APLIKASI SISTEM UJIAN ONLINE PADA LOCAL AREA NETWORK DENGAN METODE KRIPTOGRAFI GOVERNMENT STANDARD

Indra Oloan Nainggolan

Widyaiswara Muda
Kementerian Perindustrian R.I - Balai Diklat Industri Medan

olo.nainggol123@gmail.com

ABSTRAK

Proses ujian pada suatu Institusi pendidikan dapat dikomputerisasi dengan membangun suatu sistem ujian online. Sistem Ujian Online adalah sistem ujian yang dilakukan dengan menggunakan bantuan komputer dan dilakukan secara online melalui jaringan Local Area Network (LAN). Kumpulan pertanyaan dan jawaban akan dimasukkan pada bagian server. Kemudian, komputer client akan meminta koneksi kepada server dengan memasukkan data username dan password yang valid. Setelah itu, proses ujian akan dilakukan secara serentak setelah adanya instruksi dari komputer server. Proses koneksi antar komputer ini dimungkinkan dengan menggunakan teknologi yang diperkenalkan Microsoft, yaitu Microsoft Windows Socket Library, atau lebih sering disebut dengan 'Windows Socket'. Selain itu, untuk mengamankan pertanyaan yang dikirimkan antar komputer dan jawaban yang tersimpan di dalam database, maka perlu digunakan sebuah algoritma kriptografi. Dalam hal ini, digunakan algoritma kunci simetris Government Standard (GOST). Soal yang dibagikan juga diacak urutannya untuk meminimalisir terjadinya kecurangan. Aplikasi hasil rancangan dapat diterapkan untuk mengadakan proses ujian pada sebuah jaringan komputer dengan melakukan instalasi aplikasi pada komputer server dan client. Laporan yang dihasilkan terdiri dari laporan daftar nilai siswa dan laporan daftar siswa.

Kata Kunci: Sistem Ujian *Online, Local Area Network, Kriptografi Government Standard*

ABSTRACT

The examination process in an educational institution can be computerized by building an online examination system. Online Testing System is an examination system that is carried out using computer assistance and is carried out online through a network Local Area Network (LAN). A collection of questions and answers will be entered in the server section. Then, the client computer will request a connection to the server by entering valid username and password data. After that, the test process will be carried out simultaneously after the instructions from the server computer. The connection process between computers is made possible by using technology introduced by Microsoft, namely the Microsoft Windows Socket Library, or more commonly called the 'Windows Socket'. In addition, to secure questions sent between computers and answers stored in a database, it is necessary to use a cryptographic algorithm. In this case, the Government Standard (GOST) symmetric key algorithm is used. The questions that were distributed were also randomized in order to minimize fraud. Application design results can be applied to carry out the testing process on a computer network by installing applications on server and client computers. The resulting report consists of a student grade list report and a student list report.

Keywords : *Online Test System, Local Area Network, Government Standard Cryptography*

I. PENDAHULUAN

Pada zaman teknologi informasi ini, hampir semua kegiatan manusia telah menggunakan bantuan komputer. Namun, sampai saat ini, banyak sekolah dan universitas yang masih menggunakan sistem manual dalam memberikan ujian kepada para siswa / mahasiswanya. Sistem ini memiliki beberapa kelemahan, seperti memerlukan biaya yang sangat besar untuk membeli kertas yang akan digunakan dalam proses ujian, rentan terhadap kebocoran soal

ujian, rentan terhadap kecurangan pada saat proses ujian dan membutuhkan tenaga guru / dosen untuk melakukan koreksi hasil jawaban secara manual yang rentan terhadap *human error*.

Untuk mengatasi masalah ini, maka proses ujian dapat dikomputerisasi, dengan menggunakan bantuan komputer. Proses ujian dapat dilakukan pada sebuah jaringan komputer. Program ini terbagi menjadi 2 bagian, yaitu bagian *server* dan bagian *client*. Kumpulan pertanyaan dan jawaban akan dimasukkan pada bagian *server*. Kemudian, komputer *client* akan meminta koneksi kepada komputer *server* dengan memasukkan data *username* dan *password* yang valid. Setelah itu, proses ujian akan dilakukan secara serentak setelah adanya instruksi dari komputer *server*. Aplikasi *server* harus dijalankan terlebih dahulu dan aplikasi *client* akan mengkoneksikan diri dengan aplikasi *server*. Proses koneksi antar komputer ini dimungkinkan dengan menggunakan teknologi yang diperkenalkan *Microsoft*, yaitu *Microsoft Windows socket Library*, atau lebih sering disebut dengan '*Windows socket*'. Selain itu, untuk mengamankan pertanyaan yang dikirimkan antar komputer dan jawaban yang tersimpan di dalam *database*, maka perlu digunakan sebuah algoritma kriptografi. Dalam penelitian ini, digunakan algoritma kunci simetris *Government Standard* (GOST). Metode GOST merupakan algoritma *block cipher* yang dikembangkan oleh seseorang yang berkebangsaan Uni Soviet, yang memiliki jumlah proses sebanyak 32 *round* (putaran) dan menggunakan 64 bit *block cipher* dengan 256 bit *key*. Dengan menggunakan metode kriptografi ini, maka seandainya *server* dibobol oleh *hacker* atau pihak yang tidak bertanggung jawab, maka data pertanyaan dan jawaban tetap aman, karena tersimpan dalam keadaan acak atau terenkripsi.

II. Tinjauan Pustaka

2.1 Jaringan Komputer

2.1.1 Definisi

Jaringan komputer dapat didefinisikan sebagai hubungan antara dua atau lebih komputer beserta periferal lainnya melalui media transmisi untuk melakukan komunikasi data satu dengan yang lain.

Adapun komunikasi data dapat diartikan pengiriman data secara elektronik dari satu tempat ke tempat lain melalui suatu media komunikasi, dan data yang dikirimkan tersebut merupakan hasil atau akan diproses oleh suatu sistem komputer.

Dalam jaringan ada tiga komponen utama yang harus dipahami. Komponen utama dalam jaringan komputer) yaitu :

1. *Host* atau *node*, yaitu sistem komputer yang berfungsi sebagai sumber atau penerima dari data yang dikirimkan. *Node* ini dapat berupa,
   a. *Server*: komputer tempat penyimpanan data dan program-program aplikasi yang digunakan dalam jaringan,
   b. *Client*: komputer yang dapat mengakses sumber daya (berupa data dan program aplikasi) yang ada pada server,
   c. *Shared pheriperal*: peralatan-peralatan yang terhubung dan digunakan dalam jaringan (misalnya, printer, scanner, harddisk, modem, dan lain-lain).
2. *Link*, adalah media komunikasi yang menghubungkan antara node yang satu dengan node lainnya. Media ini dapat berupa saluran transmisi kabel dan tanpa kabel.
3. *Software* (Perangkat Lunak), yaitu program yang mengatur dan mengelola jaringan secara keseluruhan. Termasuk di dalamnya sistem operasi jaringan yang berfungsi sebagai pengatur komunikasi data dan periferal dalam jaringan.

2.1.2. Tipe Jaringan

Ada beberapa tipe jaringan komputer yang umumnya digunakan. Berikut ini beberapa klasifikasi tipe jaringan komputer yang ada :

1. Berdasarkan letak geografis terbagi atas,
   a. *Local Area Network (LAN)*, berada pada satu bangunan atau lokasi yang sama, dengan kecepatan transmisi data yang tinggi (mulai dari 10 Mbps).
   b. *Metropolitan Area Network (MAN)*, merupakan gabungan beberapa LAN yang terletak pada satu kota (jangkauan 50 sampai 75 mil) yang dihubungkan dengan kabel khusus atau melalui saluran telepon, dengan kecepatan transmisi antara 56 Kbps sampai 1 Mbps.
   c. *Wide Area Network (WAN)*, merupakan gabungan dari komputer LAN atau MAN yang ada di seluruh permukaan bumi ini yang dihubungkan dengan saluran telepon, gelombang elektromagnetik, atau satelit.
2. Berdasarkan arsitektur jaringan terbagi atas:
   a. Jaringan *peer to peer*.
   b. Jaringan berbasis *server* (*server-based network*/*server-client network*).
   c. Jaringan *hibrid*.

3. Berdasarkan teknologi transmisi terbagi atas:
   a. Jaringan *switch*, merupakan jaringan yang penyampaian informasi dari pengirim ke penerima melalui mesin-mesin perantara atau saluran telepon.
   b. Jaringan *broadcast*, merupakan jaringan yang penyampaian informasi dari pengirim ke penerima dilakukan secara broadcast (disiarkan ke segala arah) baik melalui saluran kabel maupun saluran tanpa kabel.

II.1.3 IP Address

IP Address adalah alamat yang diberikan untuk jaringan komputer yang menggunakan protokol TCP IP. IP address terdiri dari 32 bit angka biner yang dapat dituliskan menjadi 4 kelompok bilangan decimal yang dipisahkan dengan tanda titik, sebagai contoh : 192.168.10.1. IP address terdiri dari 2 bagian yaitu network ID dan host ID, network ID adalah alamat dari jaringan komputer sedangkan host ID adalah alamat dari host seperti komputer, router dan lain-lain. Pemberian IP address pada komputer harus bersifat unik sehingga IP address setiap Pembagian Kelas IP Address adalah sebagai berikut:

1. IP Address Kelas A, merupakan IP address dengan jumlah yang sangat besar, sehingga biasanya digunakan untuk jaringan yang sangat besar dengan jumlah host yang sangat banyak. Sebagai contoh pada penggunaan IP address : 113.46.5.6 , 113 berfungsi sebagai network ID sedangkan 46.5.6 berfungsi sebagai host ID nya.
2. IP Address Kelas B, merupakan IP address dengan jumlah host yang sedang, jumlah maksimal host berkisar 65.534 host, sehingga IP ini cocok untuk jaringan dengan jumlah host yang tidak terlalu besar dan tidak terlalu kecil. Sebagai contoh penggunaan IP address Kelas B adalah : 132.92.121.1 , 132.92 berfungsi sebagai network ID sedangkan 121.1 berfungsi sebagai host ID.
3. IP Address Kelas C, merupakan IP address dengan jumlah host yang sangat kecil sehingga IP address ini digunakan untuk jaringan kecil seperti disekolah-sekolah, dikantor-kantor maupun instansi rumahan, jumlah maksimal host pada IP address ini hanya 254 host. Seabagai contoh penggunaan IP Address Kelas C adalah : 192.168.1.2 , 192.168.1 merupakan network ID dan 2 merupakan host ID-nya.

