**PENERAPAN SISTEM KRIPTOGRAFI HYBRID MENGGUNAKAN ALGORITMA ADVANCED ENCRYPTION STANDARD  
DAN RIVEST SHAMIR ADLEMAN**

**SKRIPSI**

Diajukan untuk memenuhi salah satu persyaratan

kelulusan program Sarjana Strata Satu (S1)

Disusun oleh

**Afif Farakhan**

**NRP. 161014039**



**SEKOLAH TINGGI MANAJEMEN INFORMATIKA**

**D A N I L M U K O M P U T E R L P K I A**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**BANDUNG**

**2020**

# BAB I PENDAHULUAN

## Latar Belakang Masalah

Di zaman modern ini keamanan merupakan aspek yang sangat penting untuk kita perhatikan dalam penggunaan teknologi. Salah satu bidang keilmuan untuk menjaga keamanan data kita khususnya dalam berkomunikasi adalah kriptografi. Menurut Niels, Bruce dan Tadayoshi (2010), dalam buku nya yang berjudul Cryptography Engineering menyebutkan bahwa "Cryptography is the art and science of encryption." yang artinya kriptografi merupakan seni dan keilmuan mengamankan pesan.

Kriptografi adalah ilmu yang memanfaatkan rumus matematika, algoritma dan kunci yang diterapkan pada suatu teks (*plaintext*) untuk diacak menjadi tulisan yang tidak dapat dimengerti lagi teks asli nya (*ciphertext*). Fungsi dari kriptografi adalah untuk menjaga kerahasiaan informasi agar teks atau pesan hanya dapat dimengerti oleh orang yang berwenang untuk membaca nya.

Kriptografi di zaman modern ini sudah memiliki berbagai jenis dan berbagai macam algoritma yang mana masing – masing jenis atau algoritma tersebut memiliki karakteristik masing – masing. Diantaranya ada 2 jenis kriptografi modern yaitu jenis kunci simetris dan asimetris. Contoh algoritma kriptografi simetris diantaranya ada algoritma kriptografi DES (Data Encryption Standard), Blowfish dan AES (Advanced Encryption Standard) atau nama lainnya Rijndael. Sedangkan untuk algoritma kriptografi asimetris diantaranya ada algoritma ECC (Elliptic Curve Cryptography), ElGamal dan RSA (Rivest Shamir Adleman).

Kriptografi kunci simetris merupakan kegiatan mengacak suatu pesan yang hanya menggunakan 1 buah kunci baik itu untuk menenkrip pesan maupun mendekrip pesan. Sedangkan Kriptografi kunci asimetris merupakan kegiatan mengacak suatu pesan menggunakan 2 buah kunci dimana 1 kunci untuk menenkrip pesan dan 1 kunci lagi untuk mendekrip pesan. Dari kedua jenis kategori kriptografi modern ini masing - masing memiliki kelebihan maupun kelemahan nya tersendiri.

Pada kriptografi simetris jumlah kunci yang digunakan terbilang sedikit sehingga proses menenkrip maupun mendekrip akan sangat cepat. Kegiatan menenkrip maupun mendekrip suatu pesan hanya menggunakan 1 kunci yang sama yang artinya siapapun yang memiliki atau mengetahui kunci tersebut dapat mendekrip pesan sehingga isi pesan dapat diketahui. Disinilah yang dapat menjadi titik kelemahan kriptografi simetris dimana dibutuhkan nya saluran yang aman untuk pertukaran kunci antara pengirim pesan dan penerima pesan. Bila saluran tadi disadap oleh pihak yang tidak berwenang maka kunci pun bisa dicuri dan pesan yang sudah dienkrip dapat didekrip pula oleh pihak yang tidak berwenang tersebut.

Pada kriptografi asimetris kunci yang digunakan untuk menenkrip dan mendekrip pesan merupakan kunci yang berbeda sehingga terdapat 2 buah kunci yaitu kunci publik dan kunci privat. Kriptografi asimetris tidak membutuhkan saluran yang aman untuk pertukaran kunci namun letak kelemahan nya ada pada jumlah kunci yang digunakan. Kunci yang digunakan kriptografi asimetris terbilang banyak yang mana dalam proses enkripsi maupun dekripsi akan memakan waktu yang lama terlebih bila data yang akan dienkrip maupun didekrip memiliki ukuran yang sangat besar maka waktu yang dibutuhkan akan lebih lama lagi.

Namun sistem kriptografi modern yang baru yaitu sistem kriptografi hybrid dapat mengatasi masalah kelemahan 2 sistem kriptografi tersebut dengan menggabungkan kelebihan masing - masing untuk mengatasi kelemahan - kelemahan yang ada.

Maka dari itu permasalahan-permasalahan yang ada di atas menjadi gagasan untuk menuangkannya ke dalam tugas akhir dengan mengambil sebuah judul "PENERAPAN SISTEM KRIPTOGRAFI HYBRID MENGGUNAKAN ALGORITMA ADVANCED ENCRYPTION STANDARD DAN RIVEST SHAMIR ADLEMAN”.

## Identifikasi Permasalahan

Berdasarkan latar belakang masalah yang telah tertulis diatas, maka berikut adalah identifikasi masalah yang akan dijadikan bahan penelitian yaitu:

1. Bagaimana mengatasi kelemahan sistem kriptografi simetris dan sistem kriptografi asimetris dengan menggunakan sistem kriptografi hybrid.
2. Bagaimana proses dan penerapan sistem kriptografi hybrid dengan menggunakan algoritma kriptografi Advanced Encryption Standard (AES) dan algoritma kriptografi Rivest Shamir Adleman (RSA).

## Ruang Lingkup Permasalahan

Adapun ruang lingkup permasalahan dari penelitian ini yaitu:

1. Pesan yang akan dienkripsi dan didekripsi hanyalah pesan teks.
2. Format karakter yang digunakan untuk proses enkripsi dan dekripsi hanya karakter ASCII UTF-8.
3. Public key hanya digunakan untuk menenkrip dan private key hanya digunakan untuk mendekrip.
4. Bahasa pemrograman yang akan digunakan adalah bahasa pemrograman Python.

## Tujuan Perancangan

Tujuan dari penelitian ini yaitu menerapkan sistem kriptografi hybrid dengan menggunakan algoritma AES dan RSA untuk mengatasi kelemahan sistem kriptografi simetris dan asimetris.

## Metodologi Penelitian

Dalam menyelesaikan masalah, maka metode penelitian dan langkah – langkah yang diambil adalah sebagai berikut:

1. Tahap Observasi dan Konsultasi

Tahap pencarian terhadap sumber tertulis yang sudah tersedia dan terverifikasi baik dari buku, dokumentasi maupun jurnal yang relevan dengan permasalahan yang dibahas sehingga informasi yang didapat valid dan hasil dari skripsi ini dapat memperkual argumen – argument yang sudah ada.

1. Tahap Analisis Kebutuhan

Tahap menganalisis kebutuhan dengan mempelajari hasil studi literatur dan obervasi juga konsultasi guna mengetahui kebutuhan dan solusi untuk mengatasi masalah.

1. Proses Perancangan

Metode yang akan dilakukan ada metode pengembangan waterfall dengan pendekatan terstruktur.

1. Pembuatan Aplikasi

Tahap mengimplementasikan perencanaan yang sudah dibuat agar menjadi aplikasi yang bisa digunakan.

1. Pengujian

Tahap memastikan aplikasi berjalan sesuai dengan fungsinya dan kebutuhan yang sudah dibuat.

## Sistematika Penulisan

Dalam penulisan skripsi ini dibagi dalam 5 bab, yaitu:

**BAB I : PENDAHULUAN**

Bab ini digunakan untuk mendefinisikan persoalan, ruang lingkup dan perencanaan kegiatan dilakukan. Bab ini berisi latar belakang, identifikasi permasalahan, ruang lingkup dan batasan permasalahan, tujuan perancangan, metodologi penelitian dan sistematika penulisan.

**BAB II : DASAR TEORI**

Bab ini berisi teori-teori pendukung tentang teori permasalahan, pengembangan sistem, pengembangan perangkat lunak, yang meliputi: konsep kriptografi modern, konsep dasar algoritma kriptografi kunci simetris, asimetris dan hybrid, serta teori-teori lainnya yang digunakan untuk mendukung penganalisaan dan pengembangan sistem baru yang diusulkan.

**BAB III : ANALISIS DAN PERANCANGAN**

Bab ini berisi gambaran dan analisa yang dibutuhkan untuk penerapan algoritma kriptografi hybrid terhadap prototype perangkat lunak yang akan dibuat.

