PENGGUNAAN ALGORITMA BLOWFISH DALAM KRIPTOGRAFI

Candra Alim Sutanto

Program Studi Teknik Informatika
Institut Teknologi Bandung
Jl. Ganeca no.10 Bandung
e-mail: alphac_10@yahoo.com

ABSTRAK

Dalam kriptografi terdapat berbagai algoritma sandi yang bisa digunakan untuk mengamankan suatu data.

Salah satunya adalah algoritma Blowfish yang bisa digunakan untuk mengenkripsi suatu data. Blowfish termasuk algoritma kunci simetris yang memiliki kunci yang sama untuk mengenkripsi dan mendekripsi sebuah data. Algoritma Blowfish termasuk cipher blok.dan sampai saat ini masih dianggap aman karena belum ada attack yang benar-benar mematahkan algoritma Blowfish.

Makalah ini membahas mengenai bagaimana Algoritma Blowfish mengenkripsi sebuah data. Selain itu dalam makalah ini juga dibahas yang perlu dilakukan agar algoritma Blowfish bisa bekerja dengan optimal.

Kata kunci: Kriptografi, Kunci Simetris, Algoritma Blowfish.

1. PENDAHULUAN

Kriptografi, secara umum adalah ilmu dan seni untuk menjaga kerahasiaan berita [bruce Schneier - *Applied Cryptography*]. Selain pengertian tersebut terdapat pula pengertian ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan data, keabsahan data, integrasi data, serta autentifikasi [A. Menezes, P. van Oorschot and S. Vanstone - *Handbook of Applied Cryptography*]. Tidak semua aspek keamanan informasi ditangani oleh kriptografi.

Ada empat tujuan mendasar dari ilmu kriptografi ini yang juga merupakan aspek keamanan informasi yaitu :

1. Kerahasiaan, adalah layanan yang digunakan untuk menjaga isi dari informasi dari siapapun kecuali yang memiliki otoritas ataukunci rahasia untuk membuka/mengupas informasi yang telah disandi.
2. Integritas data, adalah berhubungan dengan penjagaan dari perubahan data secara tidak sah. Untuk menjaga integritas data, sistem harus memiliki kemampuan untuk mendeteksi manipulasi data oleh pihak-pihak yang tidak berhak, antara lain penyisipan, penghapusan, dan pensubsitusian data lain kedalam data yang sebenarnya.
3. Autentifikasi, adalah berhubungan dengan identifikasi/pengenalan, baik secara kesatuan sistem maupun informasi itu sendiri. Dua pihak yang saling berkomunikasi harus saling memperkenalkan diri. Informasi yang dikirimkan melalui kanal harus diautentikasi keaslian, isi datanya, waktu pengiriman, dan lain-lain.
4. Non-reoudiasi, atau nirpenyangkalan adalah usaha untuk mencegah terjadinya penyangkalan terhadap pengiriman/terciptanya suatu informasi oleh yang mengirimkan/membuat.

Makalah ini bertujuan untuk mempelajari algoritma Blowfish secara keseluruhan agar dapat memahami cara kerja dan struktur algoritmanya, kemudian mempelajari strategi perancangan sehingga bisa optimal. Dengan mempelajari algoritma Blowfish bisa di buat sebuah aplikasi yang memanfaatkan algoritma ini dan bisa digunakan untuk mengenkripsi data yang diinginkan.

2. ALGORITMA SANDI

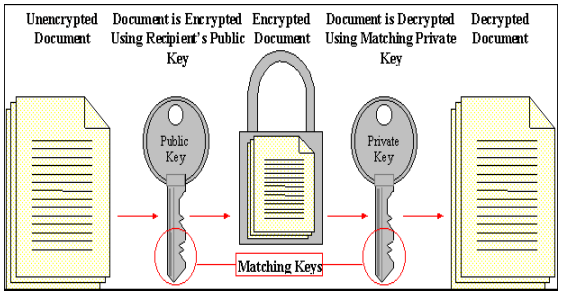
Algoritma sandi adalah algoritma yang berfungsi untuk melakukan tujuan kriptografis. Algoritma tersebut harus memiliki kekuatan untuk melakukan (dikemukakan oleh Shannon): pembingungan dan peleburan, sehingga dapat digunakan untuk mengamankan informasi.

