# BAB II
# DASAR TEORI

## 2.1 KAJIAN PUSTAKA

Penelitian Meko [7] pada tahun 2018, membandingkan algoritma DES, AES, IDEA, dan Blowfish dalam enkripsi dan dekripsi data. Tujuan dari penelitian ini adalah untuk membandingkan kinerja dari beberapa algoritma kriptografi pada proses enkripsi dan dekripsi data berdasarkan segi kecepatan, serta ukuran *file* hasil enkripsi. Pengujian dilakukan menggunakan beberapa tipe *file* dengan berbagai macam ukuran. Hasil yang diperoleh menunjukkan algoritma AES memiliki kecepatan paling tinggi dalam proses enkripsi maupun dekripsi data kemudian diikutin oleh Blowfish, DES, kemudian IDEA.

Penelitian Maata, dkk [9] pada tahun 2017, membahas pengembangan aplikasi berbasis *client server* menggunakan *socket programming* dalam lingkungan terdistribusi. Tujuan dari penelitian ini adalah mendemonstrasikan prinsip serta konsep dari *socket programming* yang tersedia di *library* Bahasa pemrograman *java*. Pengujian dilakukan dengan merancang dan mengimplementasikan aplikasi berbasis *server client* yang berjalan pada "Layanan Penagihan OpTel (OBS)" atau disebut juga lingkungan komputasi terdistribusi. Hasil yang diperoleh menunjukan bahwa gaya pemrograman, konsep, fungsi, data diagram, dan *socket programming* mudah dilakukan dalam Bahasa pemrograman Java serta aplikasi yang dirancang mampu mensimulasikan *scenario* dan mampu menggambarkan penggunaan *socket programming* dan cara kerjanya dilingkungan dunia nyata.

Penelitian Sardana, dan Future [10] pada tahun 2021, membahas mengenai pertimbangan *Instrusion Detection System* (IDS) untuk membuat deteksi dan klasifikasi intrusi, serangan, serta berbagai jenis aktivitas pencurian data. Tujuan dari penelitian ini untuk meningkatkan *system* deteksi instruksi yang aman dengan *user-defined socket* dan *random forest classifier*. Pengujian dilakukan dengan dua fase, yaitu pengukuran, dan pengelompokan. Semua metode digabungkan untuk mendukung penyerangan. Peyerangan pertama untuk pengidentifikasian data biasa ditetapkan sebagai kelas satu, kemudian serangan sisanya ditetapkan sebagai kelas

dua. Hasil yang diperoleh mumungkinkan potensi serangan IDS berkurang karena adanya enkripsi dan penerapan *socket programming*. Akurasi, nilai kepresisian, nilai *recall*, dan f-*score* lebih baik pada *system* IDS yang diusulkan.

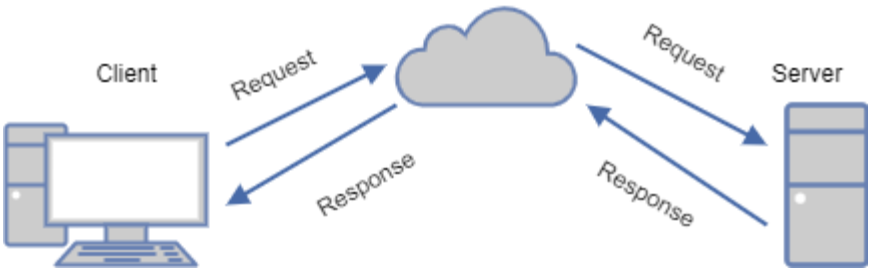
Penelitian oleh Suranta, dan Sakti [11] pada tahun 2022, membahas mengenai penerapan Algoritma AES 128 untuk enkripsi dokumen di PT. Gunung Geulis Elok Abadi. Tujuan dari penelitian merancang suatu aplikasi yang dapat mengamankan data agar tidak dapat digunakan oleh orang yang tidak memiliki kepentingan. Pengujian dilakukan dengan metode *blackbox* untuk menguji jalannya aplikasi, setelahnya dilakukan ujicoba enkripsi dan dekripsi menggunakan 20 *file* untuk mengukur kecepatan proses enkripsi dan dekripsi, serta perubahan ukuran *file* hasil enkripsi dan dekripsi. Hasil yang diperoleh dari pengujian menunjukkan bahwa aplikasi enkripsi-dekripsi file menggunakan AES-128 dapat berjalan dengan baik serta dapat melindungi dokumen dari pihak yang tidak bertanggung jawab. Terdapat kenaikan ukuran file sebesar 0.038% - 2.086% pada ujicoba enkripsi dengan 20 file, dan waktu rata-rata sebesar 12.769 *milisecond* pada ujicoba enkripsi dan rata-rata dekripsi sebesar 18.075 *milisecond*.