II.2 Kriptografi

Menurut sejarahnya, kriptografi sudah lama digunakan oleh tentara Sparta di Yunani pada permulaan tahun 400 SM. Teknik kriptografi dikenal dengan nama transposisi *cipher*, yang merupakan metode enkripsi tertua.

Kriptografi adalah ilmu dan seni untuk menjaga keamanan pesan. Keamanan pesan diperoleh dengan menyandikannya menjadi pesan yang tidak mempuyai makna. Zaman sekarang, kerahasiaan informasi menjadi sesuatu yang penting. Informasi yang rahasia perlu disembunyikan agar tidak diketahui oleh orang yang tidak berhak. Seseorang tentu tidak ingin nomor PIN kartu kredit atau kartu ATM-nya diketahui orang. Atau, jika suatu pesan ditulis secara rahasia dan tidak ingin diketahui atau dibaca oleh orang lain. Kriptografi dapat digunakan untuk menyamarkan informasi yang bersifat rahasia dari orang atau pihak yang tidak berhak membacanya.

II.2.1 Definisi

Kata kriptografi (*cryptography*) berasal dari bahasa Yunani, yaitu *kriptos* yang artinya *secret* (rahasia), dan *graphein*, yang artinya *writing* (tulisan). Jadi, kriptografi berarti *secret writing* (tulisan rahasia). Namun saat ini kriptografi lebih dari sekadar *privacy*, tetapi juga untuk tujuan *data integrity*, *authentication*, dan *non-repudiation*.

II.2.2 Tujuan dan Manfaat

Kriptografi bertujuan untuk memberi layanan keamanan (yang juga dinamakan sebagai aspek keamanan) dan memberikan manfaat sebagai berikut:

1. Kerahasiaan (*confidentiality*), adalah layanan yang ditujukan untuk menjaga agar pesan tidak dapat dibaca oleh pihak-pihak yang tidak berhak. Di dalam kriptografi, layanan ini direalisasikan dengan menyandikan pesan menjadi *ciphertext* (pesan tidak terbaca).
2. Integritas data (*data integrity*), adalah layanan yang menjamin bahwa pesan masih asli / utuh atau belum pernah dimanipulasi selama pengiriman. Untuk menjaga integritas data, sistem harus memiliki kemampuan untuk mendeteksi manipulasi pesan oleh pihak-pihak yang tidak berhak, antara lain penyisipan, penghapusan, dan pensubstitusian 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 *origin authentication*).

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

II.2.3 Jenis Algoritma

Berdasarkan jenis kunci yang digunakan, terdapat 2 (dua) jenis algoritma kriptografi, yaitu:

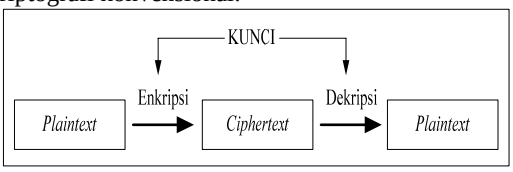
1. Algoritma Simetri (konvensional)
2. Algoritma Asimetri (kunci-publik)

II.2.3.1 Algoritma Simetri

Algoritma Simetri, atau disebut juga dengan algoritma konvensional, adalah algoritma yang menggunakan kunci enkripsi yang sama dengan kunci dekripsinya. Disebut konvensional, karena algoritma yang biasa digunakan orang sejak berabad-abad ynag lalu adalah algoritma jenis ini.

Algoritma Simetri sering juga disebut dengan algoritma kunci rahasia, algoritma kunci tunggal atau algoritma satu kunci dan mengharuskan pengirim dan penerima menyetujui suatu kunci tertentu sebelum mereka dapat berkomunikasi dengan aman. Keamanan algoritma simetri tergantung pada kunci. Membocorkan kunci berarti bahwa pihak lain dapat melakukan proses enkripsi dan dekripsi terhadap pesan yang sifatnya rahasia. Agar komunikasi tetap aman, kunci harus tetap dirahasiakan.

Gambar II.1 memperlihatkan kriptografi simetri yang biasa disebut juga sebagai kriptografi kunci konvensional. Pesan *plaintext* P, dikodekan (dienkripsi) menjadi *ciphertext* menggunakan *password* (kunci K). Untuk mengembalikan *ciphertext* menjadi *plaintext* semula, dilakukan proses dekripsi dengan kunci yang sama seperti yang digunakan pada proses enkripsi. Karena kunci yang digunakan sama, maka disebut kriptografi kunci simetri. Dan karena jenis ini telah digunakan selama berabad-abad yang lalu, maka dinamakan pula kriptografi konvensional.



Gambar II.1 Prosedur Kerja Algoritma Simetris

Dalam dunia kriptografi, *password* sering disebut sebagai kunci. Pesan asli yang belum dikodekan disebut *plaintext*. *Plaintext* tidak harus berupa teks, namun dapat berupa *file* gambar, *file* biner, *file* suara dan format lainnya. *File* yang telah disandikan disebut *ciphertext*. Enkripsi adalah proses pengubahan pesan asli menjadi karakter yang tidak dapat dibaca. Sedangkan dekripsi adalah proses pengubahan karakter yang tidak dapat dibaca menjadi pesan asal. *Ciphertext* inilah yang biasanya dikirimkan melalui saluran *internet* yang rawan penyadapan.

Notasi matematika sering digunakan untuk mempermudah penulisan dan analisis, sehingga kriptografi modern selalu berhubungan dengan matematika. Dengan pesan asal P dan kode rahasia C yang diperoleh dari enkripsi dengan kunci K, dapat dituliskan:

$$\mathbf{C = E_k (P)}$$

Notasi ini menyatakan bahwa C dihasilkan oleh fungsi Enkripsi E yang dioperasikan terhadap masukan P dengan kunci K. Operasi ini dilakukan oleh pengirim pesan. Penerima pesan melakukan operasi sebaliknya:

$$\mathbf{P = E_k (C)}$$

Operasi ini adalah proses dekripsi untuk kembali mendapatkan pesan asal P. Pemecah kode (*cryptanalyst*) sering kali hanya memiliki C dan harus menemukan nilai P.

Algoritma Simetri dapat dibagi dalam dua kategori. Jenis pertama beroperasi pada *plaintext* yang berupa satu bit tunggal pada satu waktu, yang disebut *stream algorithms* (algoritma aliran atau *stream ciphers*). Jenis kedua beroperasi pada *plaintext* dalam grup bit-bit. Grup bit-bit ini disebut blok, dan algoritmanya disebut sebagai algoritma blok atau kode rahasia blok. Untuk algoritma komputer modern, ukuran blok dasarnya adalah 64 bit atau 128 bit, cukup besar untuk menghindari analisis pemecahan kode dan cukup kecil agar dapat bekerja dengan cepat. Sebelum pemakaian komputer, algoritma biasanya beroperasi pada *plaintext*, satu karakter per satu operasi.

II.2.3.2 Algoritma Asimetri

Algoritma Asimetri, atau disebut juga dengan algoritma kunci publik, didesain sedemikian rupa sehingga kunci yang digunakan untuk enkripsi berbeda dengan kunci yang digunakan untuk dekripsi. Kunci dekripsi tidak dapat dihitung dari kunci enkripsi. Algoritma ini disebut kunci publik karena kunci enkripsi dapat dibuat publik yang berarti semua pihak boleh mengetahuinya. Pihak

manapun dapat menggunakan kunci enkripsi untuk melakukan enkripsi terhadap pesan, namun hanya orang tertentu (calon penerima pesan dan sekaligus pemilik kunci dekripsi yang merupakan pasangan kunci publik), yang dapat melakukan dekripsi terhadap pesan tersebut. Dalam sistem ini, kunci enkripsi sering disebut kunci publik, sementara kunci dekripsi sering disebut kunci privat. Kunci privat kadang-kadang disebut kunci rahasia. Yang termasuk algoritma asimetri adalah ECC, LUC, RSA, El Gamal dan DH.

Enkripsi dengan kunci publik K_e dinyatakan sebagai:

$$\mathbf{E_{Ke}\,(P) = C}$$

Dengan kunci privat (K_d) sebagai pasangan kunci publik (K_e), dekripsi dengan kunci privat yang bersesuaian dapat dinyatakan dengan:

$$\mathbf{D_{Kd}\,(C) = P}$$

Disini, K_e merupakan pasangan K_d. Artinya tidak ada K_d lain yang dapat digunakan untuk melakukan dekripsi kode C yang merupakan hasil enkripsi dengan kunci K_e. Sebaliknya, pesan dapat dienkripsi dengan kunci privat dan didekripsi dengan kunci publik. Metode ini digunakan pada skema tanda tangan digital. Artinya kunci privat dan kunci publik digunakan secara berlawanan dengan tujuan yang berbeda. Sifat ini hanya berlaku untuk algoritma kunci publik tertentu semacam RSA.

