA

**Screenshoot :**



Source kode :



**Kekurangan MD5**

Kelemahan MD5 ada pada design sehingga lebih mudah dilakukan kriptanalisis dibandingkan SHA-1

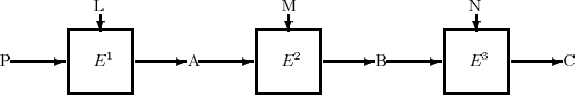
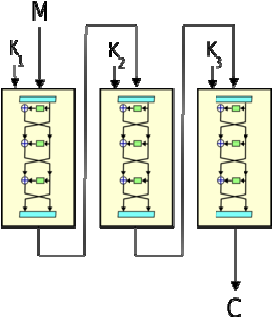
**Kelebihan MD5**

Algoritma MD5 dapat digunakan sebagai aplikasi digital signature pasda kartu Tanda penduduk (KTP) sebagai otentifikasi . Pada KTP nantinya akan ditambahkan digtal signaturenya, dimana data yang akan diambil adalah Nomor induk kependudukan . Karena setiap penduduknya nanti akan memberikan sebuh kunci kepda penduduk yakni kunci private. Dimana kunci private ini juga akan di berikan kepada instant terkait.

1. **Triple DES**

Triple DES dikenal juga dengan nama TDES, atau lebih standar lagi adalah TDEA ( Triple Data Encryption Algorithm). TDES mengaplikasikan DES sebanyak 3 kali, dengan 3 buah kunci.

Berikut adalah skema global dari TDES:

1. Analisis Keamanan

**Cara kerja Triple DES**

Triple DES merupakan varian pengembangan dari DES (Data Encryption Standar) – sebelumnya disebut sebagai “multiple DES” dikarenakan pada dasarnya triple DES hanyalah penggunaan DES secara berulang; dalam hal ini pengulangannya dilakukan tiga kali. Triple DES umumnya disebut jugadengan singkatan TDES atau dengan istilah 3DES.

Secara umum triple DES dirumuskan sebagai berikut:

Enkripsi : (C) O = EK3(DK2(EK1(P)))

Dekripsi : (P)O = DK3(EK2(DK1(C)))

Keterangan:

P = plaintext

C = ciphertext

E = enkripsi

D = dekripsi

Ki = kunci ke-i

Varian di atas umumnya disebut dengan mode EEE (dikarenakan menggunakan tiga kali proses enkripsi). Namun, kemudian dilakukan sebuah penyederhanaan terhadap varian tersebut sehingga melahirkan mode baru yang disebut sebagai EDE (enkripsidekripsi- enkripsi), dengan adanya penyisipan fungsi dekripsi. Penggunaan tiga kali DES pada triple DES diharapkan dapat meningkatkan keamanan dikarenakan juga adanya penggunaan kunci yang lebih panjang yaitu kunci dengan ukuran 168 bit (tiga kali ukuran DES, 56 bit). Pada penggunaan triple DES dengan mode EDE dapat dilakukan dengan menggunakan 3 kunci, 2 kunci ataupun hanya menggunakan 1 kunci. Berikut blok diagram triple DES :

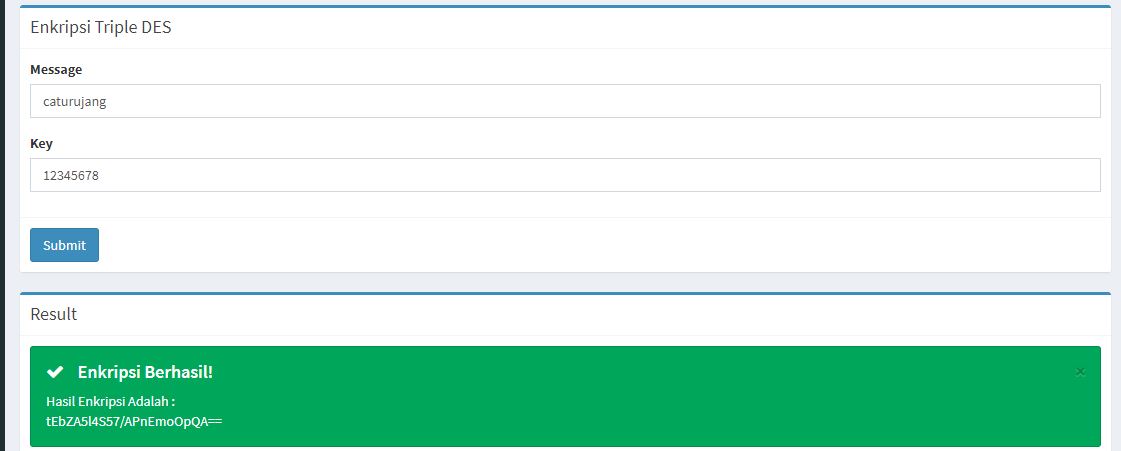
Penggunaan triple DES dengan 1 kunci merupakan bentuk penyederhanaan yang menggunakan kunci k = k1 = k2 = k3.

Menyangkut keamanan dalam penggunaan triple DES, masih memungkinkan terjadi penyerangan dengan menggunakan sekitar 232 known-plaintexts, 2113 langkah, 290 pemecahan DES, dan 288 kapasitas memori. Tentu saja untuk melakukan hal tersebut akan membutuhkan biaya yang sangat besar.

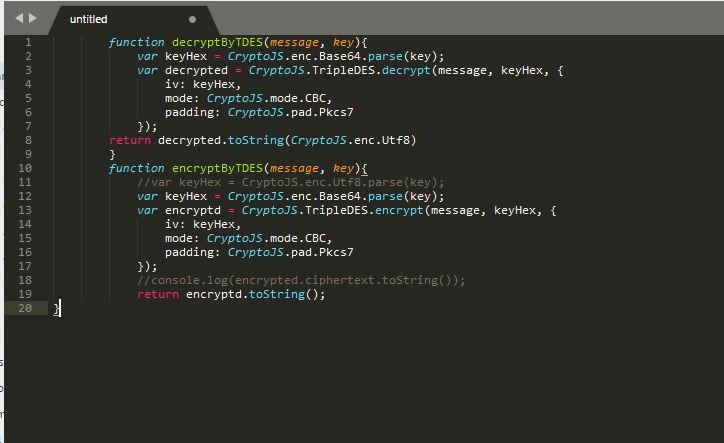
**Fungsi triple DES**

Pada awalnya, ukuran kunci sandi DES yaitu 56 bit sudah mencukupi pada saat algoritme ini dibuat. Namun, dengan meningkatnya kemampuan komputasi, serangan brutal telah mungkin terjadi. Triple DES menyediakan metode yang sederhana dengan menambah ukuran kunci DES untuk mencegah serangan tersebut, tanpa memerlukan perancangan sandi blok (*block cipher*) yang sama sekali baru.