Penelitian oleh Irawati, dan Rachmawati [12] pada tahun 2018, membahas mengenai teknik penganaman data menggunakan algoritma IDEA (*International Data Encryption Algorithm*) yang kemudian dikombinasikan dengan metode steganografi menggunakan metode *End of File.* Pengujian dilakukan pada 8 kata *plain text* yang memiliki Panjang karakter bervariasi kemudian dilakukan enkripsi dan dilakukan penyisipan pada dua jenis file yang berbeda format plain image yaitu format warna atau RGB dan *Greyscale*. Hasil yang diperoleh setelah dilakukan enkripsi dan penyisipan pada plain image menunjukkan perbedaan besar ukuran dan dimensi dari plain image (*stego image*). Sementara setelah dilakukan dekripsi dan ekstraksi pada *stego image* Panjang karakter dari *plain text* dan *plain image* Kembali seperti semula.

## 2.2 DASAR TEORI

### 2.2.1 *CLIENT SERVER*

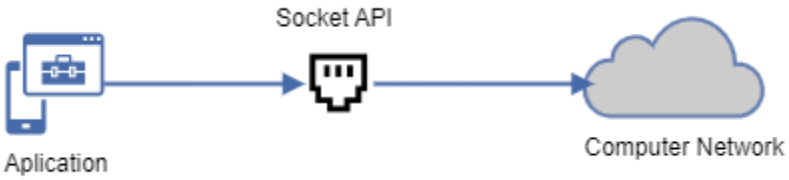
*Client server* merupakan jaringan yang terdiri dari komputer sebagai *client* atau sebagai *server*. *Server* merupakan program yang menawarkan layanan sedangkan *client* merupakan program yang meminta layanan. *Client* biasanya merujuk ke PC atau *workstation*, yang menyediakan terminal *client* dengan antar muka yang ramah [13]. Komunikasi antara *client* dan *server* dimulai dengan *client* mengirimkan *request* tehadap layanan yang dimintas, kemudian *server* akan merespon *request* dari client. Komunikasi *client server* ditunjukkan pada Gambar 2.1.




Gambar 2. 1 Komunikasi *Client Server*

### 2.2.1 *SOCKET*

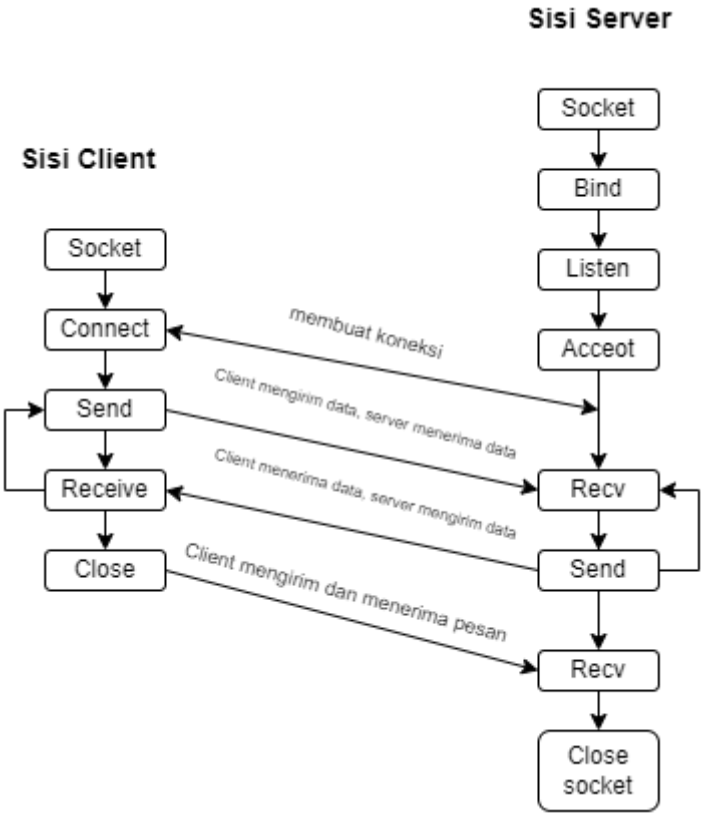
*Socket* berasal dari metafora *socket* listris atau telepon, hal ini dikarenakan *socket* bertindak sebagai antarmuka yang terhubung satu sama lain. Socket dapat didefinisikan sebagai titik akhir dari koneksi antar dua computer yang diidentifikasi sebagai alamat IP dan Port. Selain itu juga didefinisikan sebagai abstraksi dari perangkat lunak yang digunakan untuk mewakili terminal dari koneksi antar dua mesin [13].