**BAB IV : IMPLEMENTASI DAN PENGUJIAN**

Bab ini berisi uraian lingkup dan batasan, kebutuhan sumber daya, dan hasil implementasi aplikasi juga terdapat hasil dari pengujian dari perangkat lunak yang sudah dibuat.

**BAB IV : KESIMPULAN DAN SARAN**

Bab ini berisi kesimpulan dan saran untuk kajian yang dapat dituliskan.

# BAB II DASAR TEORI

## Teori Tentang Permasalahan

Pada bab ini akan dibahas mengenai kriptografi dimulai dari definisi kriptografi, sistem kriptografi modern, sistem kriptografi simetris juga sistem kriptografi Advanced Encryption Standard (AES), sistem kriptografi asimetris juga sistem kriptografi Rivest Shamir Adleman (RSA) dan sistem kriptografi hybrid.

### Kriptografi

“Kriptografi adalah teknik untuk mengubah bentuk pesan menjadi bentuk lain yang memiliki arti berbeda dengan pesan itu sendiri, bahkan memungkin membuatnya seperti file yang rusak, sehingga sulit dibaca atau dimengerti oleh pihak lain.” (Ariyus, 2020). Jadi, kriptografi merupakan seni atau ilmu untuk menjaga keamanan, kerahasiaan atau keautentikasian suatu pesan, dimana pesan ini nanti nya hanya akan dibaca oleh orang – orang yang berhak untuk membaca nya saja dan aman dari pihak – pihak yang tidak berwenang untuk membaca nya.

### Sistem Kriptografi Modern

Sistem algoritma kriptografi modern biasanya terbagi kedalam 2 jenis yaitu sistem kriptografi kunci simetris dan kunci asimetris. Namun, seiring perkembangan nya zaman terdapat jenis baru yaitu sistem algoritma kriptografi hybrid. “Algoritma kriptografi modern umumnya beroperasi dalam mode bit ketimbang mode karakter (seperti yang dilakukan pada ciphersubstitusi atau ciphertransposisi dari algoritma kriptografi klasik). Operasi dalam mode bit berarti semua data dan informasi (baik kunci, plaintext, maupun ciphertext) dinyatakan dalam rangkaian (string) bit biner, 0 dan 1.” (Eka Risky, 2012).

#### Sistem Kriptografi Kunci Simetris

Sebuah sistem kriptografi yang menggunakan satu buah kunci baik untuk proses enkripsi maupun dekripsi. Sistem kriptografi simetris menggunakan jumlah kunci yang sedikit sehingga proses enkripsi maupun dekripsi hanya memakan waktu yang sebentar. Kunci dari algoritma ini bersifat rahasia namun dalam pertukaran kunci antara pengirim dan penerima pesan dibutuhkan saluran yang aman dari penyadapan. Contoh algoritma simetris yaitu Beaufort, Spritz, Blowfish, Twofish, DES (Data Encryption Standard).

#### Sistem Kriptografi Kunci Asimetris

Merupakan sistem algoritma yang menggunakan 2 buah kunci dimana satu untuk mengenkripsi dan satu lagi untuk mendekripsi. Kunci untuk mengenkripsi disebut kunci publik yang dapat diketahui oleh siapapun karena bersifat tidak rahasia. Sedangkan kunci untuk mendekripsi disebut kunci privat yang mana harus dijaga kerahasiaan nya. Kunci ini menggunakan jumlah kunci yang lebih banyak dari pada algoritma simetris sehingga kurang cocok untuk mengenkripsi data yang berjumlah besar karena proses nya akan memakan waktu yang lama. Contoh algoritma asimetris yaitu RSA (Riverst Shamir Adleman), ECC (Elliptic Curve Cryptography) dan ElGamal.

#### Sistem Kriptografi Hybrid

Sistem kriptografi ini disebut hybrid dikarenakan sistem ini merupakan pengkombinasian antara sistem kriptografi kunci simetris dan kunci asimetris. Tujuan dari penggunaan kritografi ini adalah untuk mengatasi kelemahan dari kedua algoritma itu sendiri. Dengan memanfaatkan kelebihan dari algoritma kunci asimetris, pertukaran kunci sesi akan memiliki solusi yang tepat untuk menyediakan saluran yang aman bagi pertukaran kunci. Sedangkan dengan memanfaatkan algoritma kunci simetris maka pesan maupun data yang akan dienkripsi akan lebih cepat proses nya dikarenakan jumlah kunci yang digunakan jauh lebih kecil daripada kunci kriptografi asimetris sehingga kegiatan kriptografi bisa dilakukan dengan waktu yang cepat.

### Algoritma Kriptografi AES

Menurut Ahmad Arif dan Putri Mandarani (2016) “Algoritma AES merupakan algoritma simetris yaitu menggunakan kunci yang sama untuk proses enkripsi dan dekripsi”. Input dan output dari algoritma ini yaitu berupa blok dengan jumlah bit tertentu. Pemilihan ukuran blok data dan kunci akan menentukan jumlah proses yang harus dilalui untuk proses enkripsi dan dekripsi.

#### Prosedur Kriptografi AES

Proses kriptografi AES dimulai dengan mengkonversikan pesan yang akan dienkrip dari format teks ASCII ke dalam format hex. Setelah itu, pesan yang sudah dikonversikan tersebut akan dimasukan ke dalam array state atau matriks blok 4X4 yang mana tiap - tiap sel matriks terdapat 1 byte (8-bit) dan secara keseluruhan berjumlah 16 byte (128-bit) yang akan diproses dengan kunci yang sudah dibuat secara acak. Kunci enkripsi bisa terdiri dari 128-bit, 192-bit atau 256-bit. Tiap – tiap kunci akan mempengaruhi jumlah ronde pada proses pengenkrpsian. Kunci akan dikonversi dari ASCII ke hex dan dimasukan ke dalam state 4X4. State pesan dan state kunci yang sudah dikonversi dari ASCII ke hex akan diproses di sesi add round key ronde pertama dengan cara operasi XOR (Exclusive or operation) dengan mengkonversi dari hex ke dalam bentuk binary lalu lakukan penjumlahan XOR. Dari XOR tersebut akan dihasilkan binary yang baru dan dikonversikan kembali ke dalam bentuk hex dan dimasukan ke dalam state yang baru.

Selanjutnya adalah proses shift row. State hasil operasi XOR tadi akan dilanjutkan dengan proses shift row. Dari keempat baris akan dilakukan perpindahan byte ke arah kiri. Barisan pertama tidak ada perpindahan, barisan kedua terdapat 1 byte berpindah ke kiri, barisan ketiga terdapat 2 byte berpindah ke kiri dan berisan keempat terdapat 3 byte berpindah ke kiri.

Selanjutnya adalah proses mix column. Hasil state yang sudah dilakukan proses shift row akan dilanjutkan dengan prose mix column dimana state akan dikalikan dan di XOR kan dengan predefined matrix sehingga menghasilkan array state yang baru. Tiap – tiap row dari predefined matrix akan dikalikan berpasang – pasangan dengan column state yang akan diproses dimulai dengan mengkonversi hex dari state ke dalam bentuk binary.

Setelah proses mix column selesai maka akan dilanjutkan dengan proses add round key ronde selanjutnya. Kegiatan ini akan diulang sebanyak 10 kali namun di ronde yang terakhir mix column tidak diperlukan lagi dan hasil nya ciphertext atau pesan yang sudah terenkripsi. Untuk prosedur kegiatan dekripsi perbedaan nya hanya pada urutan kegiatan nya yang dibalik dari akhir kembali lagi awal sehingga ciphertext akan kembali lagi menjadi plaintext.