II.3 Metode Government Standard (GOST)

GOST merupakan singkatan dari "*Gosudarstvennyi Standard*" atau "*Government Standard*". Metode GOST merupakan algoritma *block cipher* yang dikembangkan oleh seseorang yang berkebangsaan Uni Soviet. Kemudian, metode ini dikembangkan oleh pemerintah Uni Soviet pada masa perang dingin untuk menyembunyikan data atau informasi yang bersifat rahasia pada saat komunikasi.

GOST merupakan blok cipher 64 bit dengan panjang kunci 256 bit. Algoritma ini mengiterasi algoritma enkripsi sederhana sebanyak 32 putaran (*round*). Untuk mengenkripsi pertama-tama plainteks 64 bit dipecah menjadi 32 bit bagian kiri, L dan 32 bit bagian kanan, R. Subkunci (subkey) untuk putaran *i* adalah K_i. Pada awalnya, bagian kanan data ditambah dengan subkunci ke-i modulus 232. Hasilnya dipecah menjadi delapan bagian 4 bit dan setiap bagian menjadi input s-box yang berbeda. Di dalam GOST terdapat 8 buah s-box, 4 bit pertama menjadi s-box pertama, 4 bit kedua menjadi s-box kedua, dan seterusnya. Output dari 8 s-box kemudian dikombinasikan menjadi bilangan 32 bit kemudian bilangan ini dirotasi 11 bit kekiri. Akhirnya hasil operasi ini di-xor dengan data bagian kiri yang kemudian menjadi bagian kanan dan bagian kanan menjadi bagian kiri (*swap*). Pada implementasinya nanti rotasi pada fungsi f dilakukan pada awal saat inisialisasi sekaligus membentuk s-box 32 bit dan dilakukan satu kali saja sehingga lebih menghemat operasi dan dengan demikian mempercepat proses enkripsi/dekripsi. Subkunci dihasilkan secara sederhana yaitu dari 256 bit kunci yang dibagi menjadi delapan 32 bit blok : k1, k2, …, k8. Setiap putaran menggunakan subkunci yang berbeda. Dekripsi sama dengan enkripsi dengan urutan k_i dibalik.

Kelebihan dari metode GOST adalah kecepatannya yang cukup baik, walaupun tidak secepat *Blowfish* tetapi lebih cepat dari IDEA.

Komponen dari metode GOST adalah:

1. *Key Store Unit* (KSU) menyimpan 256 *bit string* dengan menggunakan 32 *bit register* (K0, K1, …, K7).
2. Dua buah 32 *bit register* (L_i, R_i).
3. 32 *bit adder modulo* 2^{32} (CM1).
4. *Bitwise Adder XOR* (CM2).
5. *Substitusion block* (S) yaitu berupa 8 buah 64 *bit S-Box*.
6. *Rotasi Left* (R) sebanyak 11 *bit*.

II.3.1 Proses Pembentukan Kunci

Proses pembentukan kunci dapat dilihat pada penjabaran berikut ini :

1. *Input key* berupa 256 bit kunci dengan perincian $k_1, k_2, k_3, k_4, \ldots, k_{256}$.
2. *Input key* ini akan dikelompokkan dan dimasukkan ke dalam 8 buah KSU dengan aturan seperti berikut,

   K0 = ($k_{32}, \ldots, k_1$)
   K1 = ($k_{64}, \ldots, k_{33}$)
   K2 = ($k_{96}, \ldots, k_{65}$)
   K3 = ($k_{128}, \ldots, k_{97}$)
   K4 = ($k_{160}, \ldots, k_{129}$)
   K5 = ($k_{192}, \ldots, k_{161}$)
   K6 = ($k_{224}, \ldots, k_{193}$)
   K7 = ($k_{256}, \ldots, k_{225}$)

II.3.2 Proses Enkripsi

Proses enkripsi dari metode GOST untuk satu putaran (iterasi) dapat dapat dilihat pada penjabaran berikut ini,

1. 64 *bit plaintext* dibagi menjadi 2 buah bagian 32 *bit*, yaitu L_i dan R_i.

   Caranya : *Input* a (1),a(2),…,a(32),b(1),…, b(32)

L0 = b(32), b(31), …, b(1)
R0 = a(32), a(31), …, a(1)

2. $(R_i + K_i)$ mod 2^{32}. Hasil dari penjumlahan modulo 2^{32} berupa 32 bit.
3. Hasil dari penjumlahan modulo 2^{32} dibagi menjadi 8 bagian, dimana setiap bagian terdiri dari 4 bit. Setiap bagian dimasukkan ke dalam tabel *S-Box* yang berbeda, 4 bit pertama menjadi input dari *S-Box* pertama, 4 bit kedua menjadi *S-Box* kedua, dan seterusnya. *S-Box* yang digunakan adalah :

Tabel II.1 *S-Box* dari Metode GOST

Tabel S-Box	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S-Box 0	4	10	9	2	13	8	0	14	6	11	1	12	7	15	5	3
S-Box 1	14	11	4	12	6	13	15	10	2	3	8	1	0	7	5	9
S-Box 2	5	8	1	13	10	3	4	2	14	15	12	7	6	0	9	11
S-Box 3	7	13	10	1	0	8	9	15	14	4	6	12	11	2	5	3
S-Box 4	6	12	7	1	5	15	13	8	4	10	9	14	0	3	11	2
S-Box 5	4	11	10	0	7	2	1	13	3	6	8	5	9	12	15	14
S-Box 6	13	11	4	1	3	15	5	9	0	10	14	7	6	8	2	12
S-Box 7	1	15	13	0	5	7	10	4	9	2	3	14	6	11	8	12

Cara melihat *S-Box* yaitu *input* biner diubah menjadi bilangan desimal dan hasilnya menjadi urutan bilangan dalam *S-Box*.

Tabel II.2 Penjelasan Cara Kerja *S-Box* dari Metode GOST

Posisi	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S-Box 0	4	10	9	2	13	8	0	14	6	11	1	12	7	15	5	3

Contoh,jika data input ke *S-Box* adalah 5 maka dicari data pada posisi *S-Box* ke-5. *Output* yang dihasilkan dari *S-Box* adalah 8.

4. Hasil yang didapat dari substitusi ke *S-Box* digabungkan kembali menjadi 32 *bit* dan kemudian dilakukan rotasi kiri sebanyak 11 bit.
5. $R_{i+1} = R_i$ (hasil dari *rotate left*) XOR L_i.
6. $L_{i+1} = R_i$ sebelum dilakukan proses.

Proses penjumlahan modulo 2^{32}, *S-Box* dan *Rotate Left* dilakukan sebanyak 32 kali (putaran) dengan penggunaan kunci pada masing-masing putaran berbeda-beda sesuai dengan aturan berikut ini,

Putaran 0 – 7 : K0, K1, K2, …, K7
Putaran 8 – 15 : K0, K1, K2, …, K7
Putaran 16 – 23 : K0, K1, K2, …, K7
Putaran 24 – 31 : K7, K6, K5, …, K0

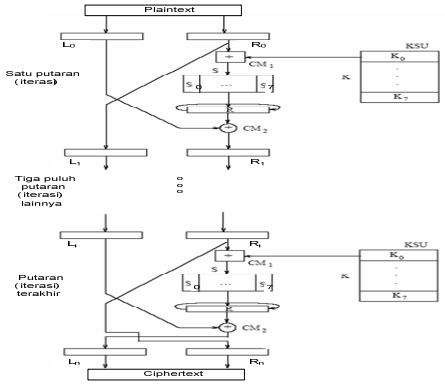
Untuk putaran ke-31, langkah 5 dan 6 agak sedikit berbeda. Langkah 5 dan 6 untuk putaran 31 adalah sebagai berikut,

L32 = L31 XOR R31 setelah proses.
R32 = R31 sebelum dilakukan proses

Sehingga, *ciphertext* yang dihasilkan adalah :

L32 : b(32), b(31), …, b(1)
R32 : a(32), a(31), …, a(1)
T = a(1), …, a(32), b(1), …, b(32)

Proses enkripsi dari metode GOST dapat digambarkan dalam bentuk skema seperti gambar II.2.



Gambar II.2 Skema Proses Enkripsi Metode GOST

II.3.3 Proses Dekripsi

Proses dekripsi merupakan kebalikan dari proses enkripsi. Penggunaan kunci pada masing-masing putaran pada proses dekripsi adalah sebagai berikut,

Putaran 0 – 7 : K0, K1, K2, …, K7
Putaran 8 – 15 : K7, K6, K5, …, K0
Putaran 16 – 23 : K7, K6, K5, …, K0
Putaran 24 – 31 : K7, K6, K5, …, K0

Algoritma yang digunakan untuk proses dekripsi sama dengan proses enkripsi dengan aturan untuk langkah 5 dan 6 pada putaran ke-31 adalah sebagai berikut,

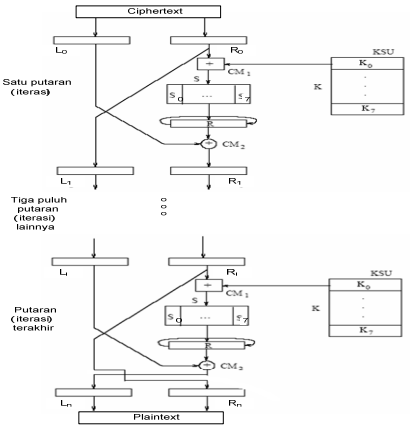
L32 = L31 XOR R31 setelah proses.
R32 = R31 sebelum dilakukan proses.

Plaintext yang dihasilkan pada proses dekripsi adalah :

L32 = b(32), b(31), …, b(1)
R32 = a(32), a(31), …, a(1)
P = a(1), …, a(32), b(1), …, b(32)

Proses dekripsi dari metode GOST dapat digambarkan dalam bentuk skema seperti terlihat pada gambar II.3.