Gambar 2. 2 Contoh Dari *Socket*

### 2.2.1 *SOCKET PROGRAMMING*

Soket mewakili koneksi tunggal antara tepat dua buah perangkat lunak. Ini adalah titik koneksi komunikasi (titik akhir) yang dapat Anda beri nama dan alamat dalam jaringan. Soket memungkinkan aplikasi untuk berkomunikasi menggunakan mekanisme standar yang dibangun ke dalam perangkat keras jaringan dan sistem operasi. Soket juga memungkinkan pertukaran informasi antara proses pada mesin yang sama atau melalui jaringan, mendistribusikan pekerjaan ke mesin yang paling efisien, dan memungkinkan akses ke data terpusat dengan mudah. Proses yang menggunakan soket dapat berada pada sistem yang sama atau pada sistem yang berbeda pada jaringan yang berbeda. Soket berguna untuk aplikasi yang berdiri sendiri maupun jaringan. Standar jaringan untuk TCP/IP Soket disediakan oleh antarmuka program aplikasi (API). Berbagai macam sistem operasi mendukung API soket. Pemrograman soket menunjukkan cara menggunakan API soket untuk membangun hubungan komunikasi antara proses jarak jauh dan lokal. Soket OS/400 mendukung beberapa protokol transportasi dan jaringan. Juga fungsi sistem soket dan fungsi jaringan soket aman untuk ulir. Programmer yang menggunakan *Integrated Language Environment* (ILE) C dapat menggunakan informasi tersebut untuk mengembangkan aplikasi soket [14].




Gambar 2. 3 Skema *Socket Programming*

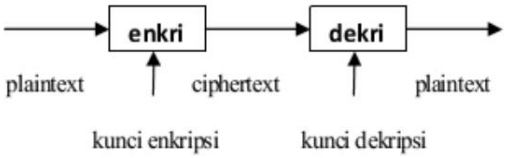
*Socket* merupakan sejenis struktur data abstrak yang disediakan oleh sistem operasi, digunakan untuk membangun akses tanpa adanya korelasi antara proses pengiriman dan penerimaan pesan. Berikut merupakan Langkah-langkah penghubungan *socket server* dengan *socket client* :

1. *socket*() pada sisi *client* dan *server* membuat soket baru menggunakan jenis soket tertentu, yang diidentifikasi dengan bilangan bulat.
2. *bind*() fungsi ini biasanya digunakan di sisi *server*, dan mengaitkan soket dengan ketentuan nomor *port* lokal dan alamat IP.
3. *listen*() fungsi ini digunakan di sisi *server*, sehingga socket TCP yang digunakan akan tertaut, sehingga akan memasuki mode *listen* (menunggu *client* terhubung).
4. *connect*() fungsi ini digunakan di sisi *client*, dan memberikan nomor *port* ke soket. Dalam kasus soket TCP, hal ini yang membuat sambungan TCP baru.

5. *accept*() fungsi ini digunakan pada sisi *server*. Hal ini digunakan untuk menerima koneksi TCP serta membuat socket baru untuk membuat koneksi dari *client.*
6. *send*(), *recv*(), *write*(), *read*(), *sendto*() dan *recvfrom*(), digunakan untuk mengirim dan menerima data ke/dari *socket* yang terkoneksi.
7. *close*() fungsi ini digunakan untuk mengakhiri koneksi dari socket.

### 2.2.2 KRIPTOGRAFI

Kriptografi terdiri dari dua kata dalam Bahasa Yunani yaitu *cryptos* yang memiliki arti rahasia dan *graphein* yang memiliki arti tulisan, secara harfiah kriptografi dapat dijelaskan sebagai tulisan rahasia. Kriptografi adalah ilmu yang mempelajari teknik matematika yang berkaitan dengan aspek keamanan informasi, seperti kerahasiaan data, otentikasi, integritas data, dan validitas. Enkripsi juga dapat diartikan sebagai ilmu yang menjaga kerahasiaan pesan. Kedua, proses yang dijelaskan dalam penelitian ini melibatkan dua proses dasar enkripsi: enkripsi dan dekripsi. Enkripsi adalah proses mengubah data asli (*plaintext*) menjadi pesan yang tidak dapat dibaca (*cipherteks*), dan dekripsi adalah proses mengubah data yang dimanipulasi menjadi data asli. Gambar 2.4 adalah diagram sederhana dari proses enkripsi.