**Screenshot :**

****

**Source kode:**

****

**Algoritma RSA**

Algoritma RSA merupakan salah satu algoritma public key yang populer dipakai dan bahkan masih dipakai hingga saat ini. Kekuatan algoritma ini terletak pada proses eksponensial, dan pemfaktoran bilangan menjadi 2 bilangan prima yang hingga kini perlu waktu yang lama untuk melakukan pemfaktorannya. Algoritma ini dinamakan sesuai dengan nama penemunya, Ron Rivest, Adi Shamir dan Adleman(Rivest-Shamir-Adleman) yang dipublikasikan pada tahun 1977 di MIT, menjawab tantangan yang diberikan algoritma pertukaran kunci Diffie Hellman.

**Cara kerja**

Skema RSA sendiri mengadopsi dari skema block cipher, dimana sebelum dilakukan enkripsi, plainteks yang ada dibagi – bagi menjadi blok – blok dengan panjang yang sama, dimana plainteks dan cipherteksnya berupa integer(bilangan bulat) antara 1 hingga n, dimana n berukuran biasanya sebesar 1024 bit, dan panjang bloknya sendiri berukuran lebih kecil atau sama dengan log(n) +1 dengan basis 2. Fungsi enkripsi dan dekripsinya dijabarkan dalam fungsi berikut :

C = Me mod n ( fungsi enkripsi )

M = Cd mod n (fungsi dekripsi)

C = Cipherteks

M = Message / Plainteks

e = kunci publik

d= kunci privat

n = modulo pembagi(akan dijelaskan lebih lanjut )

Kedua pihak harus mengetahui nilai e dan nilai n ini, dan salah satu pihak harus memilki d untuk melakukan dekripsi terhadap hasil enkripsi dengan menggunakan public key e. Penggunaan algoritma ini harus memenuhi kriteria berikut :

1. Memungkinkan untuk mencari nilai e, d, n sedemikian rupa sehingga Med mod n = M untuk semua M < n.

2. Relatif mudah untuk menghitung nilai Me mod n dan Cd mod n untuk semua nilai M < n.

3. Tidak memungkinkan mencari nilai d jika diberikan nilai n dan e.

Syarat nilai e dan d ini, gcd(d,e)=1

sebelum memulai penggunaan RSA ini, terlebih dahulu kita harus memiliki bahan – bahan dasar sebagai berikut :

1. p, q = 2 bilangan prima yang dirahasiakan

2. n, dari hasil p.q

3. e, dengan ketentuan gcd (Φ(n), e) =1

4. d, e-1 (mod Φ(n))

Saya akan berikan satu contoh :

1. Pilih 2 bilangan prima, misalnya p = 17 dan q = 11.

2. Hitung n = pq = 17 × 11 = 187.

3. Hitung Φ(n) = (p – 1)(q – 1) = 16 × 10 = 160.

4. Pilih nilai e sedemikian sehingga relatif prima terhadap Φ(n) = 160 dan kurang dari Φ(n); kita pilih e = 7.

5. Hitung d sedemikian sehingga de ≡ 1 (mod 160) dan d < 160.Nilai yang didapatkan d = 23,karena 23 × 7 = 161 = (1 × 160) + 1; d dapat dihitung dengan Extended Euclidean Algorithm.

Nah, nilai e dan d inilah yang kita sebut sebagai Public Key(e) dan Private Key(d). Pasangan Kunci Publiknya ={7,187} dan Kunci Privatnya = {23, 187}

Sekarang kita aplikasikan dalam proses enkripsi.

Misalnya kita punya M 88. Untuk proses enkripsi, kita akan menghitung C = 887 mod 187.= 887 mod 187.

=894,432 mod 187

=11

Nah, kita mendapatkan nilai C =11.

Selanjutnya, nilai C ini dikirimkan kepada penerima untuk didekripsi dengan kunci privat miliknya.

M = Cd mod n

= 1123 mod 187

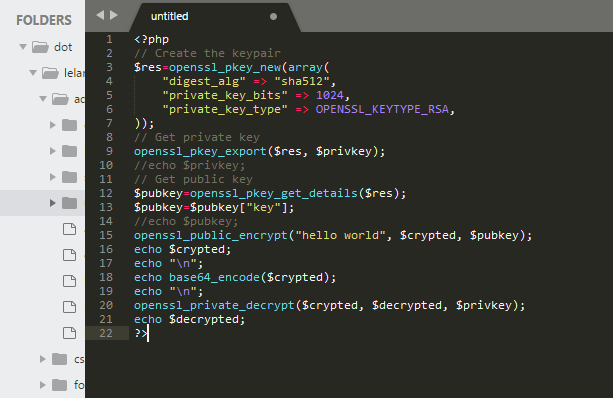
=79,720,245 mod187

= 88

**Screenshoot :**



**Source kode :**

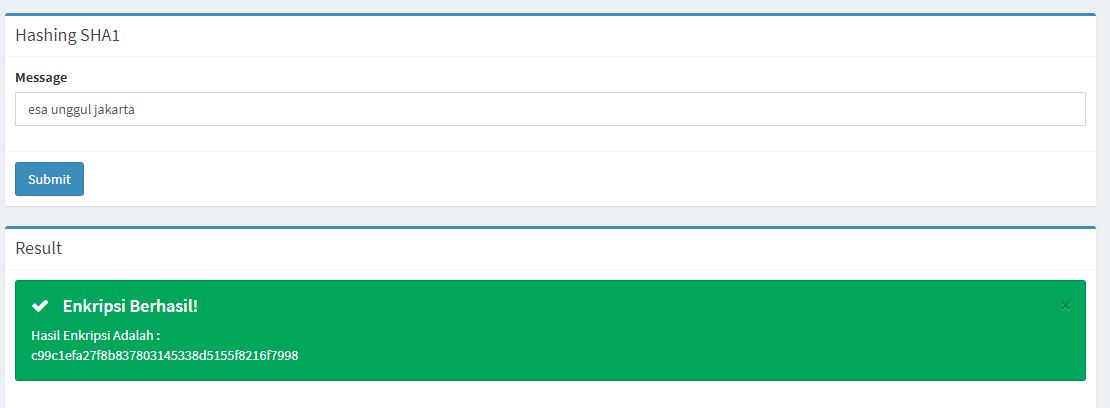


1. **SHA -1**
2. SHA adalah serangkaian fungsi cryptographic hash yang dirancangoleh National Security Agency (NSA) dan diterbitkan oleh NIST sebagai US Federal Information Processing Standard.
3. SHA adalah Secure Hash Algoritma. Jenis-jenis SHA yaitu SHA-0, SHA-1, dan SHA-2
4. Untuk SHA-2 menggunakan algoritma yang identik dengan ringkasan ukuran variabel yang terkenal sebagai SHA-224, SHA-256, SHA-384, dan SHA-512.