Indra Oloan Nainggolan
APLIKASI SISTEM UJIAN ONLINE PADA LOCAL AREA NETWORK DENGAN METODE KRIPTOGRAFI GOVERNMENT STANDARD
Halaman 90-105



Gambar II.3 Skema Proses Deskripsi Metode GOST

II.4. Basis Data

Basis data adalah pusat sumber data yang dapat dipakai oleh banyak pemakai untuk berbagai aplikasi. Inti dari basis data adalah *database management system* yang memperbolehkan pembuatan, modifikasi, pembaharuan basis data, mendapatkan kembali data dan membangkitkan laporan.

Tujuan dari basis data yaitu:

a) Memastikan bahwa data dapat dipakai diantara pemakai untuk berbagai aplikasi.
b) Memelihara baik keakuratan maupun kekonsistenan.
c) Memastikan bahwa semua data yang diperlukan untuk aplikasi sekarang dan yang akan datang disediakan dengan cepat.
d) Membolehkan basis data untuk berkembang.

Tujuan yang telah disebutkan di atas memberikan keuntungan dan kerugian pendekatan basisdata. Pertama, pemakaian data berarti bahwa data perlu disimpan hanya sekali. Membantu mencapai integritas data, karena mengubah data yang diselesaikan lebih mudah dan dapat dipercaya jika data muncul hanya sekali dalam banyak *file* berbeda. Ketika pemakai memerlukan data khusus, basisdata yang dirancang dengan baik (*well-designed*) memenuhi lebih dahulu kebutuhan data yang demikian. Akibatnya, data memiliki kesempatan tersedia yang lebih baik dalam basisdata daripada dalam sistem *file* yang konvensional. Basisdata yang dirancang dengan baik juga lebih fleksibel daripada *file* terpisah, karena itu, basisdata dapat berkembang seperti pada perubahan kebutuhan pemakai dan aplikasinya.

Akhirnya, pendekatan basis data memiliki keuntungan yang membolehkan pemakai untuk memiliki pandangan sendiri mengenai data. Pemakai tidak perlu memperhatikan struktur sebenarnya basisdata atau penyimpan fisiknya.

Kerugian pertama pendekatan basisdata adalah bahwa semua data disimpan dalam satu tempat. Oleh karena itu, data lebih mudah diserang bencana dan membutuhkan *backup* yang lengkap. Terdapat risiko bahwa administrator basisdata menjadi satu-satunya orang yang mempunyai hal istimewa atau kemampuan cukup untuk mendekati data. Prosedur birokratis perlu untuk memodifikasi atau memperbaharui basisdata secara lengkap yang terlihat tidak dapat diatasi.

Kerugian lain terjadi ketika usaha untuk mencapai dua tujuan efektif untuk mengatur sumber data, seperti:

1. Menjaga waktu yang diperlukan untuk *insert*, *update*, menghapus dan memperoleh kembali data untuk suatu jumlah yang dapat dipertahankan.
2. Menjaga harga penyimpanan data untuk jumlah yang dapat diterima.

Sebuah basisdata terdiri atas beberapa tabel (sesuai dengan kebutuhan program). Tabel adalah kumpulan dari *record-record* sejenis dengan panjang elemen yang sama tapi *data value*nya berbeda. Sebuah tabel terdiri atas beberapa *record*. *Record* adalah kumpulan dari atribut-atribut yang menginformasikan sebuah entitas secara lengkap. Sebuah *record* terdiri atas beberapa *field*. *Field* adalah item-item yang terdapat pada sebuah entitas yang dapat bertindak sebagai pengenal bagi entitas tersebut.

III. PEMBAHASAN

III.1.1 Metode GOST

Metode kriptografi GOST akan digunakan untuk mengenkripsi pertanyaan di dalam *database*. Metode GOST menggunakan 256 bit kunci atau 32 karakter ASCII (256 / 8 = 32). *Input* berupa 32 karakter ASCII akan diubah menjadi 256 bit biner dengan perincian k_1, k_2, k_3, k_4, …, k_{256}. Setiap karakter diubah menjadi nilai ASCII, sesuai dengan tabel ASCII seperti terlihat pada tabel berikut :

Tabel III.1 Tabel Karakter ASCII

Decimal	Hexadecimal	Binary	Octal	Char	Decimal	Hexadecimal	Binary	Octal	Char	Decimal	Hexadecimal	Binary	Octal	Char
0	0	0	0	[NULL]	48	30	110000	60	0	96	60	1100000	140	`
1	1	1	1	[START OF HEADING]	49	31	110001	61	1	97	61	1100001	141	a
2	2	10	2	[START OF TEXT]	50	32	110010	62	2	98	62	1100010	142	b
3	3	11	3	[END OF TEXT]	51	33	110011	63	3	99	63	1100011	143	c
4	4	100	4	[END OF TRANSMISSION]	52	34	110100	64	4	100	64	1100100	144	d
5	5	101	5	[ENQUIRY]	53	35	110101	65	5	101	65	1100101	145	e
6	6	110	6	[ACKNOWLEDGE]	54	36	110110	66	6	102	66	1100110	146	f
7	7	111	7	[BELL]	55	37	110111	67	7	103	67	1100111	147	g
8	8	1000	10	[BACKSPACE]	56	38	111000	70	8	104	68	1101000	150	h
9	9	1001	11	[HORIZONTAL TAB]	57	39	111001	71	9	105	69	1101001	151	i
10	A	1010	12	[LINE FEED]	58	3A	111010	72	:	106	6A	1101010	152	j
11	B	1011	13	[VERTICAL TAB]	59	3B	111011	73	;	107	6B	1101011	153	k
12	C	1100	14	[FORM FEED]	60	3C	111100	74	<	108	6C	1101100	154	l
13	D	1101	15	[CARRIAGE RETURN]	61	3D	111101	75	=	109	6D	1101101	155	m
14	E	1110	16	[SHIFT OUT]	62	3E	111110	76	>	110	6E	1101110	156	n
15	F	1111	17	[SHIFT IN]	63	3F	111111	77	?	111	6F	1101111	157	o
16	10	10000	20	[DATA LINK ESCAPE]	64	40	1000000	100	@	112	70	1110000	160	p
17	11	10001	21	[DEVICE CONTROL 1]	65	41	1000001	101	A	113	71	1110001	161	q
18	12	10010	22	[DEVICE CONTROL 2]	66	42	1000010	102	B	114	72	1110010	162	r
19	13	10011	23	[DEVICE CONTROL 3]	67	43	1000011	103	C	115	73	1110011	163	s
20	14	10100	24	[DEVICE CONTROL 4]	68	44	1000100	104	D	116	74	1110100	164	t
21	15	10101	25	[NEGATIVE ACKNOWLEDGE]	69	45	1000101	105	E	117	75	1110101	165	u
22	16	10110	26	[SYNCHRONOUS IDLE]	70	46	1000110	106	F	118	76	1110110	166	v
23	17	10111	27	[ENG OF TRANS. BLOCK]	71	47	1000111	107	G	119	77	1110111	167	w
24	18	11000	30	[CANCEL]	72	48	1001000	110	H	120	78	1111000	170	x
25	19	11001	31	[END OF MEDIUM]	73	49	1001001	111	I	121	79	1111001	171	y
26	1A	11010	32	[SUBSTITUTE]	74	4A	1001010	112	J	122	7A	1111010	172	z
27	1B	11011	33	[ESCAPE]	75	4B	1001011	113	K	123	7B	1111011	173	{
28	1C	11100	34	[FILE SEPARATOR]	76	4C	1001100	114	L	124	7C	1111100	174	|
29	1D	11101	35	[GROUP SEPARATOR]	77	4D	1001101	115	M	125	7D	1111101	175	}
30	1E	11110	36	[RECORD SEPARATOR]	78	4E	1001110	116	N	126	7E	1111110	176	~
31	1F	11111	37	[UNIT SEPARATOR]	79	4F	1001111	117	O	127	7F	1111111	177	[DEL]
32	20	100000	40	[SPACE]	80	50	1010000	120	P					
33	21	100001	41	!	81	51	1010001	121	Q					
34	22	100010	42	"	82	52	1010010	122	R					
35	23	100011	43	#	83	53	1010011	123	S					
36	24	100100	44	$	84	54	1010100	124	T					
37	25	100101	45	%	85	55	1010101	125	U					
38	26	100110	46	&	86	56	1010110	126	V					
39	27	100111	47	'	87	57	1010111	127	W					
40	28	101000	50	(	88	58	1011000	130	X					
41	29	101001	51	)	89	59	1011001	131	Y					
42	2A	101010	52	*	90	5A	1011010	132	Z					
43	2B	101011	53	+	91	5B	1011011	133	[					
44	2C	101100	54	,	92	5C	1011100	134	\					
45	2D	101101	55	-	93	5D	1011101	135	]					
46	2E	101110	56	.	94	5E	1011110	136	^					
47	2F	101111	57	/	95	5F	1011111	137	_					

Kemudian, bit kunci akan dikelompokkan sebagai berikut:

K0 = (k_{32}, …, k_1)
K1 = (k_{64}, …, k_{33})
K2 = (k_{96}, …, k_{65})
K3 = (k_{128}, …, k_{97})
K4 = (k_{160}, …, k_{129})
K5 = (k_{192}, …, k_{161})
K6 = (k_{224}, …, k_{193})
K7 = (k_{256}, …, k_{225})

Sebagai contoh, misalkan *input* kunci kriptografi metode GOST adalah "ISTP", maka proses pembentukan kunci yang terjadi adalah sebagai berikut:

Panjang kunci 'ISTP' adalah 4 karakter, sedangkan kunci yang dibutuhkan GOST adalah 32 karakter. Oleh karena itu, tambahkan karakter spasi di belakang kunci 'ISTP' sebanyak 28 karakter, sehingga panjang kunci sama dengan 32 karakter Ascii (atau 256 bit biner).
Kunci = 'ISTP   '
Ubah kunci menjadi bit biner (k(1), k(2), ... , k(256))
Kunci =
0100100101010011010101000101000000100000000
1000000010000000100000001000000010000000010
0000001000000010000000100000001000000001000
0000100000000100000001000000010000000100000
0010000000100000001000000010000000100000000
1000000010000000100000001000000010000000010
0000

Bagi hasil konversi per 32 bit dan masukkan ke array kunci K.
K(0) = k(32), k(31), ... , k(1) =
00001010001010101100101010010010
K(0) = 0A2ACA92
K(1) = k(64), k(63), ... , k(33) =
00000100000001000000010000000100
K(1) = 04040404
K(2) = k(96), k(95), ... , k(65) =
00000100000001000000010000000100
K(2) = 04040404
K(3) = k(128), k(127), ... , k(97) =
00000100000001000000010000000100
K(3) = 04040404
K(4) = k(160), k(159), ... , k(129) =
00000100000001000000010000000100
K(4) = 04040404
K(5) = k(192), k(191), ... , k(161) =
00000100000001000000010000000100
K(5) = 04040404
K(6) = k(224), k(223), ... , k(193) =
00000100000001000000010000000100
K(6) = 04040404
K(7) = k(256), k(255), ... , k(225) =
00000100000001000000010000000100
K(7) = 04040404

Bit kunci yang dihasilkan oleh proses pembentukan kunci akan digunakan di dalam proses enkripsi / dekripsi sebagai berikut:

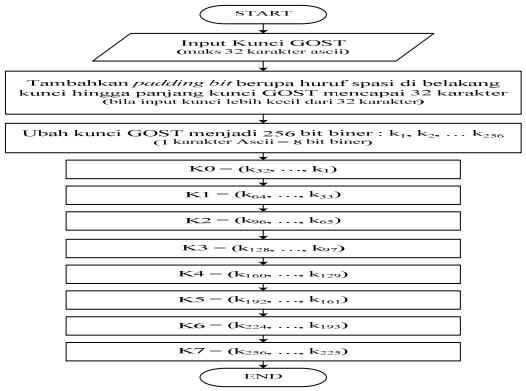
1. Proses Enkripsi

Putaran 0-7 menggunakan K(0),K(1),K(2), ...,K(7).
Putaran 8-15 menggunakan K(0),K(1),K(2),...,K(7).
Putaran16-23 menggunakanK(0),K(1),K(2),..., K(7).
Putaran 24-31 menggunakan K(7),K(6),K(5),..., K(0).

2. Proses Dekripsi

Putaran 0-7 menggunakan K(0),K(1),K(2),..., K(7).
Putaran 8-15 menggunakan K(7),K(6),K(5),..., K(0).
Putaran 16-23 menggunakan K(7),K(6),K(5),..., K(0).
Putaran 24-31 menggunakan K(7),K(6),K(5),.., K(0).

Proses pembentukan kunci pada metode GOST dapat digambarkan dalam bentuk *flowchart*, seperti terlihat pada gambar III.1.



Gambar III.1 Flowchart Proses Pembentukan Kunci Metode GOST

III.1.2.1 Proses Enkripsi

Di dalam metode GOST, proses enkripsi akan melalui 32 putaran yang memproses 64 bit pesan (8 karakter ASCII) dan mengubahnya menjadi 64 bit *cipher*.

Sebagai contoh, misalkan input kunci dalam karakter adalah "ISTP" dan *input* pesan yang akan di enkripsi adalah "KARAKTER", maka proses enkripsi yang terjadi adalah sebagai berikut:

PUTARAN-0

1) Plain Text : 'KARAKTER'
Konversi ke biner :
01001011010000010101001001000001010010110101010001000110101010010
Bagi biner menjadi 2 bagian, yaitu: a(1), a(2), ..., a(32) dan
b(1), b(2), ..., b(32)
L(0) = b(32), b(31), ..., b(1)
L(0) = 01001010101000100010101011010010 = 4AA22AD2
R(0) = a(32), a(31), ..., a(1)
R(0) = 10000010010010101000001011010010 = 824A82D2
2) Lakukan penambahan R(0) dengan K(0) dan hasilnya dimodulus 2^32
untuk menjaga panjang bit hasil penambahan tetap 32 bit.
Hasil = (R(0) + K(0)) mod 2^32
Hasil = (824A82D2 +) mod 2^32
Hasil = 824A82D2
3) Pecah hasil menjadi 8 kelompok biner, masing-masing 4 bit dan masukkan ke SBox.
Kelompok 0, 1000 = 8, masukkan ke SBox(0), hasilnya = 6 = 0110
Kelompok 1, 0010 = 2, masukkan ke SBox(1), hasilnya = 4 = 0100
Kelompok 2, 0100 = 4, masukkan ke SBox(2), hasilnya = 10 = 1010
Kelompok 3, 1010 = 10, masukkan ke SBox(3), hasilnya = 6 = 0110
Kelompok 4, 1000 = 8, masukkan ke SBox(4), hasilnya = 4 = 0100
Kelompok 5, 0010 = 2, masukkan ke SBox(5), hasilnya = 10 = 1010
Kelompok 6, 1101 = 13, masukkan ke SBox(6), hasilnya = 8 = 1000
Kelompok 7, 0010 = 2, masukkan ke SBox(7), hasilnya = 13 = 1101
4) Hasil mutasi SBox digabungkan kembali menjadi 32 bit dan
lakukan Rotate Left sebanyak 11 kali.
Hasil = RotateLeft (64A64A8D, 11)
Hasil = 32546B25
R(0) = 32546B25
5) R(1) = R(0) XOR L(0)
R(1) = 32546B25 XOR 4AA22AD2
R(1) = 78F641F7
6) L(1) = R(0) sebelum proses
L(1) = 824A82D2

PUTARAN-1

1) L(1) = 824A82D2
R(1) = 78F641F7
2) Lakukan penambahan R(1) dengan K(1) dan hasilnya dimodulus 2^32
untuk menjaga panjang bit hasil penambahan tetap 32 bit.
Hasil = (R(1) + K(1)) mod 2^32
Hasil = (78F641F7 + 04040404) mod 2^32
Hasil = 7CFA45FB
3) Pecah hasil menjadi 8 kelompok biner, masing-masing 4 bit dan masukkan ke SBox.
Kelompok 0, 0111 = 7, masukkan ke SBox(0), hasilnya = 14 = 1110
Kelompok 1, 1100 = 12, masukkan ke SBox(1), hasilnya = 0 = 0000
Kelompok 2, 1111 = 15, masukkan ke SBox(2), hasilnya = 11 = 1011
Kelompok 3, 1010 = 10, masukkan ke SBox(3), hasilnya = 6 = 0110
Kelompok 4, 0100 = 4, masukkan ke SBox(4), hasilnya = 5 = 0101
Kelompok 5, 0101 = 5, masukkan ke SBox(5), hasilnya = 2 = 0010
Kelompok 6, 1111 = 15, masukkan ke SBox(6), hasilnya = 12 = 1100
Kelompok 7, 1011 = 11, masukkan ke SBox(7), hasilnya = 14 = 1110
4) Hasil mutasi SBox digabungkan kembali menjadi 32 bit dan

lakukan Rotate Left sebanyak 11 kali.
Hasil = RotateLeft (E0B652CE, 11)
Hasil = B2967705
R(1) = B2967705
5) R(2) = R(1) XOR L(1)
R(2) = B2967705 XOR 824A82D2
R(2) = 30DCF5D7
6) L(2) = R(1) sebelum proses
L(2) = 78F641F7

PUTARAN-2

1) L(2) = 78F641F7
R(2) = 30DCF5D7
2) Lakukan penambahan R(2) dengan K(2) dan hasilnya dimodulus 2^32
untuk menjaga panjang bit hasil penambahan tetap 32 bit.
Hasil = (R(2) + K(2)) mod 2^32
Hasil = (30DCF5D7 + 04040404) mod 2^32
Hasil = 34E0F9DB
3) Pecah hasil menjadi 8 kelompok biner, masing-masing 4 bit dan masukkan ke SBox.
Kelompok 0, 0011 = 3, masukkan ke SBox(0), hasilnya = 2 = 0010
Kelompok 1, 0100 = 4, masukkan ke SBox(1), hasilnya = 6 = 0110
Kelompok 2, 1110 = 14, masukkan ke SBox(2), hasilnya = 9 = 1001
Kelompok 3, 0000 = 0, masukkan ke SBox(3), hasilnya = 7 = 0111
Kelompok 4, 1111 = 15, masukkan ke SBox(4), hasilnya = 2 = 0010
Kelompok 5, 1001 = 9, masukkan ke SBox(5), hasilnya = 6 = 0110
Kelompok 6, 1101 = 13, masukkan ke SBox(6), hasilnya = 8 = 1000
Kelompok 7, 1011 = 11, masukkan ke SBox(7), hasilnya = 14 = 1110
4) Hasil mutasi SBox digabungkan kembali menjadi 32 bit dan
lakukan Rotate Left sebanyak 11 kali.
Hasil = RotateLeft (2697268E, 11)
Hasil = B9347134
R(2) = B9347134
5) R(3) = R(2) XOR L(2)
R(3) = B9347134 XOR 78F641F7
R(3) = C1C230C3
6) L(3) = R(2) sebelum proses
L(3) = 30DCF5D7