Gambar 2. 4 Diagram Proses Enkripsi dan Dekripsi

Sebagai aturan, enkripsi memiliki empat komponen utama: *plainttext*, atau pesan yang dapat dibaca, pesan rahasia, ini merupakan pesan acak yang tidak dapat dibaca, dan kunci yang merupakan kunci untuk mengeksekusi teknik kriptografi [12].

### 2.2.3 *ADVANCED ENCRYPTION STANDART* (AES)

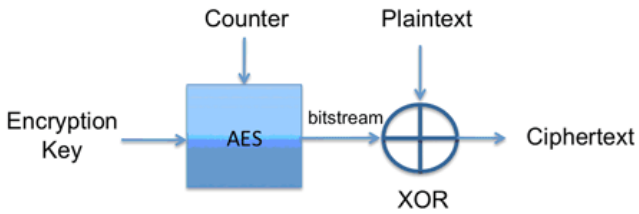
Kontes untuk memilih standar algoritma kriptografi baru untuk menggantikan DES dimulai pada tahun 1997, dengan 21 pesaing ikut serta. Setelah

proses seleksi yang ketat, hanya tersisa lima kandidat pada tahun 1999, yaitu algoritma algoritma *Serpent* (Ross Anderson-*University of Cambridge*, Eli Biham-Technion, Lars Knudsen-*University of California San Diego*), MARS (IBM Amerika), Twofish (Bruce Schneier, John Kelsey, dan Niels Ferguson-*Counterpane Internet Security* Inc, Doug Whiting-Hi/fn Inc, David Wagner-*University of California* Berkeley, Chris Hall-Princeton *University*), Rijndael (Dr. Vincent Rijmen-Katholieke *Universiteit Leuvendan* Dr. Joan Daemen-*Proton World International*), dan RC6 (RSA Amerika). Algoritma Rijndael dinobatkan sebagai AES setahun kemudian pada tahun 2000, sebagai teknik kriptografi yang aman dan efisien dalam penerapannya. Nama Rijndael merupakan gabungan dari nama kedua penemunya [15].

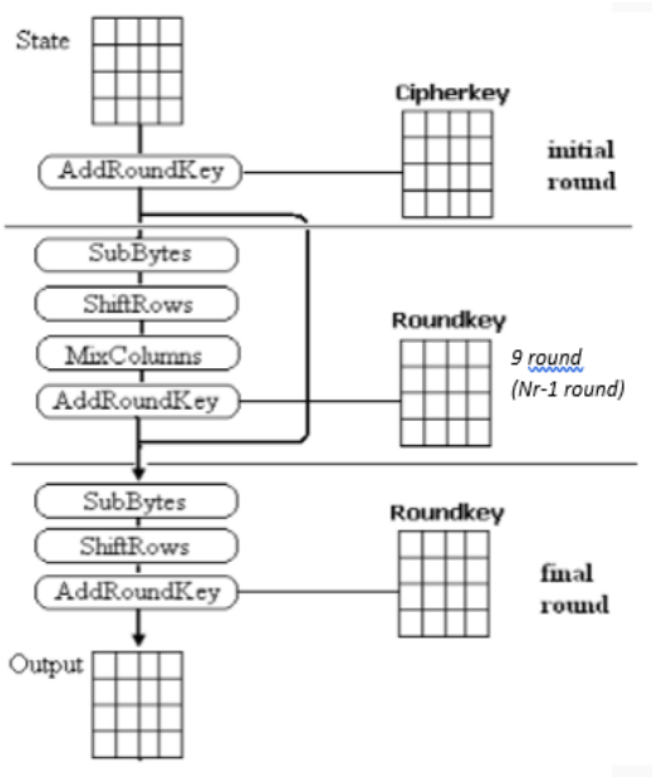
Algoritma AES menggunakan kunci yang sama dalam proses enkripsi maupun dekripsi, sehingga termasuk dalam algoritma simetri [16]. AES ini merupakan algoritma *block cipher* dengan menggunakan sistem permutasi dan substitusi (P- Box dan S-Box) [17]. Selain itu algoritma AES merupakan algoritma *chipper* yang aman untuk melindungi data atau informasi yang bersifat rahasia. Panjang kunci dari AES terdiri dari 128 bit, 192 bit, dan 256 bit. Perbedaan panjang kunci ini yang nantinya mempengaruhi jumlah putaran pada algoritma AES, untuk lebih jelasnya ditunjukkan pada Tabel 2.1.