Untuk mempermudah perhitungan dan mempercepat proses kriptografi AES ini maka digunakan nya tabel – tabel seperti tabel substitution box, tabel hex xor value dan tabel mix column sebagai berikut:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Tabel Subtitution Box** | | | | | | | | | | | | | | | | |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **A** | **B** | **C** | **D** | **E** | **F** |
| **0** | 63 | 7C | 77 | 7B | F2 | 6B | 6F | C5 | 30 | 01 | 67 | 2B | FE | D7 | AB | 76 |
| **1** | CA | 82 | C9 | 7D | FA | 59 | 47 | F0 | AD | D4 | A2 | AF | 9C | A4 | 72 | C0 |
| **2** | B7 | FD | 93 | 26 | 36 | 3F | F7 | CC | 34 | A5 | E5 | F1 | 71 | D8 | 31 | 15 |
| **3** | 04 | C7 | 23 | C3 | 18 | 96 | 05 | 9A | 07 | 12 | 80 | E2 | EB | 27 | B2 | 75 |
| **4** | 09 | 83 | 2C | 1A | 1B | 6E | 5A | A0 | 52 | 3B | D6 | B3 | 29 | E3 | 2F | 84 |
| **5** | 53 | D1 | 00 | ED | 20 | FC | B1 | 5B | 6A | CB | BE | 39 | 4A | 4C | 58 | CF |
| **6** | D0 | EF | AA | FB | 43 | 4D | 33 | 85 | 45 | F9 | 02 | 7F | 50 | 3C | 9F | A8 |
| **7** | 51 | A3 | 40 | 8F | 92 | 9D | 38 | F5 | BC | B6 | DA | 21 | 10 | FF | F3 | D2 |
| **8** | CD | 0C | 13 | EC | 5F | 97 | 44 | 17 | C4 | A7 | 7E | 3D | 64 | 5D | 19 | 73 |
| **9** | 60 | 81 | 4F | DC | 22 | 2A | 90 | 88 | 46 | EE | B8 | 14 | DE | 5E | 0B | DB |
| **A** | E0 | 32 | 3A | 0A | 49 | 06 | 24 | 5C | C2 | D3 | AC | 62 | 91 | 95 | E4 | 79 |
| **B** | E7 | C8 | 37 | 6D | 8D | D5 | 4E | A9 | 6C | 56 | F4 | EA | 65 | 7A | AE | 08 |
| **C** | BA | 78 | 25 | 2E | 1C | A6 | B4 | C6 | E8 | DD | 74 | 1F | 4B | BD | 8B | 8A |
| **D** | 70 | 3E | B5 | 66 | 48 | 03 | F6 | 0E | 61 | 35 | 57 | B9 | 86 | C1 | 1D | 9E |
| **E** | E1 | F8 | 98 | 11 | 69 | D9 | 8E | 94 | 9B | 1E | 87 | E9 | CE | 55 | 28 | DF |
| **F** | 8C | A1 | 89 | 0D | BF | E6 | 42 | 68 | 41 | 99 | 2D | 0F | B0 | 54 | BB | 16 |

**Tabel** Subtitution Box

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Tabel Subtitution Box Inverse** | | | | | | | | | | | | | | | | |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **A** | **B** | **C** | **D** | **E** | **F** |
| **0** | 52 | 09 | 6A | D5 | 30 | 36 | A5 | 38 | BF | 40 | A3 | 9E | 81 | F3 | D7 | FB |
| **1** | 7C | E3 | 39 | 82 | 9B | 2F | FF | 87 | 34 | 8E | 43 | 44 | C4 | DE | E9 | CB |
| **2** | 54 | 7B | 94 | 32 | A6 | C2 | 23 | 3D | EE | 4C | 95 | 0B | 42 | FA | C3 | 4E |
| **3** | 08 | 2E | A1 | 66 | 28 | D9 | 24 | B2 | 76 | 5B | A2 | 49 | 6D | 8B | D1 | 25 |
| **4** | 72 | F8 | F6 | 64 | 86 | 68 | 98 | 16 | D4 | A4 | 5C | CC | 5D | 65 | B6 | 92 |
| **5** | 6C | 70 | 48 | 50 | FD | ED | B9 | DA | 5E | 15 | 46 | 57 | A7 | 8D | 9D | 84 |
| **6** | 90 | D8 | AB | 00 | 8C | BC | D3 | 0A | F7 | E4 | 58 | 05 | B8 | B3 | 45 | 06 |
| **7** | D0 | 2C | 1E | 8F | CA | 3F | 0F | 02 | C1 | AF | BD | 03 | 01 | 13 | 8A | 6B |
| **8** | 3A | 91 | 11 | 41 | 4F | 67 | DC | EA | 97 | F2 | CF | CE | F0 | B4 | E6 | 73 |
| **9** | 96 | AC | 74 | 22 | E7 | AD | 35 | 85 | E2 | F9 | 37 | E8 | 1C | 75 | DF | 6E |
| **A** | 47 | F1 | 1A | 71 | 1D | 29 | C5 | 89 | 6F | B7 | 62 | 0E | AA | 18 | BE | 1B |
| **B** | FC | 56 | 3E | 4B | C6 | D2 | 79 | 20 | 9A | DB | C0 | FE | 78 | CD | 5A | F4 |
| **C** | 1F | DD | A8 | 33 | 88 | 07 | C7 | 31 | B1 | 12 | 10 | 59 | 27 | 80 | EC | 5F |
| **D** | 60 | 51 | 7F | A9 | 19 | B5 | 4A | 0D | 2D | E5 | 7A | 9F | 93 | C9 | 9C | EF |
| **E** | A0 | E0 | 3B | 4D | AE | 2A | F5 | B0 | C8 | EB | BB | 3C | 83 | 53 | 99 | 61 |
| **F** | 17 | 2B | 04 | 7E | BA | 77 | D6 | 26 | E1 | 69 | 14 | 63 | 55 | 21 | 0C | 7D |

**Tabel** Subtitution Box Inverse

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Tabel HEX XOR Value** | | | | | | | | | | | | | | | | |
|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **A** | **B** | **C** | **D** | **E** | **F** |
| **0** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| **1** | 1 | 0 | 3 | 2 | 5 | 4 | 7 | 6 | 9 | 8 | B | A | D | C | F | E |
| **2** | 2 | 3 | 0 | 1 | 6 | 7 | 4 | 5 | A | B | 8 | 9 | E | F | C | D |
| **3** | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | B | A | 9 | 8 | F | E | D | C |
| **4** | 4 | 5 | 6 | 7 | 0 | 1 | 2 | 3 | C | D | E | F | 8 | 9 | A | B |
| **5** | 5 | 4 | 7 | 6 | 1 | 0 | 3 | 2 | D | C | F | E | 9 | 8 | B | A |
| **6** | 6 | 7 | 4 | 5 | 2 | 3 | 0 | 1 | E | F | C | D | A | B | 8 | 9 |
| **7** | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | F | E | D | C | B | A | 9 | 8 |
| **8** | 8 | 9 | A | B | C | D | E | F | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| **9** | 9 | 8 | B | A | D | C | F | E | 1 | 0 | 3 | 2 | 5 | 4 | 7 | 6 |
| **A** | A | B | 8 | 9 | E | F | C | D | 2 | 3 | 0 | 1 | 6 | 7 | 4 | 5 |
| **B** | B | A | 9 | 8 | F | E | D | C | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 |
| **C** | C | D | E | F | 8 | 9 | A | B | 4 | 5 | 6 | 7 | 0 | 1 | 2 | 3 |
| **D** | D | C | F | E | 9 | 8 | B | A | 5 | 4 | 7 | 6 | 1 | 0 | 3 | 2 |
| **E** | E | F | C | D | A | B | 8 | 9 | 6 | 7 | 4 | 5 | 2 | 3 | 0 | 1 |
| **F** | F | E | D | C | B | A | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

**Tabel** HEX XOR Value

|  |  |  |  |
| --- | --- | --- | --- |
| **Tabel Mix Column (Prdefined Matrix)** | | | |
| 02 | 03 | 01 | 01 |
| 01 | 02 | 03 | 01 |
| 01 | 01 | 02 | 03 |
| 03 | 01 | 01 | 02 |

**Tabel** Subtitution Box

|  |  |  |  |
| --- | --- | --- | --- |
| **Tabel Inverse Mix Columns** | | | |
| 0E | 0B | 0D | 09 |
| 09 | 0E | 0B | 0D |
| 0D | 09 | 0E | 0B |
| 0B | 0D | 09 | 0E |

**Tabel** Subtitution Box

### Algoritma Kriptografi RSA

RSA merupakan singkatan dari Rivest Shamir Adleman yang merupakan nama – nama penemu dari algoritma ini. Algoritma RSA merupakan algoritma asimetris dimana kriptogafi ini menggunakan 2 kunci, yaitu kunci publik untuk mengenkripsi dan kunci privat untuk mendekripsi. Algoritma ini memiliki kelebihan yaitu tidak diperlukan nya saluran yang aman karena kunci untuk menenkripsi dan mendekripsi merupakan kunci yang berbeda.

#### Prosedur Kriptografi RSA

Sistem kriptografi RSA dimulai dengan pembuatan kunci publik dan privat. Pertama – tama pilih 2 bilangan prima *p* dan *q* yang tidak sama masing – masing sejumlah 1024-bits. Kemudian cari modulus *n* (public key) dengan rumus:

*n* = *p* x *q*

Kemudian cari fungsi Euler’s Totier dengan rumus:

φ = (*p*-1) x (*q*-1)

Setelah menemukan hasil dari φ (*n*) langkah selanjutnya adalah, pilih eksponen untuk *e* (encryption) dan *d* (decryption) menggunakan rumus:

K = 1 + φ

Untuk mencari nilai K berikutnya gunakan rumus:

K = K + φ

Cari hasil dari bilangan K yang bukan bilangan prima. Setelah itu lakukan faktorisasi terhadap bilang tersebut maka hasil dari faktorisasi tersebut dapat digunakan sebagai kunci publik *e* dan kunci privat *d*. Setelah menemukan nilai *e* dan nilai *d* maka kegiatan enkripsi dan dekripsi sudah dapat dilakukan.