PUTARAN-3

1) L(3) = 30DCF5D7
R(3) = C1C230C3
2) Lakukan penambahan R(3) dengan K(3) dan hasilnya dimodulus 2^32
untuk menjaga panjang bit hasil penambahan tetap 32 bit.
Hasil = (R(3) + K(3)) mod 2^32
Hasil = (C1C230C3 + 04040404) mod 2^32
Hasil = C5C634C7
3) Pecah hasil menjadi 8 kelompok biner, masing-masing 4 bit dan masukkan ke SBox.
Kelompok 0, 1100 = 12, masukkan ke SBox(0), hasilnya = 7 = 0111
Kelompok 1, 0101 = 5, masukkan ke SBox(1), hasilnya = 13 = 1101
Kelompok 2, 1100 = 12, masukkan ke SBox(2), hasilnya = 6 = 0110
Kelompok 3, 0110 = 6, masukkan ke SBox(3), hasilnya = 9 = 1001
Kelompok 4, 0011 = 3, masukkan ke SBox(4), hasilnya = 1 = 0001
Kelompok 5, 0100 = 4, masukkan ke SBox(5), hasilnya = 7 = 0111
Kelompok 6, 1100 = 12, masukkan ke SBox(6), hasilnya = 6 = 0110
Kelompok 7, 0111 = 7, masukkan ke SBox(7), hasilnya = 4 = 0100
4) Hasil mutasi SBox digabungkan kembali menjadi 32 bit dan
lakukan Rotate Left sebanyak 11 kali.
Hasil = RotateLeft (7D691764, 11)
Hasil = 48BB23EB
R(3) = 48BB23EB
5) R(4) = R(3) XOR L(3)
R(4) = 48BB23EB XOR 30DCF5D7
R(4) = 7867D63C
6) L(4) = R(3) sebelum proses
L(4) = C1C230C3

Putaran 4, 5, 6 dan seterusnya berlanjut dengan cara yang sama hingga putaran 31.

PUTARAN-31

1) L(31) = BC7D109F
R(31) = D273B905
2) Lakukan penambahan R(31) dengan K(0) dan hasilnya dimodulus 2^32
untuk menjaga panjang bit hasil penambahan tetap 32 bit.
Hasil = (R(31) + K(0)) mod 2^32
Hasil = (D273B905 +) mod 2^32
Hasil = D273B905
3) Pecah hasil menjadi 8 kelompok biner, masing-masing 4 bit dan masukkan ke SBox.
Kelompok 0, 1101 = 13, masukkan ke SBox(0), hasilnya = 15 = 1111
Kelompok 1, 0010 = 2, masukkan ke SBox(1), hasilnya = 4 = 0100

Kelompok 2, 0111 = 7, masukkan ke SBox(2), hasilnya = 2 = 0010
Kelompok 3, 0011 = 3, masukkan ke SBox(3), hasilnya = 1 = 0001
Kelompok 4, 1011 = 11, masukkan ke SBox(4), hasilnya = 14 = 1110
Kelompok 5, 1001 = 9, masukkan ke SBox(5), hasilnya = 6 = 0110
Kelompok 6, 0000 = 0, masukkan ke SBox(6), hasilnya = 13 = 1101
Kelompok 7, 0101 = 5, masukkan ke SBox(7), hasilnya = 7 = 0111

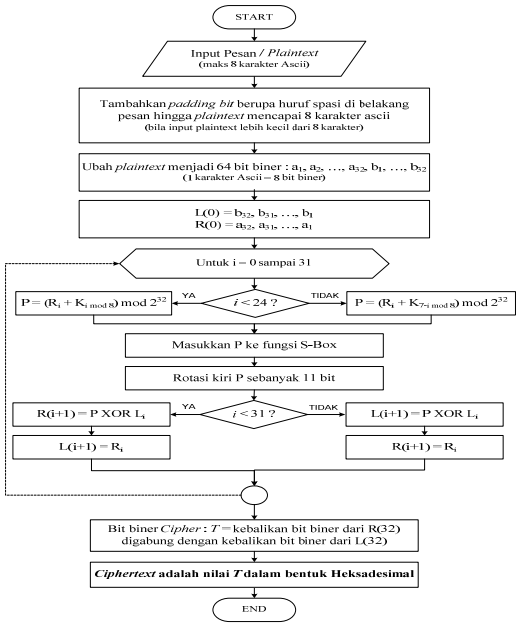
4) Hasil mutasi SBox digabungkan kembali menjadi 32 bit dan
lakukan Rotate Left sebanyak 11 kali.
Hasil = RotateLeft (F421E6D7, 11)
Hasil = 0F36BFA1
R(31) = 0F36BFA1

5) R(32) = R(31) sebelum proses
R(32) = D273B905 = 11010010011100111011100100000101
Nilai R dibalikkan menjadi: 10100000100111011100111001001011

6) L(32) = R(31) XOR L(31)
L(32) = 0F36BFA1 XOR BC7D109F
L(32) = B34BAF3E = 10110011010010111010111100111110
Nilai L dibalikkan menjadi: 01111100111101011101001011001101

L(32) = b(32), b(31), ..., b(1)
R(32) = a(32), a(31), ..., a(1)
HASIL ENKRIPSI = a(1), a(2), ..., a(32), b(1), b(2), ..., b(32)
HASIL ENKRIPSI = Hasil Pembalikkan Nilai R(32) & Hasil Pembalikkan Nilai L(32)
HASIL ENKRIPSI = 01000001001110111001110010010111011111001111010111010010011001101
HASIL ENKRIPSI = A09DCE4B7CF5D2CD (dalam heksa)

Hasil enkripsi pesan "KARAKTER" dalam bilangan heksadesimal adalah A09DCE4B7CF5D2CD. Proses enkripsi metode GOST dapat digambarkan dalam bentuk *flowchart*, seperti terlihat pada gambar III.2.



Gambar III.2 Flowchart Proses Enkripsi Metode GOST

III.1.2.2 Proses Dekripsi

Proses dekripsi sama dengan proses enkripsi, yang berbeda adalah penggunaan sub kunci. Proses dekripsi juga melalui 32 putaran yang memproses 64 bit biner *cipher* (atau 16 bilangan heksa, 1 heksa = 4 bit). Sebagai contoh, berikut adalah proses dekripsi terhadap *ciphertext* pada subbab proses enkripsi sebelumnya.

PUTARAN-0

1) Cipher Text : 'A09DCE4B7CF5D2CD'
Konversi ke biner : 01000001001110111001110010010111011111001111010111010010011001101
Bagi biner menjadi 2 bagian, yaitu: a(1), a(2), ..., a(32) dan b(1), b(2), ..., b(32)
L(0) = b(32), b(31), ..., b(1)
L(0) = 10110011010010111010111100111110 = B34BAF3E

R(0) = a(32), a(31), ..., a(1)
R(0) = 11010010011100111011100100000101 = D273B905

2) Lakukan penambahan R(0) dengan K(0) dan hasilnya dimodulus 2^32
untuk menjaga panjang bit hasil penambahan tetap 32 bit.
Hasil = (R(0) + K(0)) mod 2^32
Hasil = (D273B905 +) mod 2^32
Hasil = D273B905

3) Pecah hasil menjadi 8 kelompok biner, masing-masing 4 bit dan masukkan ke SBox.
Kelompok 0, 1101 = 13, masukkan ke SBox(0), hasilnya = 15 = 1111
Kelompok 1, 0010 = 2, masukkan ke SBox(1), hasilnya = 4 = 0100
Kelompok 2, 0111 = 7, masukkan ke SBox(2), hasilnya = 2 = 0010
Kelompok 3, 0011 = 3, masukkan ke SBox(3), hasilnya = 1 = 0001
Kelompok 4, 1011 = 11, masukkan ke SBox(4), hasilnya = 14 = 1110
Kelompok 5, 1001 = 9, masukkan ke SBox(5), hasilnya = 6 = 0110
Kelompok 6, 0000 = 0, masukkan ke SBox(6), hasilnya = 13 = 1101
Kelompok 7, 0101 = 5, masukkan ke SBox(7), hasilnya = 7 = 0111
4) Hasil mutasi SBox digabungkan kembali menjadi 32 bit dan
lakukan Rotate Left sebanyak 11 kali.
Hasil = RotateLeft (F421E6D7, 11)
Hasil = 0F36BFA1
R(0) = 0F36BFA1
5) R(1) = R(0) XOR L(0)
R(1) = 0F36BFA1 XOR B34BAF3E
R(1) = BC7D109F
6) L(1) = R(0) sebelum proses
L(1) = D273B905

PUTARAN-1

1) L(1) = D273B905
R(1) = BC7D109F
2) Lakukan penambahan R(1) dengan K(1) dan hasilnya dimodulus 2^32
untuk menjaga panjang bit hasil penambahan tetap 32 bit.
Hasil = (R(1) + K(1)) mod 2^32
Hasil = (BC7D109F + 04040404) mod 2^32
Hasil = C08114A3
3) Pecah hasil menjadi 8 kelompok biner, masing-masing 4 bit dan masukkan ke SBox.
Kelompok 0, 1100 = 12, masukkan ke SBox(0), hasilnya = 7 = 0111
Kelompok 1, 0000 = 0, masukkan ke SBox(1), hasilnya = 14 = 1110
Kelompok 2, 1000 = 8, masukkan ke SBox(2), hasilnya = 14 = 1110
Kelompok 3, 0001 = 1, masukkan ke SBox(3), hasilnya = 13 = 1101
Kelompok 4, 0001 = 1, masukkan ke SBox(4), hasilnya = 12 = 1100
Kelompok 5, 0100 = 4, masukkan ke SBox(5), hasilnya = 7 = 0111
Kelompok 6, 1010 = 10, masukkan ke SBox(6), hasilnya = 14 = 1110
Kelompok 7, 0011 = 3, masukkan ke SBox(7), hasilnya = 0 = 0000
4) Hasil mutasi SBox digabungkan kembali menjadi 32 bit dan
lakukan Rotate Left sebanyak 11 kali.
Hasil = RotateLeft (7EEDC7E0, 11)
Hasil = 6E3F03F7
R(1) = 6E3F03F7
5) R(2) = R(1) XOR L(1)
R(2) = 6E3F03F7 XOR D273B905
R(2) = BC4CBAF2
6) L(2) = R(1) sebelum proses
L(2) = BC7D109F