Tabel 2. 1 Tabel Data Urutan Algoritma AES [18]

| Tipe Algoritma | Panjang Kunci | Panjang Blok | Jumlah Putaran |
|---|---|---|---|
| AES-128 | 4 | 4 | 10 |
| AES-192 | 6 | 4 | 12 |
| AES-256 | 8 | 4 | 14 |




Gambar 2. 5 Proses Kriptografi AES (*Advance Encryption Standard)*

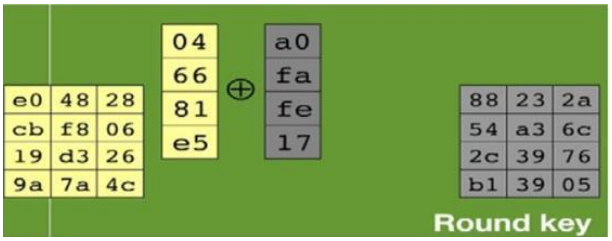



Gambar 2. 6 Alur Enkripsi AES-256

Berikut merupakan tahapan proses algoritma AES seperti yang telah diilustrasikan pada Gambar 2.6:

a. *Add Round Key*

*Add Round Key* merupakan sebuah *ciphertext* yang dikombinasikan dalam perhitungan dengan XOR.




Gambar 2. 7 *Add Round Key*

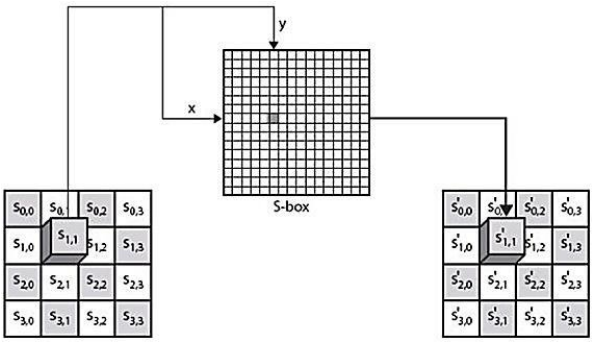
Dalam gambar tersebut terlihat tabel yang berada kiri merupakan perhitungan *ciphertext* dan yang berada disebelah kanan merupakan hasil dari *roundkey*-nya. XOR didalam yang dilakukan dalam gambar diatas dimana setiap kolom yaitu *ciphertext* pada kolom 1 dilakukan XOR dengan kolom 1 *round key* dan selanjutnya.

b. *Sub Bytes*

Prinsip dasar dalam perhitungan *sub bytes* yaitu mengubah dari isi tabel atau isi matrik lainnya yang disebut dengan S-BOX.

| | x0 | x1 | x2 | x3 | x4 | x5 | x6 | x7 | x8 | x9 | xa | xb | xc | xd | xe | xf |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0x | 63 | 7c | 77 | 7b | f2 | 6b | 6f | c5 | 30 | 01 | 67 | 2b | fe | d7 | ab | 76 |
| 1x | ca | 82 | c9 | 7d | fa | 59 | 47 | f0 | ad | d4 | a2 | af | 9c | a4 | 72 | c0 |
| 2x | b7 | fd | 93 | 26 | 36 | 3f | f7 | cc | 34 | a5 | e5 | f1 | 71 | d8 | 31 | 15 |
| 3x | 04 | c7 | 23 | c3 | 18 | 96 | 05 | 9a | 07 | 12 | 80 | e2 | eb | 27 | b2 | 75 |
| 4x | 09 | 83 | 2c | 1a | 1b | 6e | 5a | a0 | 52 | 3b | d6 | b3 | 29 | e3 | 2f | 84 |
| 5x | 53 | d1 | 00 | ed | 20 | fc | b1 | 5b | 6a | cb | be | 39 | 4a | 4c | 58 | cf |
| 6x | d0 | ef | aa | fb | 43 | 4d | 33 | 85 | 45 | f9 | 02 | 7f | 50 | 3c | 9f | a8 |
| 7x | 51 | a3 | 40 | 8f | 92 | 9d | 38 | f5 | bc | b6 | da | 21 | 10 | ff | f3 | d2 |
| 8x | cd | 0c | 13 | ec | 5f | 97 | 44 | 17 | c4 | a7 | 7e | 3d | 64 | 5d | 19 | 73 |
| 9x | 60 | 81 | 4f | dc | 22 | 2a | 90 | 88 | 46 | ee | b8 | 14 | de | 5e | 0b | db |
| ax | e0 | 32 | 3a | 0a | 49 | 06 | 24 | 5c | c2 | d3 | ac | 62 | 91 | 95 | e4 | 79 |
| bx | e7 | c8 | 37 | 6d | 8d | d5 | 4e | a9 | 6c | 56 | f4 | ea | 65 | 7a | ae | 08 |
| cx | ba | 78 | 25 | 2e | 1c | a6 | b4 | c6 | e8 | dd | 74 | 1f | 4b | bd | 8b | 8a |
| dx | 70 | 3e | b5 | 66 | 48 | 03 | f6 | 0e | 61 | 35 | 57 | b9 | 86 | c1 | 1d | 9e |
| ex | e1 | f8 | 98 | 11 | 69 | d9 | 8e | 94 | 9b | 1e | 87 | e9 | ce | 55 | 28 | df |
| fx | 8c | a1 | 89 | 0d | bf | e6 | 42 | 68 | 41 | 99 | 2d | 0f | b0 | 54 | bb | 16 |