Untuk enkripsi, gunakan rumus:

c = me (mod *n*)

maka akan didapatkan hasil ciphertext nya.

Untuk dekripsi, gunakan rumus:

m = cd (mod *n*)

maka akan didapatkan kembali plaintext nya.

### Algoritma Kriptografi Hybrid

Algoritma kriptografi hybrid merupakan kombinasi antara algoritma kunci simetris dan algoritma kunci asimetris. Baik algoritma kunci simetris maupun asimetris keduanya memiliki kelebihan dan kekurangan nya masing – masing. Namun, bila kedua algoritma ini dikombinasikan maka kekurangan dari kedua algoritma tersebut dapat diatasi dengan menggunakan sistem kriptografi hybrid ini. Kriptografi hybrid digunakan agar kegiatan pertukaran kunci bisa aman dari pencurian dengan menggunakan algoritma kunci asimetris dan kegiatan enkripsi dan dekripsi pesan akan cepat karena menggunakan algoritma kunci simetris.

#### Prosedur Kriptografi Hybrid

Diawali dengan proses pertukaran kunci secara aman dengan menggunakan algoritma kunci asimetris. Pertama – tama pengirim pesan akan membuat kunci sesi sedangkan penerima pesan akan membuat kunci publik dan kunci privat. Kunci publik milik penerima pesan akan digunakan oleh pengirim pesan untuk mengenkripsi kunci sesi yang sudah dibuat sebelumnya. Lalu kunci sesi yang sudah dienkripsi akan dikirim kepada penerima pesan. Penerima pesan akan mendekripsi kunci sesi yang sudah terenkripsi tadi dengan menggunakan kunci privat miliknya. Setelah kedua belah pihak memiliki kunci sesi masing – masing disini lah proses pertukaran pesan, data maupun informasi rahasia dimulai. Pengirim pesan akan menggunakan kunci sesi untuk mengenkripsi pesan atau data yang akan dikirim kepada penerima pesan. Penerima pesan pun akan dapat mendekripsi pesan yang terenkripsi tadi dengan kunci sesi yang sudah diterima oleh nya sebelum nya.

### Bahasa Pemrograman Python

Python merupakan interpreter bahasa pemrograman tingkat tinggi berbasis objek dengan semantik yang dinamis, dimana bersifat freeware atau perangkat bebas dalam arti sebenarnya, tidak ada batasan dalam penyalinannya atau mendistribusikannya. Lengkap dengan source codenya, debugger dan profiler, antarmuka yang terkandung di dalamnya untuk pelayanan antarmuka, fungsi sistem, antarmuka pengguna grafis (GUI), dan basis datanya. Python dapat digunakan dalam beberapa sistem operasi, seperti kebanyakan sistem UNIX, PCs (DOS, Windows, OS/2), Macintosh, dan lainnya. Pada kebanyakan sistem operasi linux, bahasa pemrograman ini menjadi standarisasi untuk disertakan dalam paket distribusinya.

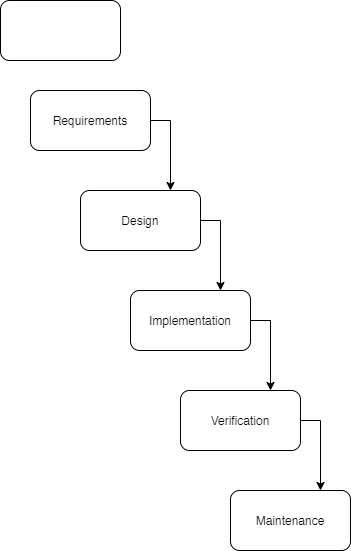
## Metodologi Yang Digunakan

Sub bab ini menjelaskan tentang metodologi yang digunakan yaitu model *waterfall* yang digunakan dalam perancangan sistem. Pada sub bab ini juga dijabarkan SDLC yaitu tools atau alat yang digunakan untuk tahapan perancangan.

### Pengembangan Perangkat Lunak (*Waterfall*)

Model *waterfall* adalah proses pengembangan perangkat lunak tradisional yang umum digunakan dalam proyek-proyek perangkat lunak yang paling pembangunan (Fahrurrozi, 2012). Model ini disebut *waterfall* dikarenakan prosesnya yang bertahap dari satu langkah ke langkah lainnya. Langkah – langkah model waterfall terdiri dari:

1. Kebutuhan berbasis pengujian (Requirements)
2. Desain (Design)
3. Implementasi (Implementation)
4. Pengujian: Verifikasi dan Validasi (Verification)
5. Pemeliharaan (Maintenance)



**Gambar** *Waterfall* Model

### II.2.2 Unified Modeling Language (UML)

Menurut Adi Nugroho, *Unified Modeling Language (UML)* adalah alat bantu analis serta perancangan perangkat lunak berbasis objek. UML merupakan metode perancangan sistem berorientasi objek dengan permodelan secara visual. UML memiliki berbagai macam diagram mulai dari use case diagram, activity diagram, sequene diagram dll.

### II.2.3 Teknik Pengumpulan Data Yang Digunakan

**Studi Literatur**

Menurut Danial dan Warsiah (2009), studi literatur adalah teknik penelitian dengan mengumpulkan sejumlah buku-buku, majalah, liflet, artikel, dan lain-lain yang berkenaan dengan masalah dan tujuan penelitian. Studi literature merupakan penelusuran sumber – sumber tulisan untuk menyelesaikan suatu masalah.

# BAB III ANALISIS DAN PERANCANGAN

Pada bab ini akan membahas analisis dan perancangan mulai dari gambaran perangkat lunak, analisis fungsional, perancangan antarmuka, perancangan arsitektural dan perancangan prosedural.

## Gambaran Perangkat Lunak

Perangkat lunak yang akan dibuat merupakan implementasi sistem kriptografi hybrid dimana sistem ini mengkombinasikan antara algoritma kunci simetris dan asimetris. Sistem kriptografi ini diharapkan dapat mengatasi permasalahan dari algoritma kunci simetris dan asimetris yaitu dalam hal keamanan dan kecepatan. Keamanan yang dimaksud disini adalah dari sisi algoritma kunci simetris dimana dibutuhkannya saluran yang aman untuk pertukaran kunci dikarenakan kunci yang digunakan untuk mengenkripsi dan mendekripsi merupakan kunci yang sama. Sedangkan kecepatan yang dimaksud disini yaitu dari sisi algoritma kunci asimetris dikarenakan kunci yang digunakan terbilang jauh lebih besar dari pada kunci simetris sehingga proses enkripsi maupun dekripsi pun memakan waktu yang jauh lebih lama. Implementasi sistem kriptografi hybrid ini akan menggunakan algoritma kriptografi kunci simetris AES (Advanced Encryption Standard) dan algoritma kriptografi kunci asimetris RSA (Rivest Shamir Adleman). Perangkat lunak yang akan dibuat akan berbasis desktop dan akan berfokus pada alur, rumus dan cara kerja kriptografi hybrid itu sendiri dan bukan pada pertukaran pesan seperti aplikasi email atau chatting.

## Analisis Algoritma Kriptografi Hybrid

Algoritma kriptografi hybrid merupakan kombinasi antara 2 buat algoritma yaitu algoritma kunci simetris dan asimetris. Tujuan pemanfaatan algoritma ini adalah untuk mengatasi kelemahan dari algoritma simetris dimana dibutuhkan nya saluran yang aman untuk pertukaran kunci dikarenakan kunci yang digunakan baik untuk enkripsi maupun dekripsi merupakan kunci yang sama. Sedangkan untuk algoritma kunci asimetris adalah besar nya kunci yang digunakan mengakibatkan proses enkripsi maupun dekripsi pesan atau data memakan waktu yang lama terlebih bila pesan atau data tersebut memiliki ukuran yang sangat besar maka proses nya akan lebih lama lagi.