Algoritma sandi yang handal adalah algoritma sandi yang kekuatannya terletak pada kunci, bukan pada kerahasiaan algoritma itu sendiri. Teknik dan metode untuk menguji kehandalan algoritma sandi adalah kriptoanalisis.

Algoritma sandi berdasarkan kesamaan kunci dibagi menjadi dua, yaitu kunci asimetris dan kunci simetris.

Pada sistem kunci-asimentris digunakan sepasang kunci yang berbeda, umumnya disebut kunci public (*public key*) dan kunci pribadi (*private key*), digunakan untuk proses enkripsi dan proses dekripsinya. Jadi kunci untuk membuat pesan yang disandikan berbeda dengan kunci untuk membuka pesan yang disandikan itu. Beberapa contoh algoritma yang menggunakan kunci-asimetris : Knapsack, RSA (Rivert-Shamir-Adelman), Diffie-Hellman.

Sedangakan pada skema kunci-simetris, digunakan sebuah kunci rahasia yang sama untuk melakukan proses enkripsi dan dekripsinya. Algoritma sandi kunci simetris adalah jenis kriptografi yang paling umum dipergunakan. Kunci untuk membuat pesan yang disandikan sama dengan kunci untuk membuka pesan yang disandikan itu. Siapapun yang memiliki kunci tersebut, dapat membuat dan membongkar rahasia ciphertext. Problem yang paling jelas disini terkadang bukanlah masalah pengiriman ciphertext-nya, melainkan masalah bagaimana menyampaikan kunci simetris tersebut kepada pihak yang diinginkan. Beberapa contoh algoritma yang menggunakan kunci-simetris: DES (*Data Encryption Standard*), Blowfish, Twofish, MARS, IDES, 3DES - DES diaplikasikan 3 kali, AES (*Advanced Encryption Standard*).



Gambar 1. Kriptografi kunci simetris

Sebuah skema algoritma sandi akan disebut kunci-simetris apabila untuk setiap proses enkripsi maupun dekripsi data secara keseluruhan digunakan kunci yang sama. Skema ini berdasarkan jumlah data per proses dan alur pengolahan data didalamnya dibedakan menjadi dua kelas, yaitu *stream-cipher* dan *block-cipher*.

Stream-cipher merupakan sebuah algoritma sandi yang mengenkripsi data persatuan data, seperti bit, byte, nible atau per lima bit (saat data yang di enkripsi berupa data Boudout). Setiap mengenkripsi satu satuan data di gunakan kunci yang merupakan hasil pembangkitan dari kunci sebelum.

Block-cipher adalah skema algoritma sandi yang akan membagi-bagi teks terang yang akan dikirimkan dengan ukuran tertentu (disebut blok) dengan panjang t, dan setiap blok dienkripsi dengan menggunakan kunci yang sama.

Semenjak pertama kali ditemukan, telah banyak penemuan-penemuan baru dalam penerapan *cipher* blok. Salah satunya adalah algoritma Blowfish yang dirancang oleh Bruce Schneier pada tahun 1993. Sejak saat itu, telah dilakukan berbagai macam analisis, dan perlahan-lahan mulai mendapat penerimaan sebagai algoritma enkripsi yang kuat.

3. ALGORITMA BLOWFISH

Blowfish atau yang disebut juga "OpenPGP.Cipher.4″ adalah algoritma kunci simetrik cipher blok yang dirancang pada tahun 1993 oleh Bruce Schneider untuk menggantikan DES (*Data Encryption Standard*). Algoritma Blowfish dibuat untuk digunakan pada komputer yang mempunyai microposesor besar (32-bit keatas dengan cache data yang besar).

Pada saat itu banyak sekali rancangan algoritma yang ditawarkan, namun hampir semua terhalang oleh paten atau kerahasiaan pemerintah Amerika. Schneier menyatakan bahwa blowfish bebas paten dan akan berada pada domain publik. Dengan pernyataan Schneier tersebut blowfish telah mendapatkan tempat di dunia kriptografi, khususnya bagi masyarakat yang membutuhkan algoritma kriptografi yang cepat, kuat, dan tidak terhalang oleh lisensi.