PUTARAN-2

1) L(2) = BC7D109F
R(2) = BC4CBAF2
2) Lakukan penambahan R(2) dengan K(2) dan hasilnya dimodulus 2^32
untuk menjaga panjang bit hasil penambahan tetap 32 bit.
Hasil = (R(2) + K(2)) mod 2^32
Hasil = (BC4CBAF2 + 04040404) mod 2^32
Hasil = C050BEF6
3) Pecah hasil menjadi 8 kelompok biner, masing-masing 4 bit dan masukkan ke SBox.
Kelompok 0, 1100 = 12, masukkan ke SBox(0), hasilnya = 7 = 0111
Kelompok 1, 0000 = 0, masukkan ke SBox(1), hasilnya = 14 = 1110
Kelompok 2, 0101 = 5, masukkan ke SBox(2), hasilnya = 3 = 0011
Kelompok 3, 0000 = 0, masukkan ke SBox(3), hasilnya = 7 = 0111
Kelompok 4, 1011 = 11, masukkan ke SBox(4), hasilnya = 14 = 1110
Kelompok 5, 1110 = 14, masukkan ke SBox(5), hasilnya = 15 = 1111
Kelompok 6, 1111 = 15, masukkan ke SBox(6), hasilnya = 12 = 1100
Kelompok 7, 0110 = 6, masukkan ke SBox(7), hasilnya = 10 = 1010
4) Hasil mutasi SBox digabungkan kembali menjadi 32 bit dan
lakukan Rotate Left sebanyak 11 kali.
Hasil = RotateLeft (7E37EFCA, 11)
Hasil = BF7E53F1
R(2) = BF7E53F1
5) R(3) = R(2) XOR L(2)
R(3) = BF7E53F1 XOR BC7D109F
R(3) = 0303436E
6) L(3) = R(2) sebelum proses

L(3) = BC4CBAF2

PUTARAN-3

1) L(3) = BC4CBAF2
R(3) = 0303436E

2) Lakukan penambahan R(3) dengan K(3) dan hasilnya dimodulus 2^32
untuk menjaga panjang bit hasil penambahan tetap 32 bit.
Hasil = (R(3) + K(3)) mod 2^32
Hasil = (0303436E + 04040404) mod 2^32
Hasil = 07074772

3) Pecah hasil menjadi 8 kelompok biner, masing-masing 4 bit dan masukkan ke SBox.
Kelompok 0, 0000 = 0, masukkan ke SBox(0), hasilnya = 4 = 0100
Kelompok 1, 0111 = 7, masukkan ke SBox(1), hasilnya = 10 = 1010
Kelompok 2, 0000 = 0, masukkan ke SBox(2), hasilnya = 5 = 0101
Kelompok 3, 0111 = 7, masukkan ke SBox(3), hasilnya = 15 = 1111
Kelompok 4, 0100 = 4, masukkan ke SBox(4), hasilnya = 5 = 0101
Kelompok 5, 0111 = 7, masukkan ke SBox(5), hasilnya = 13 = 1101
Kelompok 6, 0111 = 7, masukkan ke SBox(6), hasilnya = 9 = 1001
Kelompok 7, 0010 = 2, masukkan ke SBox(7), hasilnya = 13 = 1101

4) Hasil mutasi SBox digabungkan kembali menjadi 32 bit dan
lakukan Rotate Left sebanyak 11 kali.
Hasil = RotateLeft (4A5F5D9D, 11)
Hasil = FAECEA52
R(3) = FAECEA52

5) R(4) = R(3) XOR L(3)
R(4) = FAECEA52 XOR BC4CBAF2
R(4) = 46A050A0

6) L(4) = R(3) sebelum proses
L(4) = 0303436E

Putaran 4, 5, 6 dan seterusnya berlanjut dengan cara yang sama hingga putaran ke-31.

PUTARAN-31

1) L(31) = 78F641F7
R(31) = 824A82D2

2) Lakukan penambahan R(31) dengan K(0) dan hasilnya dimodulus 2^32
untuk menjaga panjang bit hasil penambahan tetap 32 bit.
Hasil = (R(31) + K(0)) mod 2^32
Hasil = (824A82D2 +) mod 2^32
Hasil = 824A82D2

3) Pecah hasil menjadi 8 kelompok biner, masing-masing 4 bit dan masukkan ke SBox.
Kelompok 0, 1000 = 8, masukkan ke SBox(0), hasilnya = 6 = 0110
Kelompok 1, 0010 = 2, masukkan ke SBox(1), hasilnya = 4 = 0100
Kelompok 2, 0100 = 4, masukkan ke SBox(2), hasilnya = 10 = 1010
Kelompok 3, 1010 = 10, masukkan ke SBox(3), hasilnya = 6 = 0110
Kelompok 4, 1000 = 8, masukkan ke SBox(4), hasilnya = 4 = 0100
Kelompok 5, 0010 = 2, masukkan ke SBox(5), hasilnya = 10 = 1010
Kelompok 6, 1101 = 13, masukkan ke SBox(6), hasilnya = 8 = 1000
Kelompok 7, 0010 = 2, masukkan ke SBox(7), hasilnya = 13 = 1101

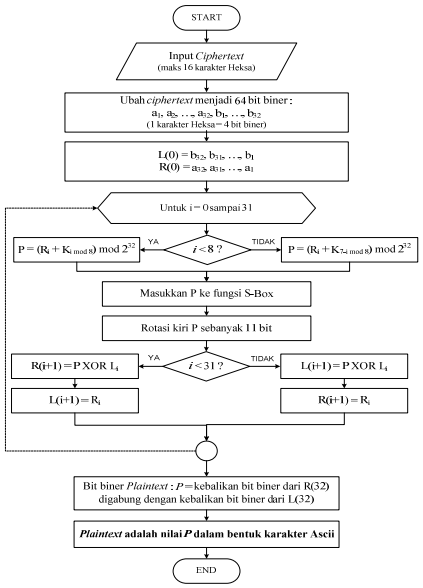
4) Hasil mutasi SBox digabungkan kembali menjadi 32 bit dan
lakukan Rotate Left sebanyak 11 kali.
Hasil = RotateLeft (64A64A8D, 11)
Hasil = 32546B25
R(31) = 32546B25

5) R(32) = R(31) sebelum proses
R(32) = 824A82D2 = 10000010010010101000001011010010
Nilai R dibalikkan menjadi: 01001011010000010101001001000001

6) L(32) = R(31) XOR L(31)
L(32) = 32546B25 XOR 78F641F7
L(32) = 4AA22AD2 = 01001010101000100010101011010010
Nilai L dibalikkan menjadi: 01001011010101000100010101010010

L(32) = b(32), b(31), ..., b(1)
R(32) = a(32), a(31), ..., a(1)
HASIL DEKRIPSI = a(1), a(2), ..., a(32), b(1), b(2), ..., b(32)
HASIL DEKRIPSI = Hasil Pembalikkan Nilai R(32) & Hasil Pembalikkan Nilai L(32)
HASIL DEKRIPSI = 01001011010000010101001001000001010010110101010001000101010010
HASIL DEKRIPSI = 4B4152414B544552 (dalam heksa) = KARAKTER (ascii)

Hasil dekripsi *ciphertext* "D409BE7682E6F087" dalam karakter *Ascii* adalah "KARAKTER". Proses dekripsi metode GOST dapat digambarkan dalam bentuk *flowchart*, seperti terlihat pada gambar III.3.



Gambar III.3 Flowchart Proses Deskripsi Metode GOST

III.1.2 Sistem Ujian Online

Pada komputer *server*, terdapat beberapa data yang harus diatur seperti data Mata Pelajaran, data Guru, data Siswa, data administrator, Pertanyaan dan jawaban, data tanggal validasi, data data naik kelas Siswa dan daftar Mata Pelajaran yang diadopsi guru. Selain itu, proses ujian dimulai dari komputer *server*. Pada komputer *client*, *user* (murid) dapat memulai hanya dengan melakukan registrasi data *user* ke sistem untuk mengikuti proses ujian. Setelah proses ujian dimulai, murid dapat menjalankan ujian.

IV. ALGORITMA DAN IMPLEMENTASI

IV.1 Algoritma

Beberapa algoritma yang akan digunakan untuk membangun aplikasi ujian *online* ini adalah algoritma koneksi, algoritma pengiriman dan penerimaan pesan, serta algoritma kriptografi GOST.