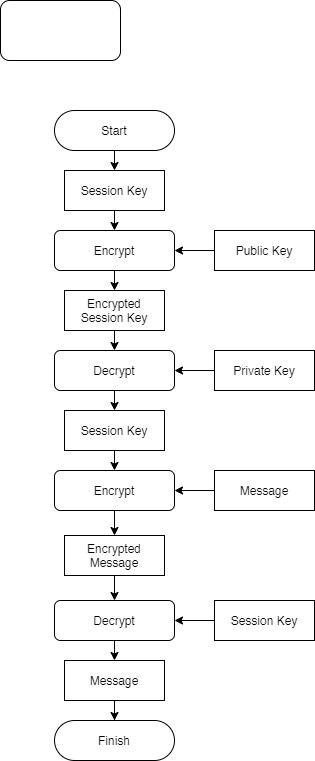
Algoritma kriptografi hybrid terbagi kedalam 2 tahap, yaitu:

1. Pertukaran kunci
2. Pertukaran pesan

Di dalam tahap pertukaran kunci, algoritma kriptografi yang digunakan adalah algoritma kriptografi kunci asimetris. Pada tahap ini pengirim pesan akan membangkitkan sebuah kunci sesi dimana kunci sesi ini merupakan kunci simetris untuk melakukan pertukaran pesan. Dikarenakan algoritma kunci simetris hanya menggunakan sebuah kunci yang sama dalam kegiatan enkripsi maupun dekripsi maka dibutuhkan nya saluran atau metode pertukaran kunci yang aman agar kunci tidak dicuri oleh pihak ketiga. Maka digunakanlah algoritma kunci asimetris atau sering juga disebut kunci publik dan privat dimana kunci sesi akan dienkripsi dengan menggunakan kunci publik milik penerima pesan dan hanya penerima pesan saja yang memiliki kunci privat untuk mendekripsi kunci sesi yang sudah dienkripsi oleh kunci publik milik nya. Dengan begini kunci sesi akan aman dari pencurian dikarenakan hanya penerima pesan saja yang dapat mendekripsi kunci sesi dengan menggunakan kunci privat miliknya.

Selanjutnya merupakan tahap pertukaran pesan. Pada tahap ini algoritma yang digunakan adalah algoritma kunci simetris. Baik pengirim pesan maupun penerima pesan sudah memiliki kunci sesi untuk pertukaran pesan. Yang harus dilakukan selanjutnya yaitu pengirim pesan akan mengenkripsi pesan yang akan dia kirim kepada penerima pesan dengan menggunakan kunci sesi yang sudah dibuat sebelumnya. Baik pesan maupun data sebesar apapun akan lebih cepat dienkripsi maupun didekripsi dengan menggunakan algoritma ini dikarenakan kunci yang digunakan jauh lebih kecil dibandingkan dengan algoritma kunci asimetris, inilah alasan mengapa kita tidak menggunakan kunci asimetris untuk setiap pertukaran pesan. Setelah pesan yang terenkripsi diterima oleh penerima pesan selanjutnya pesan tersebut akan didekripsi menggunakan kunci sesi yang sudah ia terima dari pengirim pesan. Dengan begini baik pesan, informasi, maupun data sebesar apapun akan lebih cepat proses nya dibandingkan menggunakan algoritma asimetris untuk kegiatan enkripsi dan dekripsi data yang besar.

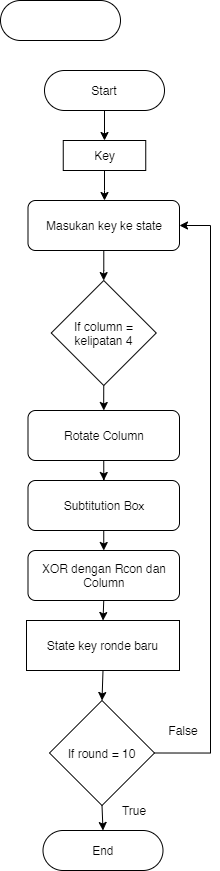
## Flowchart Kriptografi Hybrid



**Gambar** flowchart sistem kriptografi hybrid

Dimulai dengan kunci sesi yang akan dienkripsi dengan kunci publik oleh pengirim pesan. Setelah kunci sesi sudah dienkripsi maka akan dikirim kepada penerima pesan. Penerima pesan akan mendekripsi dengan kunci privat milik nya dan sekarang dia pun memiliki kunci sesi yang sama dengan pengirim pesan. Lalu pengirim pesan akan menggunakan kunci sesi nya untuk mengenkripsi pesan yang akan dikirim kepada penerima pesan dan penerima pesan akan mendekripsi dengan kunci sesi yang sama untuk mendekripsi pesan tersebut.

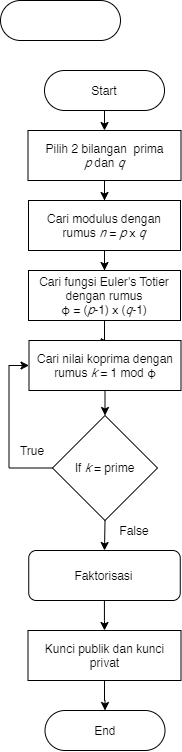
### Flowchart Pembangkitan Kunci Sesi (Simetris)



**Gambar** flowchart pembangkitan kunci sesi (algoritma AES)

Buat kunci dan masukkan ke dalam state 4X4. Setiap words kelipatan 4 akan dilakukan rotasi 1 kali. Setelah itu lakukan Subtitution dan dilanjutkan operasi XOR dengan Rcon dan Column sehingga akan menghasilkan State key ronde baru. Bila ronde belum 1 kali maka lakukan kembali sampai ronde mencapai 10 kali atau sama dengan words ke 43.

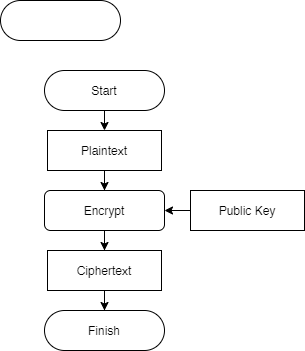
### Flowchart Pembangkitan Kunci Publik dan Privat (Asimetris)



**Gambar** flowchart pembangkitan kunci publik dan privat (algoritma RSA)

Pilih 2 bilangan prima *p* dan *q*. Cari modulus *n* = *p* x *q*. Cari Phi atau fungsi Euler’s Totier. Cari nilai koprima nya. Lakukan faktorisasi pada bilangan yang dapat dijadikan kunci eksponen *e* dan *d*.

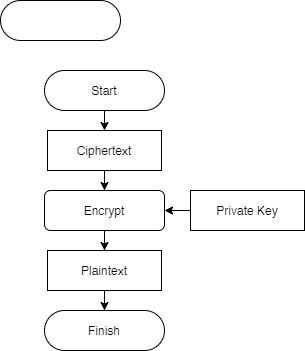
### Flowchart Enkripsi Kunci Sesi



**Gambar** flowchart enkripsi kunci sesi (algoritma RSA)

Plaintext akan dienkripsi dengan menggunakan publik key dan akan menjadi ciphertext.

### Flowchart Dekripsi Kunci Sesi



**Gambar** flowchart dekripsi kunci sesi (algoritma RSA)

Ciphertext akan dienkripsi dengan menggunakan private key dan akan kembali llagi menjadi plaintext.

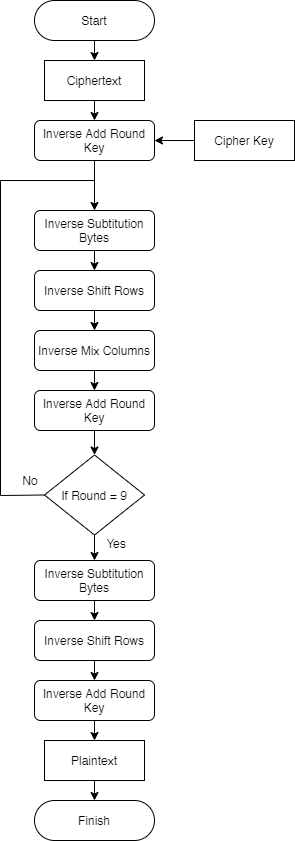
### Flowchart Enkripsi Pesan



**Gambar** flowchart enkripsi pesan (algoritma AES)

Masukan plaintext dan cipherkey yang sudah dibuat tadi ke dalam state 4X4. Lakukan Add Round Key, Subtitution, Shift Row, Mix Column dan Add Round Key lagi sebanyak 10 ronde. Namun pada ronde terakhir Mix Column tidak diperlukan sehingga akhirnya menghasilkan ciphertext.

### Flowchart Dekripsi Pesan



**Gambar** flowchart dekripsi pesan (algoritma AES)

Pada kegiatan dekripsi ciphertext perbedaan nya hanya pada alur kegiatan nya dimana kali ini semua kegiatan dibalikkan sehingga kembali menghasilkan plaintext.

## Analisis Rumus

### Rumus Pembangkitan Kunci Sesi

Pembangkitan kunci sesi dengan menggunakan algoritma kunci simetris menggunakan rumus – rumus sebagai berikut.