Blowfish dirancang dan diharapkan mempunyai kriteria perancangan yang diiginkan sebagai berikut :

1. Cepat, Blowfish melakukan enkripsi data pada microprocessor 32-bit dengan rate 26 clock cycles per byte.
2. *Compact*, Blowfish dapat dijalankan pada memory kurang dari 5K.
3. Sederhana, Blowfish hanya menggunakan operasi – operasi sederhana, Blowfish hanya menggunakan operasi – operasi sederhana, seperti : penambahan, XOR, dan lookup tabel pada operan32-bit.
4. Memiliki tingkat keamanan yang bervariasi, panjang kunci yang digunakan oleh Blowfish dapat bervariasi dan bisa sampai sepanjang minimal 32-bit, maksimal 448 -bit, Multiple 8 bit, default 128 bit

Namun, dalam penerapannya sering kali algortima ini menjadi tidak optimal. Karena strategi implementasi yang tidak tepat. Algoritma Blowfish akan lebih optimal jika digunakan untuk aplikasi yang tidak sering berganti kunci, seperti jaringan komunikasi atau enkripsi file otomatis.

3.1 Kriptoanalisis

Dr Dobb's Journal mensponsori sebuah kontes kriptoanalisis terhadap algoritma Blowfish. Total ada lima pengiriman, yang telah memberikan analisinya terhadap algoritma Blowfish.

1. John Kelsey mengembangkan sebuah serangan yang bisa memecahkan 3-bulat Blowfish, namun tidak mampu untuk memperpanjang itu. Serangan ini memanfaatkan fungsi F dan fakta bahwa mod tambahan 232 dan XOR tidak berubah.
2. Vikramjit Singh Chhabra memandang cara untuk mengeefisienkan adalah dengan menerapkan *brute force keysearch machine*.
3. Serge Vaudenay memeriksa varian yang disederhanakan dari Blowfish, dengan S-kotak dikenal dan dan tidak *key-dependent*. Untuk varian ini, sebuah serangan diferensial dapat memulihkan P-array dengan 28r 1 dipilih plainteks (r adalah jumlah putaran). Serangan ini tidak mungkin untuk 8-bulat Blowfish dan yang lebih tinggi, karena lebih plaintext diperlukan daripada mungkin dapat dihasilkan dengan 64-bit blok cipher.
4. Tesis Ph.D milik Vincent Rijmen mencantumkan second-order differential attack pada 4 iterasi algoritma Blowfish. Namun, attack tersebut tidak dapat dilanjutkan lagi untuk iterasi selanjutnya.

Kunci lemah tertentu menghasilkan S-kotak lemah (kemungkinan untuk mendapatkan mereka secara acak adalah 1 dalam 2^{14}), serangan yang sama memerlukan 24r 1 dipilih untuk memulihkan plainteks P-array (dengan asumsi S-kotak yang dikenal). Dengan S-kotak yang tidak diketahui, serangan ini dapat mendeteksi apakah kunci lemah sedang digunakan, tetapi tidak dapat menentukan apapun (baik S-kotak, P-array, maupun kunci itu sendiri). Serangan ini hanya bekerja terhadap pengurangan-bulat varian; itu benar-benar tidak efektif terhadap 16-bulat Blowfish.

Meskipun demikian, penemuan kunci lemah di Blowfish adalah signifikan. Sebuah kunci lemah adalah salah satu yang dua entri untuk suatu S-box adalah identik. Ada cara untuk memeriksa kunci lemah sebelum melakukan *key expansion*. Lakukan ekspansi kunci dan periksa identik entri S-box setelah dihasilkan kunci Blowfish apabila memang dikhawatirkan terjadi *weak key*.

3.2. Struktur Algoritma Blowfish

Blowfish merupakan blok cipher 64-bit dengan panjang kunci variabel. Algoritma ini terdiri dari dua bagian: *key expansion* atau perluasan kunci dan enkripsi data.

3.2.1. Key-Expansion

Berfungsi merubah kunci (Minimum 32-bit, Maksimum 448-bit) menjadi beberapa array subkunci (*subkey*) dengan total 4168 byte.