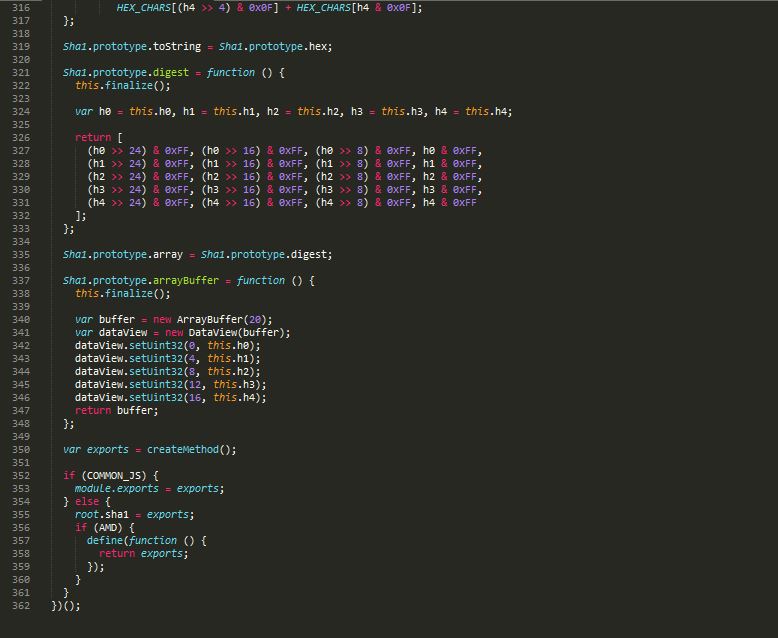
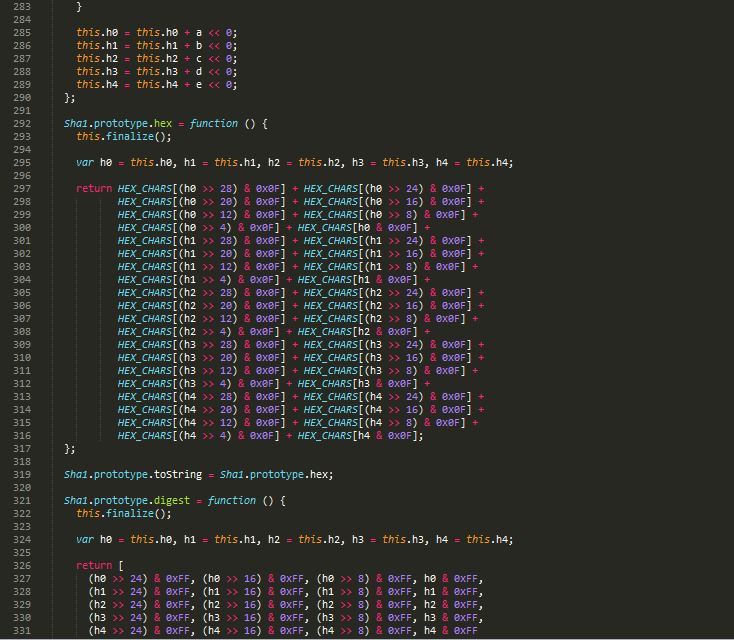
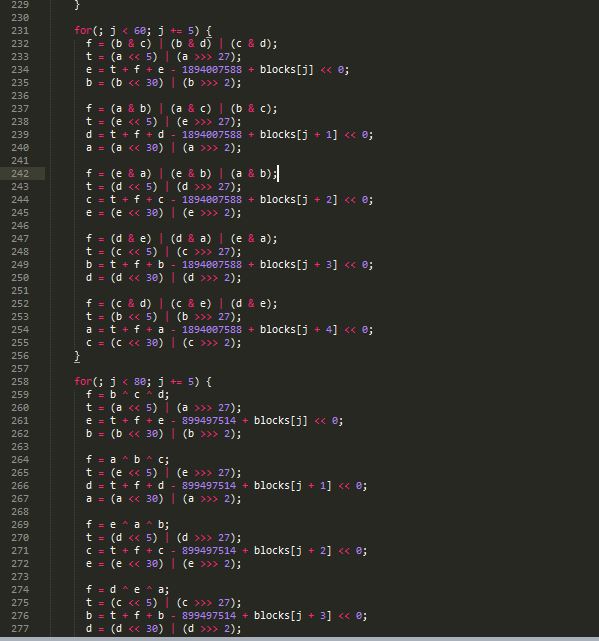
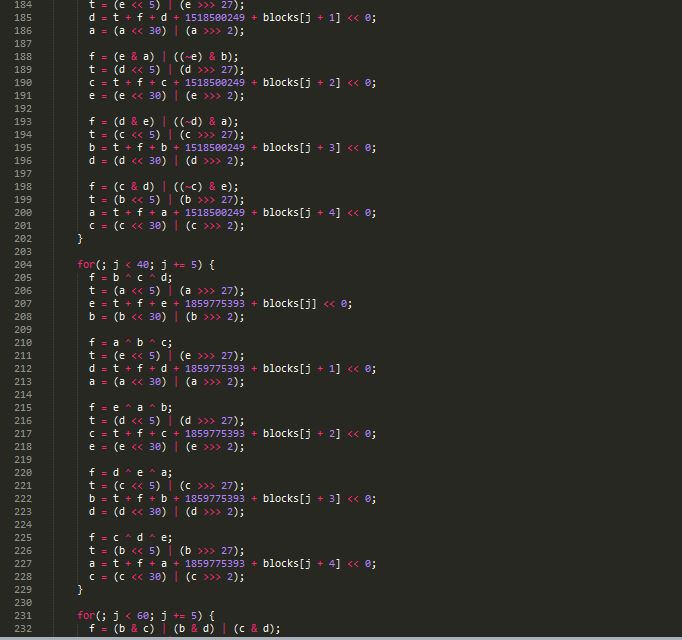