IV.1.1 Algoritma Proses Koneksi

Proses koneksi di dalam aplikasi menggunakan *windows socket* dari aplikasi VB.Net, yaitu dengan memanggil *syntax* "Imports.System.Net.Sockets". Algoritma proses koneksi *client* ke *server* adalah sebagai berikut:

1. Set Client = class baru dari TcpClient
2. Koneksi ke *IP Address Server* dengan memanggil syntax berikut:
   "client.Connect(IPAddress.Parse(txtAddress.Text), 8888)"
3. Set reader = New BinaryReader(client.GetStream())
4. Set writer = New BinaryWriter(client.GetStream())
5. clientThread = New System.Threading.Thread(AddressOf readSocket)
6. Mulai koneksi dengan memanggil "clientThread.Start()"

IV.1.2 Algoritma Kriptografi GOST

IV.1.2.1 Proses Pembentukan Kunci

Ukuran kunci GOST adalah 256 bit kunci atau 32 karakter Ascii. Algoritma proses pembentukan kunci GOST adalah sebagai berikut:

1. Tambahkan karakter spasi pada kunci sehingga panjang kunci mencapai 32 karakter Ascii.
2. Ubah semua karakter kunci ke bentuk biner, dengan panjang total mencapai 256 bit biner.
3. Pecah bit kunci menjadi:
   a. K0 = (k_{32}, …, k_1)
   b. K1 = (k_{64}, …, k_{33})
   c. K2 = (k_{96}, …, k_{65})
   d. K3 = (k_{128}, …, k_{97})
   e. K4 = (k_{160}, …, k_{129})
   f. K5 = (k_{192}, …, k_{161})
   g. K6 = (k_{224}, …, k_{193})
   h. K7 = (k_{256}, …, k_{225})

Nilai K0 hingga K7 akan digunakan pada proses pembentukan kunci.

IV.1.2.2 Proses Enkripsi

Proses ini berfungsi untuk melakukan proses enkripsi terhadap pertanyaan dan jawaban ujian di dalam *database*.

Algoritma proses enkripsi pesan dengan metode GOST adalah sebagai berikut:

1. Tambahkan karakter spasi pada pesan sehingga panjang pesan merupakan kelipatan 8.
2. Untuk nI = 1 sampai panjang pesan, dengan penambahan nI pada setiap *looping* adalah 8, maka lakukan proses berikut:
   i. HasilProses = fungsi EnkripsiGost dari pesan, dimulai pada posisi nI sepanjang 8 karakter.
   ii. Hasil = Hasil & HasilProses
3. Hasil = hasil enkripsi pesan dengan menggunakan metode GOST.

Algoritma fungsi proses enkripsi pesan per 1 blok (8 karakter) dengan metode GOST adalah sebagai berikut:

1. cText1 = hasil konversi biner dari pesan, sepanjang 64 bit biner.

2. R(0) = nilai biner cText1 dimulai dari posisi bit-32, bit-31 hingga posisi bit-1.
3. L(0) = nilai biner cText1 dimulai dari posisi bit-64, bit-63 hingga posisi bit-33.
4. Untuk I = 0 sampai 31,
    a. nTemp1 = hasil konversi R(I) ke desimal.
    b. Jika I < 24, maka:
        i. Z = I Mod 8
        ii. nTemp2 = konversi pKunciGOST(Z) ke desimal.
    c. Jika tidak, maka:
        i. Z = 7 – (I Mod 8)
        ii. nTemp2 = konversi pKunciGOST(Z) ke desimal.
    d. nTemp = FMod(nTemp1 + nTemp2, 2 ^ 32)
    e. cTemp = hasil konversi nTemp ke biner, sepanjang 32 bit.
    f. CM(I) = cTemp
    g. Masukkan setiap blok 4 bit dari cTemp ke Sbox.
    h. S(I) = penggabungan semua *output* Sbox.
    i. cTemp = rotasi kiri dari S(I) sebanyak 11 bit.
    j. RSL(I) = cTemp
    k. Jika I tidak sama dengan 31, maka:
        i. R(I + 1) = FXORBiner(cTemp,L(I), 32)
        ii. L(I + 1) = R(I)
    l. Jika tidak, maka:
        i. R(I + 1) = R(I)
        ii. L(I + 1) = FXORBiner(cTemp,L(I), 32)
5. Hasil enkripsi = pembalikan dari bit R(32) digabung dengan pembalikan dari bit L(32).

IV.1.3.3 Proses Dekripsi

Algoritma proses dekripsi pesan dengan metode GOST adalah sebagai berikut:

1. Untuk nI = 1 sampai panjang pesan, dengan penambahan nI pada setiap *looping* adalah 16, maka lakukan proses berikut:
    i. HasilProses = fungsi DekripsiGost dari pesan, dimulai pada posisi nI sepanjang 16 karakter.
    ii. Hasil = Hasil & HasilProses
2. Hasil = hasil dekripsi pesan dengan menggunakan metode GOST.

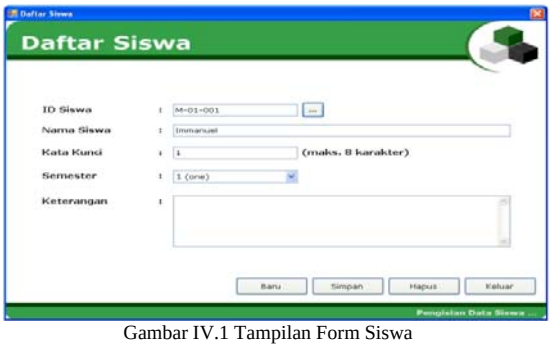
Algoritma fungsi proses dekripsi pesan per 1 blok (8 karakter) dengan metode GOST adalah sebagai berikut:

1. cText1 = hasil konversi biner dari pesan, sepanjang 64 bit biner.
2. R(0) = nilai biner cText1 dimulai dari posisi bit-32, bit-31 hingga posisi bit-1.
3. L(0) = nilai biner cText1 dimulai dari posisi bit-64, bit-63 hingga posisi bit-33.
4. Untuk I = 0 sampai 31,
    a. nTemp1 = hasil konversi R(I) ke desimal.
    b. Jika I < 8, maka:
        i. Z = I Mod 8
        ii. nTemp2 = konversi pKunciGOST(Z) ke desimal.
    c. Jika tidak, maka:
        i. Z = 7 – (I Mod 8)
        ii. nTemp2 = konversi pKunciGOST(Z) ke desimal.
    d. nTemp = FMod(nTemp1 + nTemp2, 2 ^ 32)
    e. cTemp = hasil konversi nTemp ke biner, sepanjang 32 bit.
    f. CM(I) = cTemp
    g. Masukkan setiap blok 4 bit dari cTemp ke Sbox.
    h. S(I) = penggabungan semua *output* Sbox.
    i. cTemp = rotasi kiri dari S(I) sebanyak 11 bit.
    j. RSL(I) = cTemp
    k. Jika I tidak sama dengan 31, maka:
        i. R(I + 1) = FXORBiner(cTemp, L(I), 32)
        ii. L(I + 1) = R(I)
    l. Jika tidak, maka:
        i. R(I + 1) = R(I)
        ii. L(I + 1) = FXORBiner(cTemp, L(I), 32)
5. Hasil dekripsi = pembalikan dari bit R(32) digabung dengan pembalikan dari bit L(32).

IV.2 Implementasi

Implementasi sistem yang dibahas mencakup cara penggunaan perangkat lunak. Berikut diberikan contoh hasil eksekusi dari perangkat lunak:

1. Form Siswa

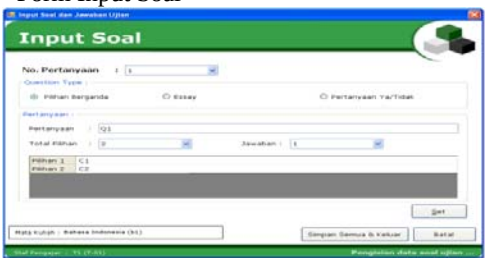


Gambar IV.1 Tampilan Form Siswa

2. Form Staf Pengajar

Gambar IV.2 Tampilan Form Staf Pengajar

3. Form Input Soal



Gambar IV.3 Tampilan Form Input Soal

V. KESIMPULAN DAN SARAN

V.1 Kesimpulan

Setelah menyelesaikan tugas akhir ini, penulis menarik beberapa kesimpulan sebagai berikut:

1. Sistem dapat diterapkan untuk mengadakan proses ujian pada sebuah jaringan komputer dengan melakukan instalasi aplikasi pada komputer *server* dan *client*.
2. Metode GOST dapat digunakan untuk mengamankan pertanyaan-pertanyaan ujian, bila *database* didapatkan oleh pihak yang lain.
3. Laporan yang dihasilkan terdiri dari laporan daftar nilai siswa dan laporan daftar siswa**.**

V.2 Saran

1. Perlu dilakukan pelatihan terhadap siswa dan staf pengajar yang akan menggunakan perangkat lunak sehingga siswa dan staf pengajar mampu mengakses dan menggunakan semua fasilitas yang disediakan oleh perangkat lunak.
2. Perangkat lunak perlu ditambahkan sebuah fasilitas *back up database*, untuk mencegah kehilangan data.

DAFTAR PUSTAKA

Ariyus, D., **Pengantar Ilmu Kriptografi (Teori, Analisis dan Implementasi)**, Andi, Yogyakarta, 2008.

Saarinen, Markku-Juhani (1998). *A chosen key attack against the secret S-boxes of GOST*

Igbal, M., The Understanding of GOST Cryptograpy Technique, International Journal of Engineering Trends and Technolgy, 2016

Kenneth, E.K., dan Julie, E.K, **Analisis dan Perancangan Sistem**, Alih bahasa oleh Thamir Abdul Hafedh Al-Hamdany, Jilid 1 dan Jilid 2, Edisi ke-5, PT Prenhallindo, Jakarta, 2003.

Kurniawan, Y., **Kriptografi, Keamanan Internet dan Jaringan Komunikasi**, Informatika, Bandung, 2004.

Mackenzie & Sharkey, **Visual Basic.Net (Belajar sendiri dalam 21 hari)**, Andi, Yogyakarta, 2005

Munir, R., **Matematika Diskrit**, Informatika, Bandung, edisi Ketiga 2005.

Munir, R., **Kriptografi**, Informatika, Bandung, 2006.