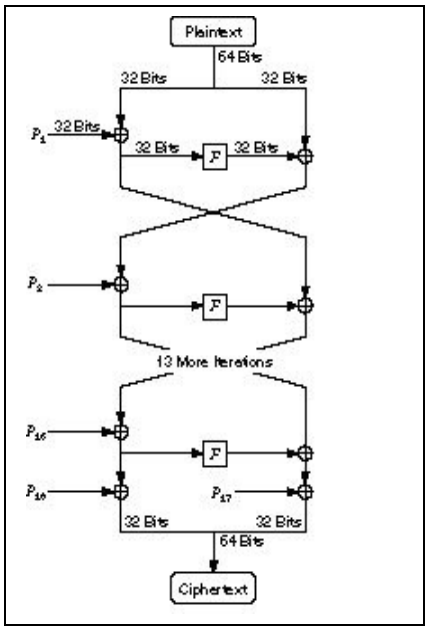
3.2.2. Enkripsi Data

Terdiri dari iterasi fungsi sederhana (*Feistel Network*) sebanyak 16 kali putaran. Setiap putaran terdiri dari permutasi kunci-dependent dan substitusi kunci- dan data-dependent. Semua operasi adalah penambahan (*addition*) dan XOR pada variabel 32-bit. Operasi tambahan lainnya hanyalah empat penelusuran tabel (*table lookup*) array berindeks untuk setiap putaran.

Untuk alur algoritma enkripsi dengan metoda Blowfish dijelaskan sebagai berikut :

1. Bentuk inisial array P sebanyak 18 buah (P1,P2, ……..P1 masing-msing bernilai 32-bit.
   Array P terdiri dari delapan belas kunci 32-bit subkunci :
   P1,P2,…….,P18
2. Bentuk S-box sebanyak 4 buah masing-masing bernilai 32-bit yang memiliki masukan 256.
   Empat 32-bit S-box masing-masing mempunyai 256 entri :
   S1,0,S1,1,……..,S1,255
   S2,0,S2,1,……..,S2,255
   S3,0,S3,1,……..,S3,255
   S4,0,S4,1,……..,S4,255
3. Plainteks yang akan dienkripsi diasumsikan sebagai masukan, Plainteks tersebut diambil sebanyak 64-bit, dan apabila kurang dari 64-bit maka kita tambahkan bitnya, supaya dalam operasi nanti sesuai dengan datanya.
4. Hasil pengambilan tadi dibagi 2, 32-bit pertama disebut XL, 32-bit yang kedua disebut XR.
5. Selanjutnya lakukan operasi
   XL = XL xor Pi dan XR = F(XL) xor XR
6. Hasil dari operrasi diatas ditukar XL menjadi XR dan XR menjadi XL.
7. Lakukan sebanyak 16 kali, perulangan yang ke-16 lakukan lagi proses penukaran XL dan XR.
8. Pada proses ke-17 lakukan operasi untuk
   XR = XR xor P17 dan XL = XL xor P18.
9. Proses terakhir satukan kembali XL dan XR sehingga menjadi 64-bit kembali.

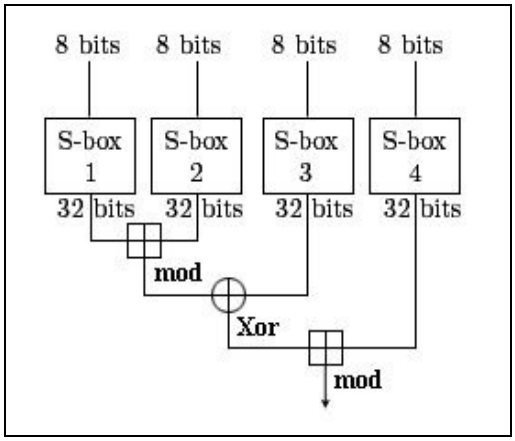
Blowfish menggunakan jaringan Feistel yang terdiri dari 16 buah putaran. Skema jaringan Feistel dapat dilihat di gambar 1.



Gambar 2. Jaringan Feistel untuk Algoritma Blowfish

Algoritma Blowfish memiliki keunikan dalam hal proses dekripsi, yaitu proses dekripsi dilakukan dengan urutan yang sama persis dengan proses enkripsi, hanya saja pada proses dekripsi P1, P2, …, P18 digunakan dalam
urutan yang terbalik.

Dalam algoritma Blowfish juga terdapat fungsi f. Berikut ini gambar mengenai fungsi f tersebut.