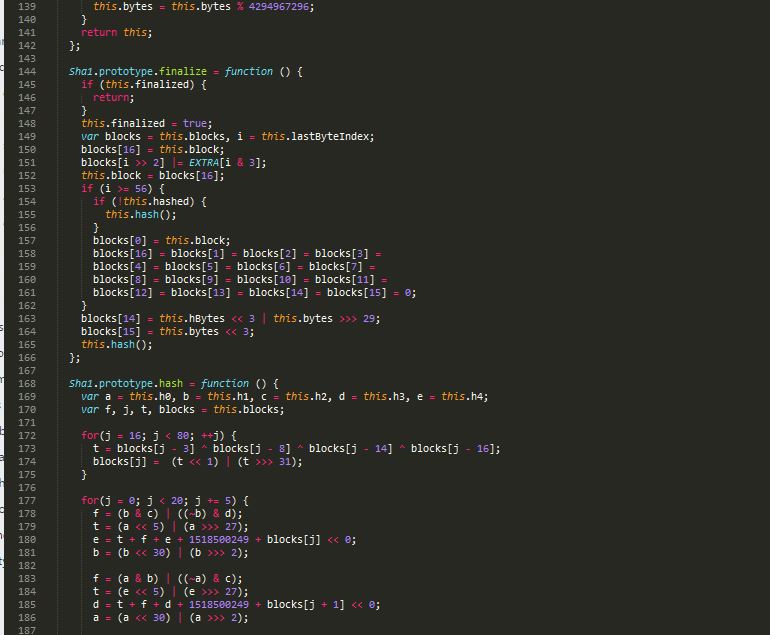
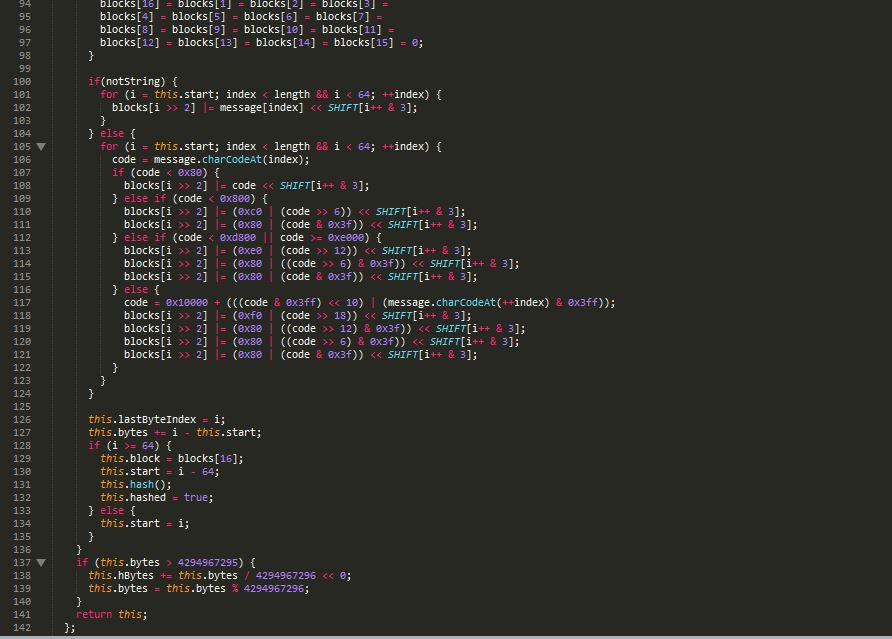
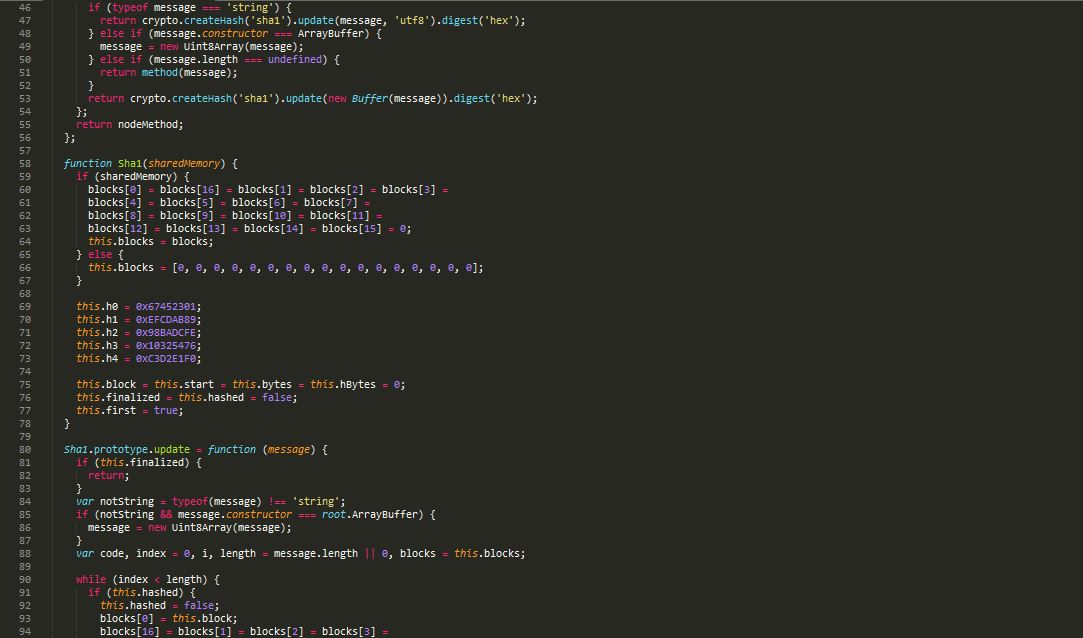
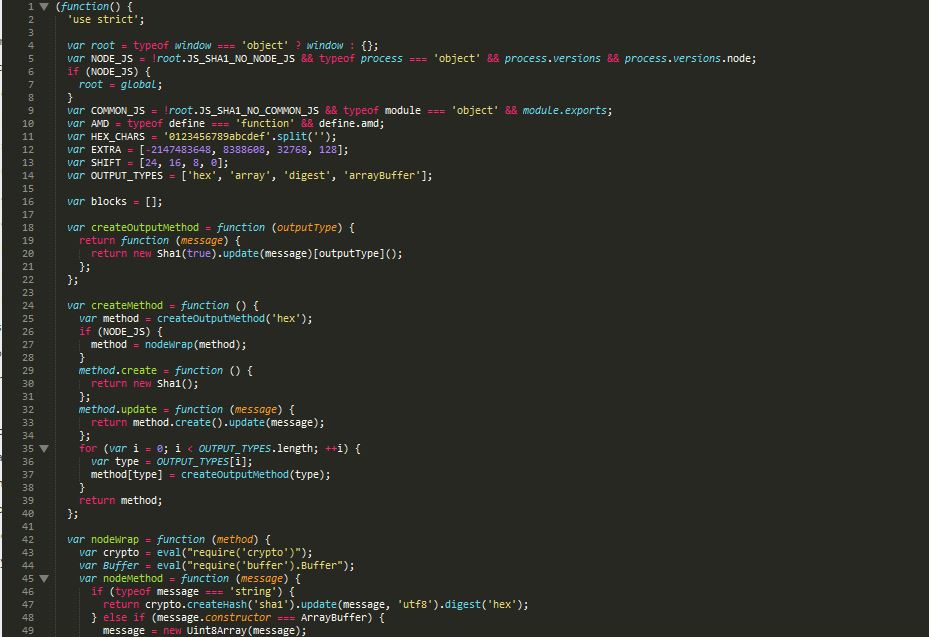
CARA KERJA SHA - 1