(a)




(b)

Gambar 2. 8 (a) Tabel S-BOX, (b) Ilustrasi Sub-bytes

Berdasarkan penggambaran *sub-bytes* diatas terdapat nomor kolom dan baris. Pada tiap-tiap kotak didalam blok *cipher* terdapat informasi dalam bentuk dua digit bilangan *hexadecimal*, dapat berupa angka, angka huruf, maupun huruf angka yang telah tercantum didalam rijndael S-BOX. Dalam setiap tahapnya diambil satu dari isi kotak pada matrik untuk dicocokkan dengan digit kiri untuk baris dan digit sebelah kanan untuk kolom. Setelah diketahui kolom dan baris mada dapat mengambil isi tabel dari rijndael S-BOX. Langkah terakhir didapatkan blok baru yang berasal dari pengubahan keseluruhan blok *cipher* yang berisi hasil dari pengukaran tiap – tiap isi pada blok yang telah disebutkan pada langkah sebelumnya.

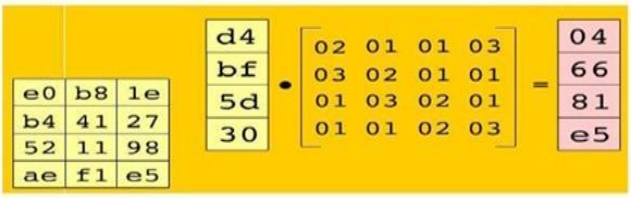
c. *ShiftRows*

Merupakan pergesaran tiap – tiap elemen blok yang dilakukan perbaris. Pada baris pertama tidak dilakukan pergeseran, baris kedua dilakukan

pergesaran sebanyak satu bit lalu baris ketiga dilakukan pergeseran sebanyak tiga bit.

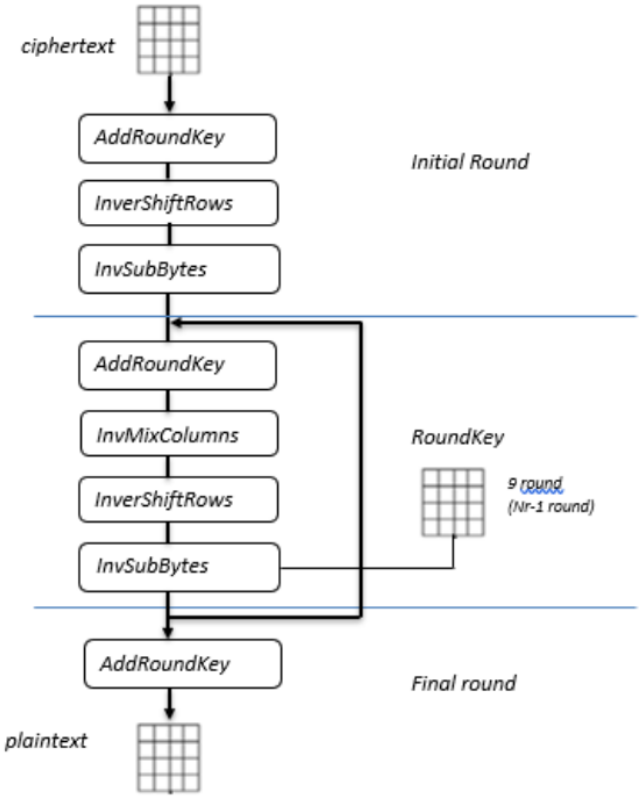
d. *Mix Columns*

Berguna untuk mengalikan setiap elemen pada blok cipher dengan matriks, perkalian dilakukan dengan menggunakan perkalian matriks biasa yang akan dimasukkan ke blok *cipher* baru, gambaran berikut menjelaskan proses perkalian [19].




Gambar 2. 9 *Mix Columm*

Sedangkan pada proses dekripsi AES-256, merupakan kebalikan dari proses enkripsi, yaitu *AddRowndKey*, kemudian *state* akan mengalami transformasi *InvShiftrows*, *InvSubBytes*, dan *InvMixColumns*. Gambaran alur algoritma Dekripsi AES-256 ditunjukkan pada gambar 2.10.