Gambar 3. Fungsi f dalam algoritma Blowfish

Sebelumnya dijelaskan bahwa Array P terdiri dari delapan belas subkunci. Subkunci dihitung menggunakan algoritma Blowfish, metodenya adalah sebagai berikut :

1. Pertama-tama inilialisasi P-array dan kemudian empat S-box secara berurutan dengan string yang tetap. String ini terdiri atas digit hexadesimal dari Pi.
2. XOR P1 dengan 32-bit pertama kunci, XOR P2 dengan 32-bit kedua dari kunci dan seterusnya untuk setiap bit dari kunci (sampai P18).Ulangi terhadap bit kunci sampai seluruh P-array di XOR dengan bit kunci.
3. Enkrip semua string nol dengan algoritma Blowfish dengan menggunakan subkunci seperti dijelaskan pada langkah (1) dan (2).
4. Ganti P1 dan P2 dengan keluaran dari langkah (3).
5. Enkrip keluaran dari langkah (3) dengan algoritma Blowfish dengan subkunci yang sudah dimodifikasi.
6. Ganti P3 dan P4 dengan keluaran dari langkah (5).
7. Lanjutkan proses tersebut, ganti seluruh elemen dari P-array, kemudian seluruh keempat S-box berurutan, dengan keluaran yang berubah secara kontiyu dari algoritma Blowfish.

Secara keseluruhan terdapat 521 iterasi atau putaran yang dibutuhkan untuk membangkitkan seluruh upa-kunci yang dibutuhkan. Aplikasi kemudian dapat menyimpan upa-kunci yang telah dihasilkan. Proses pembangkitan kunci ini tidak perlu selalu dilakukan setiap saat.

3.3. Keamanan Algoritma Blowfish

Sampai saat ini algoritma Blowfish belum ditemukan kelemahan yang berarti hanya adanya *weak key* dimana dua entri dari S-box mempunyai nilai yang sama. Belum ada cara untuk mengecek *weak key* sebelum melakukan *key expansion*, tetapi hal ini tidak berpengaruh terhadap hasil enkripsi.

Hasil enkripsi dengan algoritma Blowfish sangat tidak mungkin dan tidak praktis untuk di terjemahkan tanpa bantuan kunci. Sampai kini belum ada *Cryptoanalyst* yang dapat membongkar pesan tanpa kunci yang dienkripsi dengan memakai bantuan algoritma Blowfish. Agar aman dari pembongkaran pesan maka dalam algoritmanya harus menggunakan 16 putaran agar pesan tersebut tidak dapat dibongkar.

Algoritma Blowfish pun dapat digabungkan dengan algoritma-algoritma enkripsi yang lain dalam pengkripsian sebuah pesan untuk lebih menjamin isi dari pesan tersebut. Sehingga algoritma Blowfish cukup aman jika ingin digunakan untuk mengenkripsi data yang ingin di amankan.

3.4. Penggunaan Algoritma Blowfish

Blowfish adalah salah satu algoritma cipher blok yang tercepat dan digunakan secara luas di dunia, kecuali ketika pergantian kunci. Setiap kunci baru memerlukan pemrosesan awal yang sebanding dengan mengenkripsikan teks denga ukuran sekitar 4 kilobyte. Pemrosesan awal ini sangat lambat dibandingkan dengan algoritma cipher blok lainnya. Hal ini menyebabkan

Blowfish tidak mungkin digunakan dalam beberapa aplikasi, tetapi tidak menimbulkan masalah dalam banyak aplikasi lainnya.

Pemrosesan awal yang lama pada Blowfish digunakan sebagai ide untuk metode *password-hashing* yang digunakan pada OpenBSD. Metode *password-hashing* ini menggunakan algoritma yang diuturnnkan dari algoritma Blowfish yang menggunakan penjadwalan kunci yang lambat. Algoritma ini digunakan dengan pertimbangan bahwa usaha komputasi ekstra yang harus dilakukan dapat memberikan proteksi lebih terhadap serangan terhadap password berbasiskan kamus (*dictionary attacks*).

Dalam beberapa implementasi, Blowfish memerlukan memori yang relatif besar, yaitu sekitar 4 kilobyte. Hal ini tidak menjadi masalah bahkan untuk komputer desktop dan laptop yang sudah berumur tua. Tetapi hal ini juga membuat implementasi Blowfish pada embedded system terkecil (seperti pada smartcard pada awal kemunculannya) tidak mungkin untuk dilakukan.