Pesan diberi tambahan untuk membuat panjangnya menjadi kelipatan 512 bit ( l x 512 ). Jumlah bit asal adalah k bit. Tambahkan bit secukupnya sampai 64 bit kurangnya dari kelipatan 512 ( 512 – 64 = 448 ), yang disebut juga kongruen dengan 448 ( mod 512 ). Kemudian tambahkan 64 bit yang menyatakan panjang pesan. Inisiasi 5 md variable dengan panjang 32 bit yaitu a,b,c,d,e. Pesan dibagi menjadi blok-blok berukuran 512 bit dan setiap blok diolah. Kemudian keluaran setiap blok digabungkan dengan keluaran blok berikutnya, sehingga diperoleh output ( diggest ). Fungsi kompresi yang digunakan oleh algoritma sha-1 adalah sebagai berikut : A,b,c,d,e ← ( e + f (t,b,c,d) + s5 (a) + wt + kt),a,s30(b),c,d.]

**Screenshoot :**



Source kode :

****

**Kelebihan SH-1**

Keamanan terhadapserangan brute-force. Hal yang paling penting adalah bahwa SHA-1 menghasilkan diggest 32-bit lebih panjang dari MD5. Dengan brute-force maka SHA-1 lebihkuat dibanding MD5

**Kelemahan SH-1**

Kecepatan. Kedua algoritma bekerja pada modulo 232 sehingga keduanya bekerja baik pada arsitektur 32 bit. SHA-1 mempunyai langkah lebih banyak dibandingkan MD5 ( 80 dibanding MD5 64 ) dan harus memproses 160 bit buffer dibanding DM5 128 bit buffer, sehingga SHA-1 bekerja lebih lambat dibanding MD5 pada perangkat keras yang sama.

**DSA (Digital Signature Algorithm)**

DSA (Digital Signature Algorithm) menggunakan fungsi hash SHA (Secure Hash Algorithm) untuk mengubah pesan menjadi intisari pesan yang berukuran 160 bit. DSA dan algoritma tanda-tangan digital lainnya mempunyai tiga proses utama yaitu:

1.      Pembangkitan pasangan kunci (Key Pair Generation)

2.      Pembangkitan tanda-tangan digital (Digital Signature Generation)

3.      Verifikasi tanda-tangan digital (Digital Signature Verification)

**Parameter DSA**

DSA dikembangkan dari algoritma ElGamal. DSA mempunyai properti berupa parameter sebagai berikut.

1.      p, adalah bilangan prima dengan panjang L bit,

dimana 2L-1 < p < 2L dengan 512 ≤ L ≤ 1024 dan L adalah kelipatan 64.  Parameter p bersifat publik dan dapat digunakan bersama oleh orang di dalam kelompok.

2.      q adalah bilangan prima 160 bit, faktor dari p- 1 dengan kata lain (p-1) mod q = 0. Parameter q bersifat publik dan dapat digunakan bersama-sama oleh orang di dalam kelompok dimana 2159 < q < 2160 .

3.      Pembangkit g pada grup siklik tunggal dengan order q di dalam Ζ𝑝 ∗. Hitung g =h ( p – 1 )/q mod p, dengan h ∈ Ζ𝑝 ∗, 1< h < p – 1 sehingga g > 1. Parameter g bersifat publik dan dapat digunakan bersama-sama oleh orang di dalam kelompok.

4.      x bilangan bulat dimana 0 < x < q dengan panjang 160 bit. Parameter x bersifat privat yang hanya boleh diketahui oleh pengirim pesan.

5.      y = gx mod p adalah kunci publik.

6.      m, pesan yang akan diberi tanda-tangan.

**Prosedur Pembangkitan Sepasang Kunci**

1.  Pilih bilangan prima p dan q, yang dalam hal ini *p*1mod *q*0

2.  Hitung g = h(p – 1)/q mod p, yang dalam hal ini 1*h**p*1dan *h*( *p*1) /  *q*mod *p*1

3.  Tentukan kunci privat x, yang dalam hal ini x< q.

4.  Hitung kunci publik y = gx mod p.

Jadi, prosedur di atas menghasilkan:

kunci publik dinyatakan sebagai (p, q, g, y);

kunci privat dinyatakan sebagai (p, q, g, x).

**Prosedur Pembangkitan Tanda-tangan (Signing)**

1.  Ubah pesan m menjadi intisari pesan dengan fungsi *hash*SHA, H.

2.  Tentukan bilangan acak k < q.

3. Tanda-tangan dari pesan m adalah bilangan r dan s. Hitung r dan s sebagai berikut:

*r*(*g k*mod *p*)mod *q*

*s**k*1*H**m**x*\**r*mod*q*

4.      Kirim pesan m beserta tanda- tangan r dan s.

**Prosedur Verifikasi Keabsahan Tanda-tangan (Verifying)**

1.      Hitung

*w*=  mod *q*

= (*H*(*m)*\**w*)mod *q*

= (*r*\**w*)mod *q*

*v*= (( \* )mod *p*)mod *q*

2.      Jika v = r, maka tanda-tangan sah, yang berarti bahwa pesan masih asli dan dikirim oleh pengirim yang benar.

**2.2        KASUS DSA (DIGITAL SIGNATURE ALGORITHM)**

Kasus ini memperlihatkan bagaimana pengirim memproses pesan sehingga didapat tanda-tangan digital dengan menggunakan algoritma DSA. Kemudian penerima melakukan verifikasi tanda-tangan digital tadi. Juga diperlihatkan pengaruh pada tanda-tangan digital jika pesan asli yang diterima penerima telah dimodifikasi pada saat pengiriman pesan. Untuk contoh kasus, pesan dan ukuran kunci yang digunakan adalah sebagai berikut:

1.  Pesan asli:

     UAS Kriptografi diadakan hari

     kamis tgl 11 September 2008

2.  Pesan Modifikasi

  UAS Kriptografi diadakan hari kamis

  tgl 18 September 2008

3.  Ukuran Kunci : 512

Dari hasil pengujian didapatkan:

1.            **Pemilihan parameter DSA.**

Dimana nilai parameter-parameter DSA yaitu kunci privat, x, kunci publik, y, prima, p;  sub\_prima, q; dan basis, g.

Pada panel *key generation,*untuk ukuran kunci 512 maka didapatkan hasil seperti pada Tabel 1.

|  |  |  |
| --- | --- | --- |
| Parameter | Nilai | Byte |
| X | 69740090510395  91325499276868  93854335526802  766055 | 20 |
| Y | 10462052936752  65067170877748  81303542036397  77333746902057  08914966185801  56609908665674  73125180011426  88167011281201  97131378402385  19608888755071  24792727315043  5 | 64 |
| P | 13232376895198  61240754793071  82674357577285  27029623408872  24515603975771  30290363687191  46452186041204  23735052178524  03370487520714  62798273003935  64623677745922  3 | 65 |
| Q | 85739377120809  42021042596279  90318636601332  086981 | 21 |
| G | 54216440574364  75141609648488  32570512804742  83943804743768  34667300766108  26261390054268  12890807137245  97310673074119  35513608579598  20973906708903  67185141189796 | 64 |

Dari tabel diatas dapat dilihat bahwa:

1.      Nilai parameter kunci privat, *x*= 20 byte = 20 x 8 = **160 bit**

Dari hasil pengujian nilai *x*sebesar 48 digit. Hal ini benar, jika nilai kunci harus berada dalam kisaran 0 ≤ nilai kunci ≤ 2 panjang kunci -1. Hal ini dapat dibuktikan dengan:

2160 = 1,46150164...x 1048 = 48 digit

Dimana 0 ≤ nilai kunci ≤ 2 panjang kunci – 1≈0 ≤ 2160 ≤ 2160 ─1

Nilai x = 1,46150164...x1048≤ ( 1,46150164...x1048)─ 1

Hal ini benar karena nilai *x*hasil pengujian 48 digit ≤ 48 digit (nilai parameter *x*).