Untuk memasukan kunci ke dalam state per words column menggunakan sebagai berikut:

W[i] = W[i-4] XOR W[i-1]

Namun, khusus setiap words column kelipatan 4 rumus yang digunakan adalah sebagai berikut:

W[i] = W[i-4] XOR setelah Rcon

### Rumus Pembangkitan Kunci Publik dan Privat

Untuk membangkitkan kunci publik dan kunci privat rumus yang digunakan adalah:

1. Pilih 2 bilangan prima *p* dan *q*. Bilangan prima tidak boleh sama
2. Cari modulus *n* dengan rumus *n* = *p* x *q*
3. Cari fungsi Euler’s Totier dengan rumus φ (*n*) = (*p*-1) x (*q*-1)
4. Cari nilai koprima dengan rumus *k* = 1 mod φ sampai menemukan hasil bilangan yang bukan prima
5. Pilih bilangan yang bukan prima untuk difaktorisasi dan dijadikan kunci enkripsi dan dekripsi
6. Kunci publik untuk enkripsi akan disimbolkan sebagai *e*
7. Kunci privat untuk dekripsi akan disimbolkan sebagai *d*
8. Pesan yang akan dienkripsi akan disimbolkan sebagai *m*
9. Rumus untuk kunci publik adalah *m*e mod φ

### Rumus Enkripsi Kunci Sesi

Setelah membangkitkan kunci publik dan kunci privat selanjutnya adalah tahap enkripsi menggunakan kunci publik dengan rumus:

*c = me*mod φ

### Rumus Dekripsi Kunci Sesi

Untuk mendekripsi ciphertext maka rumus yang digunakan adalah:

*m* = *cd*mod φ

## Perancangan Antarmuka

### Perancangan Antarmuka Main Menu

Perancangan antarmuka halaman utama atau main menu ditunjukkan pada gambar berikut.



**Gambar** perancangan antarmuka *Main Menu*

Gambar diatas merupakan perancangan tampilan antarmuka halaman utama dari program kriptografi hybrid. Terdapat 3 tombol yaitu tombol session key exchange untuk melakukan kegiatan pertukaran kunci sesi, tombol message encryption & decryption untuk melakukan kegiatan enkripsi dan dekripsi pesan dan tombol quit untuk keluar dari program.

### Perancangan Antarmuka Session Key Exchange

Perancangan antarmuka untuk pertukaran kunci sesi ditunjukkan pada gambar berikut.



**Gambar** perancangan antarmuka *Session Key Exchange*

Pada gambar diatas ditampilkan perancangan antarmuka untuk menu kegiatan session key exchange atau pertukaran kunci sesi. Di menu ini terdapat 4 tombol yaitu tombol generate session key untuk membuat kunci sesi baru, tombol generate public & private key untuk membuat kunci publik dan kunci privat baru, tombol encrypt session key untuk mengenkripsi kunci sesi yang sudah dibuat dan tombol decrypt session key untuk mendekripsi kunci sesi yang sudah dienkripsi sebelumnya.

### Perancangan Antarmuka Generate Session Key

Perancangan antarmuka untuk pembuatan kunci sesi baru ditunjukkan pada gambar berikut.



**Gambar** perancangan antarmuka *Generate Session Key*

Pada gambar diatas ditampilkan perancangan antarmuka untuk membuat kunci sesi baru dimana ketika tombol generate session key diklik maka pada entry generated session key akan dibuat kunci sesi baru secara otomatis dan random.

### Perancangan Antarmuka Generate Public & Private Key

Perancangan antarmuka untuk pembuatan kunci publik dan privat baru ditunjukkan pada gambar berikut.



**Gambar** perancangan antarmuka *Generate Public & Private Key*

Pada gambar diatas ditunjukkan perancangan antarmuka untuk pembuatan kunci publik dan privat. Ketika tombol diklik maka sebuah kunci publik baru dan sebuah kunci privat baru akan dibuat secara otomatis dan random.

### Perancangan Antarmuka Encrypt Session Key

Perancangan antarmuka untuk menu kegiatan enkripsi kunci sesi dengan kunci publik ditunjukkan pada gambar berikut.



**Gambar** perancangan antarmuka *Encrypt Session Key*

Pada gambar diatas ditunjukkan perancangan antarmuka untuk kegiatan mengenkripsi kunci sesi dengan kunci publik. Kunci sesi dimasukkan ke dalam entry session key lalu kunci publik dimasukkan ke dalam text widget public key lalu klik tombol encrypt dan kunci sesi akan dienkrip dengan kunci publik.

### Perancangan Antarmuka Decrypt Session Key

Perancangan antarmuka untuk menu kegiatan dekripsi kunci sesi dengan kunci privat ditunjukkan pada gambar berikut.



**Gambar** perancangan antarmuka *Decrypt Session Key*

Pada gambar diatas ditunjukkan perancangan antarmuka untuk kegiatan mendekripsi kunci sesi dengan kunci privat. Kunci sesi yang sudah dienkrip dimasukkan ke dalam text widget encrypted session key lalu kunci privat dimasukkan ke dalam text widget private key lalu klik tombol decrypt dan kunci sesi akan didekrip dengan kunci privat.

### Perancangan Antarmuka Message Encryption & Decryption

Perancangan antarmuka untuk menu kegiatan enkripsi dan dekripsi pesan ditunjukkan pada gambar berikut.



**Gambar** perancangan antarmuka *Message Encrypt & Decrypt*

Pada gambar diatas ditunjukkan perancangan antarmuka untuk menu mengenkripsi dan mendekripsi pesan. Terdapat 2 tombol yaitu tombol encrypt message dan tombol decrypt message. Tombol encrypt message untuk membuka menu untuk kegiatan mengenkripsi pesan. Tombol decrypt message untuk membuka menu untuk kegiatan mengdekripsi pesan.

### Perancangan Antarmuka Encrypt Message

Perancangan antarmuka untuk menu kegiatan mengenkripsi pesan ditunjukkan pada gambar berikut.



**Gambar** perancangan antarmuka *Message Encrypt & Decrypt*

Pada gambar diatas ditunjukkan perancangan antarmuka untuk menu kegiatan mengenkripsi pesan. Masukan kunci sesi ke dalam entry session key lalu masukan pesan yang akan dienkrip dengan kunci sesi. Setelah itu klik tombol encrypt dan pesan tersebut akan dienkrip dan hasil nya ditampilkan di text widget message.

### Perancangan Antarmuka Decrypt Message

Perancangan antarmuka untuk menu kegiatan mendekripsi pesan ditunjukkan pada gambar berikut.



**Gambar** perancangan antarmuka *Message Encrypt & Decrypt*

Pada gambar diatas ditunjukkan perancangan antarmuka untuk menu kegiatan mendekripsi pesan. Masukan kunci sesi ke entry session key lalu masukan pesan yang akan didekripsi dengan kunci sesi. Klik tombol decrypt maka pesan tersebut akan didekrip dan ditampilkan di text widget decrypted message.

## Perancangan Prosedural

### Pseudocode

**Pseudocode generate\_session\_key**

begin

var key, decode\_key, GenSKEntry

key <– Fernet.generate\_key()

decode\_key <– key.decode("utf-8")

GenSKEntry.delete(0, END)

GenSKEntry.insert(0,0)

stop

**Pseudocode generate\_public\_private\_key**

begin

var keyPair, pubKey, pubKeyPEM, pubKeyPEMDecode, GenPublicKeyText, privKeyPEM, privKeyPemDecode, GenPrivateKeyText

keyPair <– RSA.generate(1024)

pubKey <– keyPair.publickey()

pubKeyPEM <– pubKey.exportKey()

pubKeyPEMDecode <– (pubKeyPEM.decode('ascii'))

GenPublicKeyText.delete('1.0', END)

GenPublicKeyText.insert('1.0', pubKeyPemDecode)

privKeyPEM <– KeyPair.exportKey()

privKeyPEMDecode <– (privketPEM.decode('ascii'))

GenPrivateKeyText.delete('1.0', END)