3.4.1. Strategi Penerapan Perangkat Lunak

Hal-hal yang harus dipertimbangkan dalam perancangan perangkat lunak yang menerapkan algoritma Blowfish antara lain
adalah prinsip perancangan algoritma Blowfish (sifat dasar algoritma Blowfish), yaitu:

1. Lebih optimal jika digunakan untuk aplikasi yang tidak sering berganti kunci.\

2. Lebih cepat jika diterapkan pada 32-bit mikroprosessor dengan data cache yang besar. Namun, karena hal ini merupakan batasan teknis sehingga tidak akan dipakai dalam strategi perancangan.

3. Lebih aman jika diterapkan tanpa adanya pengurangan jumlah iterasi (dengan asumsi user tidak menggunakan *weak key*).

Setelah mencari beberapa referensi dan beberapa artikel di internet, ternyata algoritma Blowfish tidak memiliki karakteristik lainnya yang harus dipenuhi untuk menjalankan enkripsi dengan menggunakan algoritman Blowfish. Ternyata selama aplikasi yang digunakan tidak sering berganti kunci, algoritma Blowfish merupakan algoritma yang cepat, dan selama tidak dilakukan pengurangan jumlah iterasi (dengan asumsi tidak ada penggunaan *weak key* oleh user) algoritma Blowfish merupakan algoritma enkripsi yang aman.

3.4.1. Aplikasi Pengguna Algoritma Blowfish

Penggunaan algoritma Blowfish antara lain terdapat pada:

1. 96Crypt, oleh fever.link.
96Crypt merupakan aplikasi untuk enkripsi dan dekripsi arsip dan folder.
2. A-Lock, oleh Trillium Technology Group.A
A-Lock merupakan perangkat lunak enkripsi yang terintegrasi dengan aplikasi e-mail untuk Windows yang terkenal.
3. Access Manager, oleh Citi-Software Ltd.A
Aplikasi password manager untuk sistem operasi Windows.
4. AntiSnoop Password Dropper.
Sebuah password basis data yang dirancang untuk melindungi terhadap keystrokes loggers, mouse monitors, dan sebagainya.
5. Canner, oleh Cinnabar Systems.
Digunakan untuk melindungi kode Java dari reverse engineering dengan cara membuat native Windows executable yang mengandung kelas dan resource dan aplikasi dalam keadaan terenkripsi. Kelas dan resource ini kemudian akan didekripsikan pada memori ketika muncul permintaan dari mesin virtual Java.
6. Nautilus.
Merupakan produk telepon berbasiskan internet yang aman. Produk ini dapat digunakan dan didapatkan secara gratis.
7. PuTTY, oleh Simon Tatham.
Klien SSH untuk Win32. Aplikasi ini dapat digunakan dan didapatkan secara gratis.

IV. KESIMPULAN

Kesimpulan yang dapat diambil dari studi terhadap algoritma Blowfish untuk kepentingan kriptografi data adalah sebagi berikut.

1. Algoritma Blowfish merupakan salah satu solusi yang baik untuk mengatasi masalah keamanan dan kerahasiaan data yang memerlukan kriptografi di dalamnya.
2. Algoritma Blowfish merupakan algoritma yang menggunakna jaringa feistel dan tingkat keamanannya ditentukan oleh jumlah iterasi dan panjang kunci yang digunakan.
3. Implementasi algoritma Blowfish yang optimal dapat dilakukan dengan aplikasi yang tidak sering berubah-ubah kunci.
4. Setiap algoritma sandi memiliki kelemah, demikian juga algortima Blowfish. Dalam algoritma blowfish mungkin terjadi *weak key* dimana kemungkinannya terjadinya cukup kecil.

REFERENSI

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

[2] http://id.wikipedia.org/wiki/blowfish. Tanggal akses: 17 Desember 2009.

[2] http://id.wikipedia.org/wiki/kriptografi Tanggal akses: 17 Desember 2009.

[2] http://www.schneier.com/blowfish.html Tanggal akses: 15 Desember 2009.

[2] http://www.schneier.com/paper-blowfish-oneyear.html Tanggal akses: 15 Desember 2009.

[2] http://www.splashdata.com/splashid/blowfish.htm Tanggal akses: 15 Desember 2009.