2.      Nilai parameter kunci publik, *y*= 64byte = 64 x 8 = **512 bit**

Berdasarkan hasil pengujian nilai *y*sebesar 154 digit. Hal ini benar, apabila nilai kunci berada dalam kisaran 0 ≤ nilai kunci ≤ 2 panjang kunci – 1. Hal ini dapat dibuktikan dengan:

2512 = 1,34078079...x 10154 = 154 digit

dimana 0 ≤ nilai kunci ≤ 2 panjang kunci – 1 ≈ 0 ≤ 2512 ≤ 2512 ─1

Nilai y = 1,3408079...x10154≤  (1,34078079...x10154)─ 1

Hal ini benar karena nilai *y*hasil pengujian 154 digit ≤ 154 digit (nilai parameter*y*).

3.      Nilai parameter *p*= 65 byte = 65 x 8 = **520 bit**

Dari hasil pengujian nilai *p*sebesar 154 digit. Setiap digit desimal yang tertampilkan, sebenarnya adalah 65 byte atau 520 bit. Hal ini benar,jika nilai kunci harus berada dalam kisaran 0 ≤ nilai kunci ≤ 2 panjang kunci – 1. Hal ini dapat dibuktikan dengan:

2520 = 3,43239883...x 10156 = 156 digit

dimana 0 ≤ nilai kunci ≤ 2 panjang kunci – 1 ≈ 0 ≤ 2520 ≤ 2520 ─1

Nilai x = 3,43239883 ...x 10156≤ (3,43239883 ...x 10156) ─ 1

Hal ini benar karena nilai *p*hasil pengujian 154 digit ≤ 156 digit (nilai parameter*p*).

4.      Nilai parameter *q*= 21 byte = 21 x 8 = **168 bit**

Berdasarkan hasil pengujian dengan ukuran kunci sebesar 512 bit, maka dihasilkan nilai q sebesar 48 digit. Hal ini benar, jika nilai kunci harus berada dalam kisaran 0 ≤ nilai kunci ≤ 2panjang kunci– 1.

Hal ini dapat dibuktikan dengan:

2168 = 3,74144419...x 1050 = 50 digit

dimana 0 ≤ nilai kunci ≤ 2 panjang kunci – 1 ≈ 0 ≤ 2168 ≤ 2168 ─1

Nilai x =3,74144419 ...x 1050≤ ( 3,74144419 ...x 1050) ─ 1

Hal ini benar karena nilai *q*hasil pengujian 48 digit ≤ 50 digit (nilai parameter *q*)

5.      Nilai parameter *g*= 64 byte = 64 x 8 = **512 bit**

            Dari hasil pengujian nilai *g*sebesar 154 digit. Sehinggasetiap digit desimal yang tertampilkan, sebenarnyaadalah 64 byte atau 512 bit. Hal ini dapat dibuktikandengan:

2512 = 1,34078079...x 10154 = 154 digit

dimana 0 ≤ nilai kunci ≤ 2 panjang kunci – 1≈ 0 ≤ 2512≤ 2512 ─1

Nilai x = 1,3408079..x10154 ≤ ( 1,34078079...x10154)─ 1

Hal ini benar karena nilai g hasil pengujian 154 digit ≤ 154 digit (nilai parameter g).

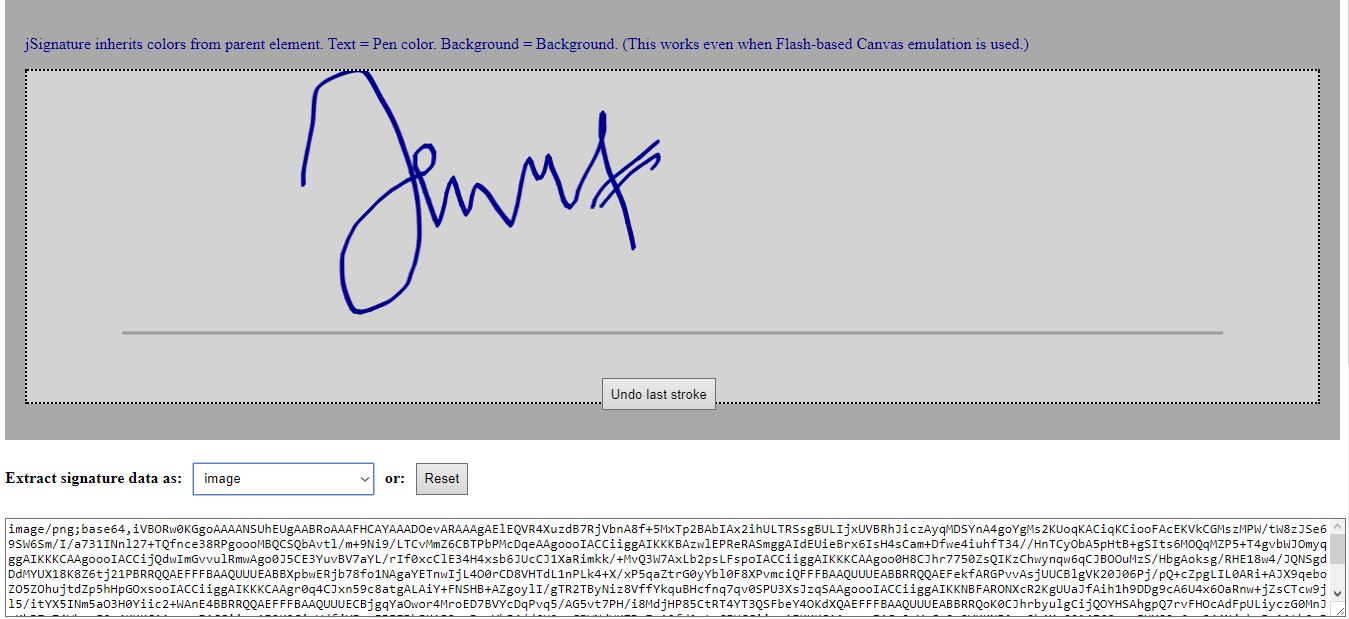
**2.         Signature Generation**.