GenPrivateKeyText.insert('1.0', privKeyPEMDecode)

stop

**Pseudocode encrypt\_session\_key**

begin

var input\_session\_key, input\_session\_key\_encode, input\_public\_keyText, recipient\_key, encryptor, encrypted, encryptedSKhex, encryptedSKhex\_decode, encryptedSKText

input\_session\_key <– session\_key.get()

input\_session\_key\_encode <– input\_session\_key.encode()

input\_public\_keyText <– public\_keyText.get('1.0', 'end-1c')

recipient\_key <– RSA.import\_key(input\_public\_keyText)

encryptor <– PKCS1\_OAEP.new(recipient\_key)

encrypted <– encryptor.encrypt(input\_session\_key\_encode)

encryptedSKhex <– binascii.hexllify(encrypted)

encryptedSKhex\_decode = encryptedSKhex.decode('utf-8')

encryptedSKtext.delete('1.0', END)

encryptedSKtext.insert('1.0', encryptedSKhex\_decode)

stop

**Pseudocode decrypt\_session\_key**

begin

var input\_encryptedSK, input\_private\_keyText, input\_encryptedSK\_encode, decryptedSKunhex, private\_key, decryptor, decrypted, decryptedSKText

input\_encryptedSK <– encryptedSKText.get('1.0', 'end-1c')

input\_private\_keyText <– private\_keyText.get('1.0', 'end-1c')

input\_encryptedSK\_encode <– input\_encryptedSK.encode()

decryptedSKunhex <– binascii.unhexlify(input\_encryptedSK\_encode)

private\_key <– RSA.import\_key(input\_private\_keyText)

decryptor <– PKCS1\_OAEP.new(private\_key)

decrypted <– decryptor.decrypt(decryptedSKunhex)

decryptedSKText.delete('1.0', END)

decryptedSKText.insert('1.0', decrypted)

stop

**Pseudocode encrypt\_message**

begin

var input\_messageText, input\_session\_key, input\_messageText\_encode, input\_session\_key\_encode, cipher\_suite, encrypt\_messageText, decode\_encrypt\_messageText, encryptedMsgText

input\_messageText <– messageText.get('1.0', 'end-1c')

input\_session\_key <– session\_key.get()

input\_messageText\_encode <– input\_messageText.encode()

input\_session\_key\_encode <– input\_session\_key.encode()

cipher\_suite <– Fernet(input\_session\_key\_encode)

encrypt\_messageText <– cipher\_suite.encrypt(input\_messageText\_encode)

decode\_encrypt\_messageText <– encrypt\_messageText.decode("ascii")

encryptedMsgText.delete('1.0', END)

encryptedMsgText.insert('1.0', decode\_encrypt\_messageText)

stop

**Pseudocode decrypt\_message**

begin

var input\_messageText, input\_session\_key, input\_messageText\_encode, input\_session\_key\_encode, cipher\_suite, decrypt\_messageText, decode\_encrypt\_messageText, decryptedMsgText

input\_messageText <– messageText.get('1.0', 'end-1c')

input\_session\_key <– session\_key.get()

input\_messageText\_encode <– input\_messageText.encode()

input\_session\_key\_encode <– input\_session\_key.encode()

cipher\_suite <– Fernet(input\_session\_key\_encode)

decrypt\_messageText <– cipher\_suite.decrypt(input\_messageText\_encode)

decode\_encrypt\_messageText <– decrypt\_messageText.decode("ascii")

decryptedMsgText.delete('1.0', END)

decryptedMsgText.insert('1.0', decode\_encrypt\_messageText)

stop

# BAB IV IMPLEMENTASI DAN PENGUJIAN

## Implementasi

### Sistem Kriptografi Hybrid

Implementasi sistem kriptografi hybrid ini menggunakan algoritma kriptografi AES (Advanced Encryption Standard) untuk kunci simetris nya dan algoritma kriptografi RSA (Rivest Shamir Adleman) untuk kunci asimetrisnya. Sistem kriptografi hybrid terbagi ke dalam 2 tahap yaitu tahap pertukaran kunci dan tahap pertukaran pesan.

Katakanlah ada seseorang yang akan mengirim pesan atau informasi yang bersifat rahasia bernama Alice kepada penerima pesan bernama Bob. Dalam kegiatan pengiriman pesan ini, Alice dan Bob menggunakan sistem kriptografi hybrid.

Langkah pertama yang harus dilakukan adalah Alice akan membangkitkan kunci sesi baru untuk kegiatan pertukaran pesan menggunakan algoritma kunci simetris. Bob pun harus membangkitkan kunci publik dan kunci privat atau juga disebut kunci asimetris. Ini bertujuan untuk mengamankan kunci sesi dari penyadapan atau pencurian kunci sesi oleh pihak ketiga katakanlah bernama Eve. Bila hanya menggunakan algoritma kunci simetris permasalahan nya ada pada sisi keamanan dari kunci itu sendiri. Dikarenakan kunci untuk mengenkripsi maupun mendekripsi adalah sebuah kunci yang sama maka Eve dapat melakukan penyadapan terhadap Alice dan Eve pun dapat melakukan dekripsi terhadap pesan terenkripsi milik Alice. Disinilah peran kunci asimetris milik Bob. Alice akan menggunakan kunci publik milik Bob untuk mengenkripsi kunci sesi miliknya menggunakan algoritma kriptografi RSA. Dengan ini, hanya Bob lah satu – satu nya pemilik kunci privat yang dapat mendekripsi kunci sesi milik Alice dan walaupun Eve bisa mendapatkan kunci sesi yang terenkripsi dari Alice namun hanya Bob lah yang dapat mendekripsi nya karena hanya dia yang memiliki kunci privat. Setelah Bob menerima kunci sesi yang sudah terenkripsi tadi, Bob hanya perlu mendekripsi nya dengan kunci privat milik nya. Sekarang kunci sesi sudah aman di tangan Bob dan Eve gagal mengetahui kunci sesi dari Alice.

Alice dan Bob masing – masing sudah memiliki kunci sesi untuk pertukaran pesan. Selanjutnya Alice akan mengenkripsi pesan yang akan dia kirim kepada Bob dengan menggunakan algoritma kriptografi AES. Alasan menggunakan algoritma kunci simetris adalah sewaktu – waktu Alice harus mengirimkan data yang berukuran sangat besar, algoritma kunci simetris adalah algoritma yang efektif untuk melakukan ini dikarenakan kunci yang digunakan oleh algoritma simetris berjumlah sedikit sehingga proses untuk kegiatan enkripsi maupun dekripsi dapat dilakukan dengan waktu yang cepat. Alice mengenkripsi pesan nya dengan kunci sesi yang sudah ia buat sebelumnya dan mengirim nya kepada Bob. Setelah Bob menerima pesan terenkripsi dari Alice, Bob hanya perlu mendekripsi nya dengan kunci sesi yang sudah ia dapat sebelumnya. Sekarang kedua nya dapat melakukan pertukaran pesan, informasi rahasia, maupun data yang besar dengan waktu yang cepat karena proses enkripsi dan dekripsi data menggunakan kunci yang sedikit.

### Tahap Pembentukan Kunci Sesi (Simetris)

Alice akan membuat kunci sesi baru dengan menggunakan algoritma kunci simetris AES. Kunci simetris AES yang paling dasar berjumlah 128-bit dan memiliki 10 ronde. Katakan Alice memiliki kunci sesi “KunciAES16ByteYA” kunci ini akan diproses sehingga dapat mengenkripsi plaintext menjadi ciphertext.

Pertama – tama kunci akan dikonversi kedalam bentuk hex seperti berikut:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ASCII | K | u | n | c | i | A | E | S | 1 | 6 | B | y | t | e | Y | A |
| HEX | 4B | 75 | 6E | 63 | 69 | 41 | 45 | 53 | 31 | 36 | 42 | 79 | 74 | 65 | 59 | 41 |

Setelah itu hasil hex tersebut akan dimasukan kedalam words column ronde ke 0 berbentuk state seperti berikut:

|  |  |  |  |
| --- | --- | --- | --- |
| 4B | 69 | 31 | 74 |
| 75 | 41 | 36 | 65 |
| 6E | 45 | 42 | 59 |
| 63 | 53 | 79 | 41 |

Pada setiap kolom kelipatan ke 4 terdapat 1 putaran atau rotate byte ke atas seperti:

|  |
| --- |
| 74 |
| 65 |
| 59 |
| 41 |

|  |
| --- |
| 65 |
| 59 |
| 41 |
| 74 |

=

Setelah itu lakukan subtitution dengan menggunakan Tabel Subtitution Box:

|  |
| --- |
| 65 |
| 59 |
| 41 |
| 74 |

|  |
| --- |
| 4D |
| CB |
| 83 |
| 92 |

=

Setelah melakukan proses substitution box langkah selanjutnya adalah operasi XOR dengan tabel Rcon dan words column pertama:

|  |
| --- |
| 4D |
| CB |
| 83 |
| 92 |

|  |
| --- |
| 01 |
| 00 |
| 00 |
| 00 |

|  |
| --- |
| 4B |
| CB |
| 83 |
| 92 |

+ +

Konversikan semua nya ke dalam bentuk biner dan lakukan XOR:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

+

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |

+ =

Kembalikan hasil biner kedalam bentuk hex:

|  |
| --- |
| 07 |
| BE |
| ED |
| F1 |

Lanjutkan perhitungan pada tiap – tiap kolim sampai membentuk ronde selanjutnya yaitu ronde 1:

|  |  |  |  |
| --- | --- | --- | --- |
| 07 | 6E | 5F | 2B |
| BE | FF | C9 | AC |
| ED | A8 | EA | B3 |
| F1 | A2 | DB | 9A |