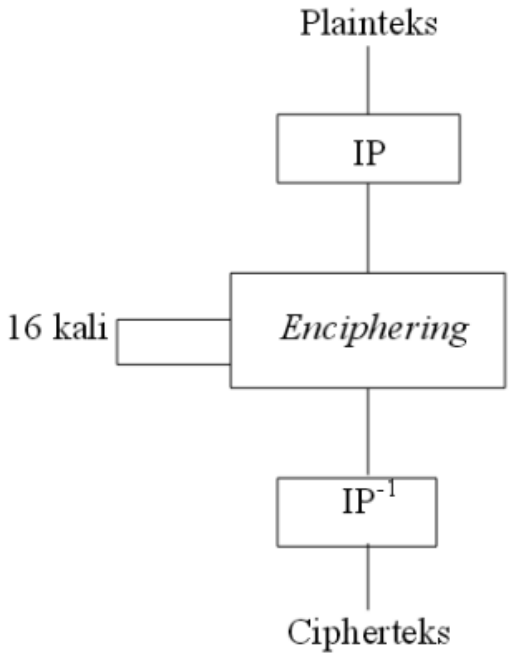



Gambar 2. 10 Alur Dekripsi Algoritma AES-256

### 2.2.4 *DATA ENCRYPTION ALGORITHM* (DES)

DES, juga dikenal sebagai ANSI's *Data Encryption Algorithm* (DEA) dan ISO's DEA1, adalah algoritma kriptografi simetris yang paling umum digunakan saat ini. Secara umum, DES adalah kriptosistem simetris dan jenis *cipher* blok. DES beroperasi pada ukuran blok 64-bit. DES mengenkripsi *plaintext* 64-bit menjadi *ciphertext* 64-bit menggunakan kunci internal 56-bit (kunci internal) [4]. Adapun skema global dari algoritma DES adalah sebagai berikut :

1. Blok *plaintext* dipermutasi dengan *matrix* permutasi awal (*initial permutation* atau IP)
2. Hasil permutasi awal kemudian di enciphering-sebanyak 16 kali (16 putaran). Setiap putaran menggunakan kunci yang berbeda.
3. Hasil *enciphering* kemudian dipermutasi balikan (invers initial permutation atau $IP^{-1}$) menjadi blok *ciphertext*.




Gambar 2. 11 Skema Algoritma DES

Pada proses enkripsi pada algoritma DES dimulai dengan *enchipering* (enkripsi) terhadap blok *plaintext* setelah dilakukan permutasi awal. Setiap blok plaintext mengalami *enchipering* sebanyak 16 putaran. Fungsi *expansi* yang memperluas blok Ri-1 dengan 15anjang 32 bit menjadi blok 48 bit. Kemudian hasil

expansi dari E(Ri-1) di XOR-kan dengan Ki sehingga menghasilkan *vector* A. kemudian *vector* A dibagi menjadi 8 kelompok, dimana pada setiap kelompok terdiri 6 bit yang menjadi masukkan pada proses substitusi. Pada proses substitusi dilakukan menggunakan delapan kotak S-BOX yang terdiri dari S1 sampai S8. Pada setiap kotak menerima masukkan 6 bit dan menghasilkan *output* 4 bit, dimana pada kelompok 6 bit pertama disubstitusikan dengan kotak S-BOX1, kemudian kelompok 6 bit kedua disubstitusikan dengan kotak S-BOX2, begitu seterusnya.