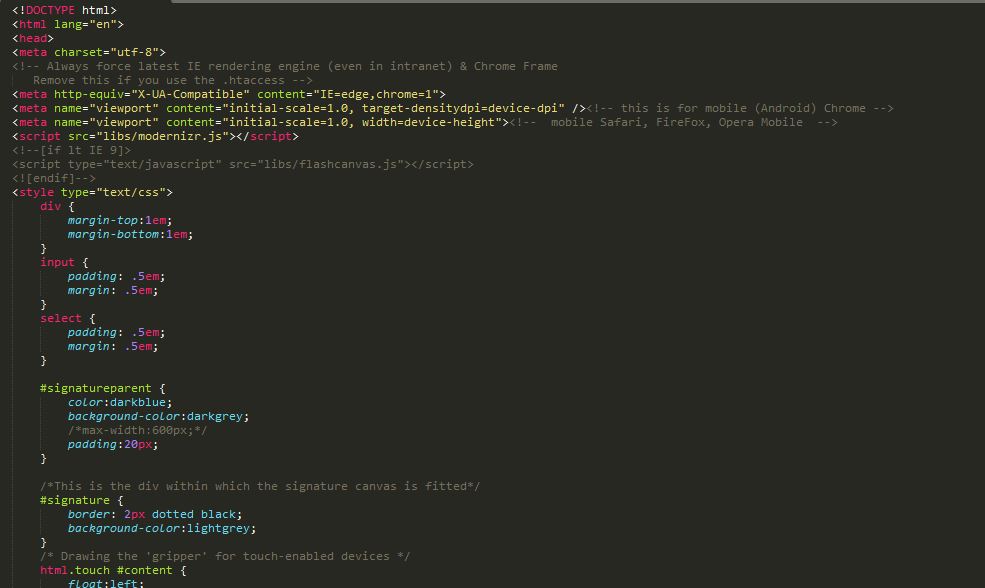
            Pada tahap ini sender mengambil pasangan kunci privat dan publik yang telah disimpan. Untuk kemudian sender mengisi pesan yang akan dikirm. Dimana pesan tersebut akan diperoleh nilai *message* *digest*nya menggunakan fungsi *hash*SHA-1. Ketika pesan dengan sebarang panjang < *264*bit dimasukkan, SHA-menghasilkan 160 bit keluaran yang disebut sebagai *message digest*. *Message digest*ini kemudian dimasukkan ke dalam DSA. Setelah didapat kunci privat dan kunci publik maka tahap selanjutnya adalah membangkitkan *digital signature*dari pesan yang akan dikirim.

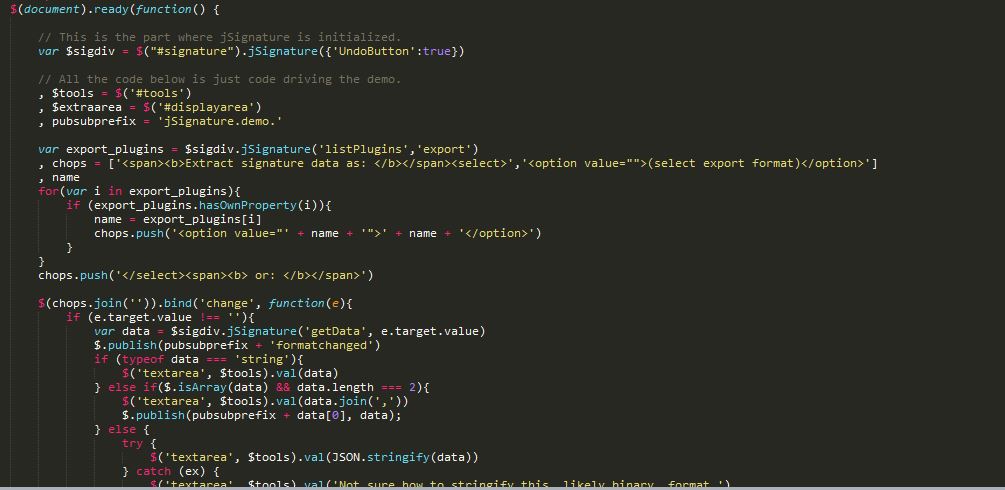
**3.         Signature Verification.**

Disisi penerima, tanda-tangan diverifikasi untuk dibuktikan keotentikannya dengan cara sebagai berikut: *pertama,*Tandatangan digital (r,s) didekripsi dengan menggunakan kunci publik pengirim pesan, menghasilkan*message digest*semula, MD. *Kedua, p*enerima kemudian mengubah pesan M menjadi *message digest*MD’ menggunakan fungsi *hash*satu arah yang sama dengan fungsi *hash*yang digunakan oleh pengirim. *Ketiga, J*ika MD = MD’, berarti tanda-tangan yang diterima otentik dan berasal dari pengirim yang benar.

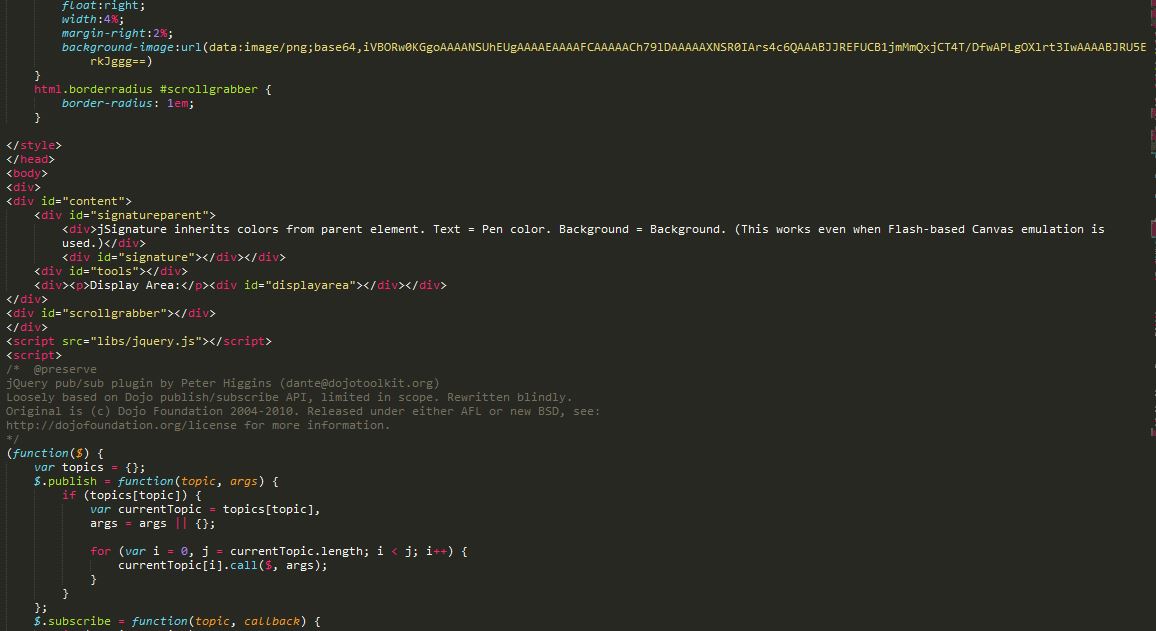
Sangat mungkin terjadi penyadapan pesan oleh *eavesdropper*sewaktu pesan M ditransmisikan dari sender ke receiver. Pesan M dimodifikasi menjadi M’ dan kemudian bersama tanda-tangan digital, pesan asli M dikirimkan kembali ke penerima. Jika terjadi hal seperti itu maka dengan tanda-tangan digital bisa diketahui apakah pesan yang dikirim masih utuh atau sudah dimodifikasi. Sebagai contoh, jika pesan asli dimodifikasi menjadi “UAS Kriptografi diadakan hari kamis tgl 11 September 2008” kemudian diverifikasi maka outputnya adalah “INVALID DIGITAL SIGNATURE”. Hal ini terjadi karena tanda-tangan digital tadi tidak valid untuk pesan yang dimodifikasi. Dengan begitu maka tanda-tangan digital pada DSA juga bisa memeriksa keutuhan pesan asli yang dikirim oleh sender. Dari simulasi DSA ini dapat disimpulkan bahwa tanda-tangan digital pada DSA memenuhi tiga layanan keamanan informasi yaitu keaslian data.