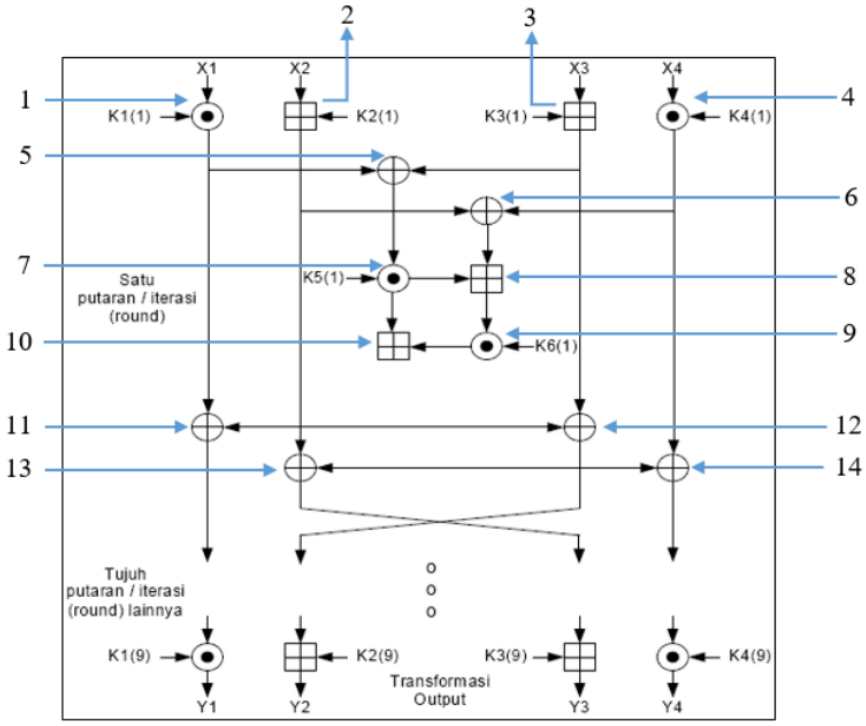
Pada proses dekripsi algoritma DES, lanhkah-langkah yang dilakukan merupakan kebalikan dari proses enkripsi. Kunci pada proses dekripsi merupakan kebalikan dari kunci proses enkripsi, yang terdiri dari K[16], K[15] sampai dengan K[1]. Kemudian dilakukan putaran sebanyak 16 kali pada proses *deciphering* [4].

### 2.2.5 *INTERNATIONAL DATA ENCRYPTION ALGORITHM* (IDEA)

IDEA adalah algoritma *cipher* blok kunci rahasia dan aman yang dikembangkan oleh James Massey dan Xuejia Lai. Algoritma ini dikembangkan pada tahun 1992 dari algoritma asli yang dikenal sebagai *Proposed Encryption Standard* (PES) dan *Inproved Proposed Encryption Standard* (IPES). IDEA adalah algoritma simetris yang beroperasi pada blok *plaintext* dengan panjang 64 bit dan panjang kunci 128 bit. Algoritma IDEA menggunakan algoritma yang sama untuk enkripsi dan dekripsi. Dan pesan rahasia yang dihasilkan oleh algoritma ini adalah blok pesan rahasia dengan lebar atau ukuran 64 bit [4]. Pada algoritma IDEA terdapat 2 sistem utama, yaitu :

1. Proses Enkripsi : ek (M) = C
2. Proses Dekripsi : dk ( C ) = M

Dengan keterangan e merupakan fungsi enkripsi, d merupakan fungsi dekripsi, K merupakan kunci enkripsi atau dekripsi, kemudian M merupakan *plaintext* atau pesan terbuka,sedangkan C merupakan *ciphertext* atau pesan rahasia [20].




Gambar 2. 12 Alur Proses Enkripsi Algoritma IDEA [21]

Proses enkripsi pada algoritma IDEA diawali dengan dibaginya *plaintext* 64 bit menjadi 4 buah sub blok yang memiliki panjang 16 bit, yaitu X1, X2, X3, dan X4. Kemudian empat sub blok tersebut menjadi masukkan pada putaran tahap pertama dari algoritma. Terdapat ada 8 putaran, dimana pada setiap putaran 4 sub blok di-XOR-kan, ditambahkan, dikalikan dengan yang lain serta dengan 6 buah subkey 16 bit diantara putaran sub blok kedua dan ketiga. Pada tahap akhir 4 sub blok dikombinasikan dengan 4 subkey pada transformasi *output* .

Pada proses dekripsi algoritma yang dipakai sama seperti pada saat proses enkripsi. Selain itu *subkey* yang dipakai terbalik dengan *subkey* pada proses Ketika melakukan enkripsi. Kemudian pada proses dekripsi menggunakan 52 buah *subkey* yang merupakan turunan dari enkripsi [22].

### 2.2.6 VIRTUALBOX

Oracle VM VirtualBox merupakan salah satu perangkat lunak virtualisasi, yang dapat digunakan untuk menginstall sistem operasi tambahan disistem operasi utama. VirtualBox berfungsi untuk melakukan virtualisasi sistem operasi. Penggunaan VirtualBox ditargetkan untuk *Server*, Desktop, dan penggunaan

Embeded. Berdasarkan jenis VMM yang ada, VirtualBox merupakan jenis *hypervisor type2* [23].

### 2.2.7 WIRESHARK

Wireshark merupakan salah satu *tool* yang berguna untuk menganalisa paket data pada jaringan. Wireshark juga berfungsi untuk memonitor jaringan dengan cara melakukan *capture* lalulintas data yang terjadi [24].