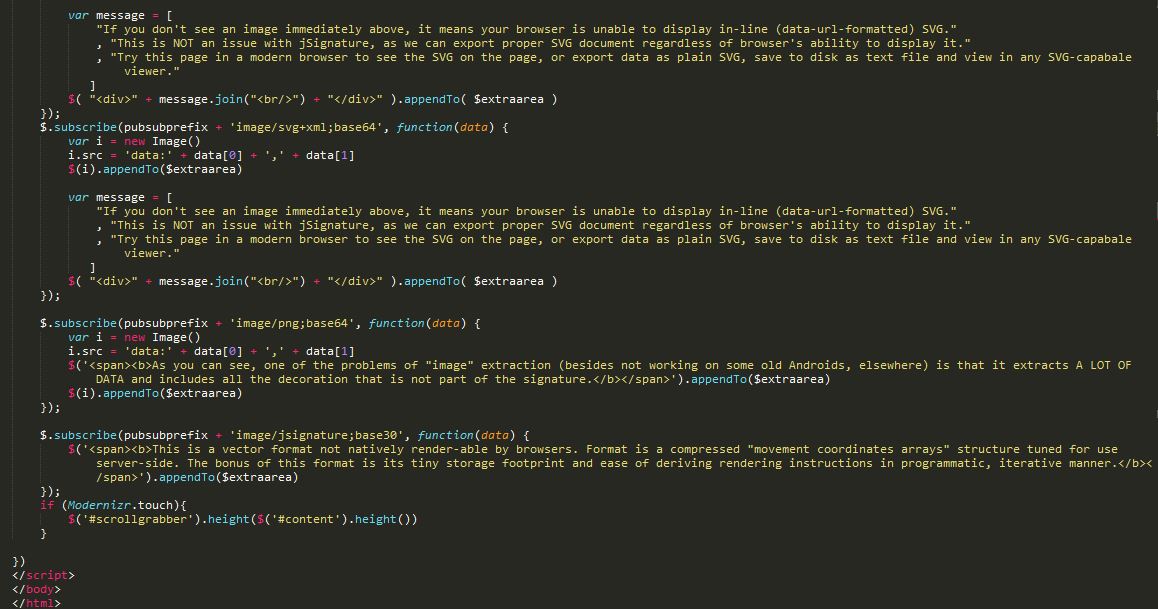


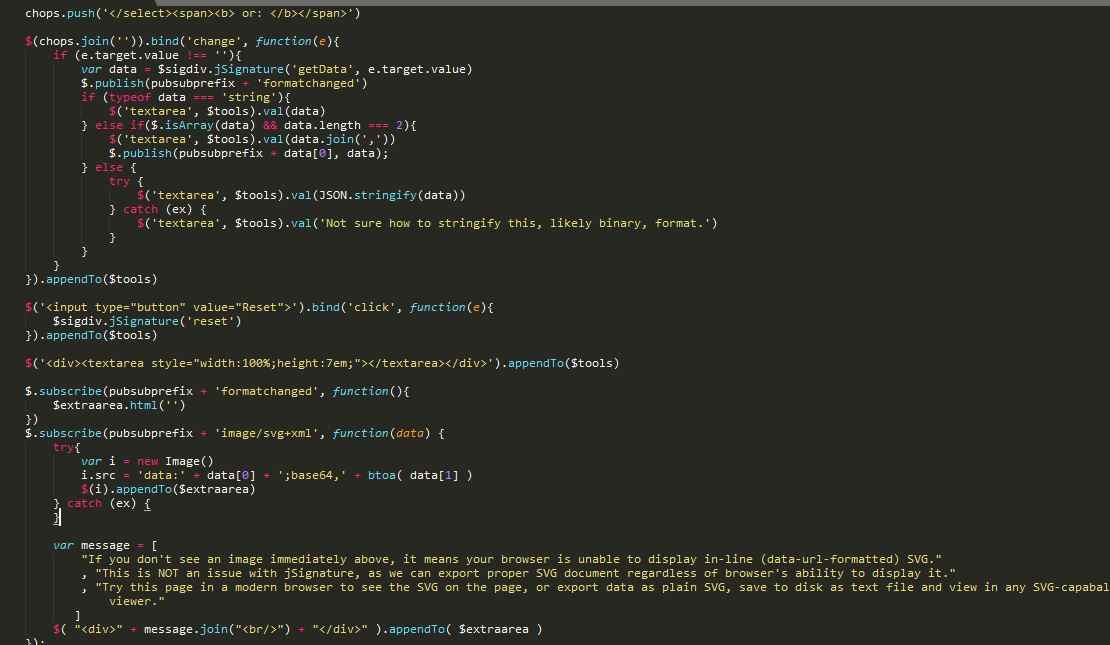












KESIMPULAN

Kriptografi merupakan salah satu dari media komunikasi dan informasi kuno yang masih dimanfaatkan hingga saat ini. Kriptografi di Indonesia disebut persandian yaitu secara singkat dapat berarti seni melindungi data dan informasi dari pihak-pihak yang tidak dikehendaki baik saat ditransmisikan maupun saat disimpan. Sedangkan ilmu persandiannya disebut kriptologi yaitu ilmu yang mempelajari tentang bagaimana tehnik melindungi data dan informasi tersebut beserta seluruh ikutannya.Pengguna diberikan ID dan password untuk mengakses sistem yang ada. Password dienkripsi untuk mencegah terjadinya akses illegal terhadap sistem misalnya pencurian data-data penting oleh mereka yang tidak berhak. Demikian juga enkripsi pada file-file penting dapat dilakukan (misalnya file yang berisi data keuangan). Metode enkripsi yang digunakan dapat berbentuk enkripsi kunci simetris, misalnya menggunakan algoritma DES, RSA, dll. Untuk mendapatkan algoritma enkripisi ini tidak dibutuhkan biaya karena telah dipublikasikan secara umum. Oleh karena itu, dapat disimpulkan bahwakriptografi masih merupakan sistem yang efektif dalam hal keamanan dan proteksi serta dapat digunakan secara luas di berbagai bidang usaha dan teknologi.

**DAFTAR PUSTAKA**

<http://unlimitedinformatic.blogspot.co.id/2016/06/tugas-kelompok-kriptografi-dsa-digital.html>

<https://github.com/NaufalHSyahputra/CryptoTools/>