Lakukan terus sampai mencapai ronde ke 10 atau sama dengan words column 43 sehingga menghasilkan kunci untuk kegiatan enkripsi pesan:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| W | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 0 | 4B | 69 | 31 | 74 | 07 | 6E | 5F | 2B | 94 | FA | A5 | 8E | AB | 51 | F4 | 7A |
| 1 | 75 | 41 | 36 | 65 | BE | FF | C9 | AC | D3 | 2C | E5 | 49 | 9A | B6 | 53 | 1A |
| 2 | 6E | 45 | 42 | 59 | ED | A8 | EA | B3 | 55 | FD | 17 | A4 | 44 | B9 | AE | 0A |
| 3 | 63 | 53 | 79 | 41 | F1 | A2 | DB | 9A | 00 | A2 | 79 | E3 | 19 | BB | C2 | 21 |
|  | ROUND 0 | | | | ROUND 1 | | | | ROUND 2 | | | | ROUND 3 | | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| 01 | 50 | A4 | DE | 66 | 36 | 92 | 4C | 9F | A9 | 3B | 77 | 86 | 2F | 14 | 63 |
| FD | 4B | 18 | 02 | B4 | FF | E7 | E5 | E8 | 17 | F0 | 15 | 85 | 92 | 62 | 77 |
| B9 | 00 | AE | A4 | AD | AD | 03 | A7 | BA | 17 | 14 | B3 | CE | D9 | CD | 7E |
| C3 | 78 | BA | 9B | DE | A6 | 1C | 87 | F7 | 51 | 4D | CA | 02 | 53 | 1E | D4 |
| ROUND 4 | | | | ROUND 5 | | | | ROUND 6 | | | | ROUND 7 | | | |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 |
| F3 | DC | C8 | AB | 49 | 95 | 5D | F6 | 8E | 1B | 46 | B0 |
| 76 | E4 | 86 | F1 | B8 | 5C | DA | 2B | 4D | 11 | CB | E0 |
| 86 | 5F | 92 | EC | 56 | 09 | 9B | 77 | 8F | 86 | 1D | 6A |
| F9 | AA | B4 | 60 | 9B | 31 | 85 | E5 | D9 | E8 | 6D | 88 |
| ROUND 8 | | | | ROUND 9 | | | | ROUND 10 | | | |

### Tahap Pembentukan Kunci Publik dan Privat (Asimetris)

Setelah Alice selesai membuat kunci sesi, selanjutnya Bob akan membangkitkan kunci publik dan kunci privat dengan algoritma kriptografi RSA.

Pertama – tama pilih 2 bilangan prima *p* dan *q* dan masing – masing bilangan harus berbeda. Bob memilih bilangan 2 dan 7 sebagai *p* dan *q*. Selanjutnya Bob mencari hasil dari modulus dengan rumus *n* = *p* x *q* maka hasil yang didapat adalah 14. Setelah itu Bob akan mencari nilai dari *Phi* dengan rumus φ (*n*) = (*p*-1) x (*q*-1). Maka akan ditulis sebagai berikut:

φ (*n*) = (*p*-1) x (*q*-1)

φ (*n*) = (2-1) x (7-1)

φ (*n*) = 1 x 6

φ (*n*) = 6

Setelah nilai Phi ditemukan selanjutnya Bob harus mencari nilai dari *k* untuk menentukan kunci publik dan kunci privat untuk kegiatan enkripsi dan dekripsi. Rumus yang digunakan untuk mencari nilai *k* adalah sebagai berikut:

*k*  = 1 mod φ

*k* = (6 + 1) mod φ

*k* = 7 mod φ

Sekarang untuk mendapatkan nilai e dan d untuk enkripsi dan dekripsi lakukan pencarian dengan menggunakan rumus sebelumnya dengan menjumlahkan hasil *k* dengan φ, ditulis sebagai berikut:

*k* = 7 + 6 = 13

*k* = 13 + 6 = 19

*k* = 19 + 6 = 25

*k* = 25 + 6 = 31

*k* = 31 + 6 = 37

*k* = 37 + 6 = 43

*k* = 43 + 6 = 49

*k* = 49 + 6 = 55

Bilangan “55” dapat difaktorisasikan dan menghasilkan faktorisasi 5\*11. Bilangan 5 akan digunakan sebagai kunci enkripsi *e* sedangkan bilangan 11 dapat digunakan untuk kunci dekripsi *d*.

### Tahap Enkripsi Kunci Sesi dengan Algoritma RSA (Asimetris)

Pada tahap enkripsi kunci sesi, Alice memiliki kunci sesi sebagai berikut:

KunciAES16ByteYA

Kita ambil huruf pertama yaitu ”K” (huruf k kapital). Mula- mula konversikan huruf “K” ke dalam bentuk urutan angka agar perhitungan nya lebih mudah.

|  |  |
| --- | --- |
| Alfabet | K |
| Angka | 11 |

Setelah itu masukan ke dalam rumus enkripsi:

*c* = Ciphertext

*m* = Plaintext

*e* = Eksponen enkripsi

*n* = Modulus

Rumus enkripsi ; *c* = me mod *n*

Sehingga akan ditulis sebagai berikut:

*c* = 115 mod 14

*c* = 161051 mod 14

*c* = 9 mod 14

Hasil dari enkripsi adalah 9 atau dalam alfabet sama dengan huruf “I”

### Tahap Dekripsi Kunci Sesi dengan Algoritma RSA (Asimetris)

Pada tahap dekripsi, Bob akan menggunakan nilai eksponen *d* atau kunci privat nya untuk mendekripsi ciphertext milik Alice. Kembali menggunakan kasus pertama yaitu “I”. Konversikan huruf “I” ke dalam urutan angka menjadi angka 9. Gunakan rumus dekripsi sebagai berikut:

*c* = Ciphertext

*m* = Plaintext

*d* = Eksponen dekripsi

*n* = Modulus

Rumus dekripsi : *m* = cd mod *n*

Sehingga akan ditulis sebagai berikut:

*m* = 911 mod 14

*m* = 31,381,059,609 mod 14

*m* = 11 mod 14

Hasil dari dekripsi ciphertext adalah kembali lagi ke angka 11 yang bila dikembalikan ke dalam bentuk alfabet adalah “K”.

### Tahap Enkripsi Pesan dengan Algoritma AES (Simetris)

Pada tahap enkripsi pesan dari Alice kepada Bob, kedua pihak akan menggunakan kunci sesi yang sudah masing – masing miliki dari kegiatan pertukaran kunci sebelumnya.

Katakan Alice akan mengirim pesan kepada Bob yang berisi “Cryptography-128” dan akan mengenkripsinya dengan kunci sesi sebesar 128-bit yang artinya akan ada 10 ronde dalam tahap enkripsi nya.

Langkah pertama yang harus dilakukan adalah mengambil plaintext dan melakukan XOR dengan kunci, seperti pada contoh kasus berikut:

Plaintext : Cryptography-128

Key : KunciAES16ByteYA

Lakukan konversi pada plaintext dan key dari format ASCII ke dalam format Hex.

Plaintext :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ASCII | C | r | y | p | t | o | g | r | a | p | h | y | - | 1 | 2 | 8 |
| HEX | 43 | 72 | 79 | 70 | 74 | 6F | 67 | 72 | 61 | 70 | 68 | 79 | 2D | 31 | 32 | 38 |

Key :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ASCII | K | u | n | c | i | A | E | S | 1 | 6 | B | y | t | e | Y | A |
| HEX | 4B | 75 | 6E | 63 | 69 | 41 | 45 | 53 | 31 | 36 | 42 | 79 | 74 | 65 | 59 | 41 |

Kemudian masukan masing – masing hasil konversi tadi ke dalam state 4X4 seperti berikut:

State plaintext State Key

|  |  |  |  |
| --- | --- | --- | --- |
| 43 | 74 | 61 | 2D |
| 72 | 6F | 70 | 31 |
| 79 | 67 | 68 | 32 |
| 70 | 72 | 79 | 38 |

|  |  |  |  |
| --- | --- | --- | --- |
| 4B | 69 | 31 | 74 |
| 75 | 41 | 36 | 65 |
| 6E | 45 | 42 | 59 |
| 63 | 53 | 79 | 41 |

⊕

Dimulai dari state plaintext pertama yaitu 43 yang dikonversi ke binary menjadi 01000011 dan state key 4B yang dikonversi ke binary menjadi 01001011. Kemudian lakukan XOR:

0100 0011

⊕ = 0000 1000

0100 1011

Hasil 00001000 dikonversikan ke hex menjadi 08. Lakukan pada setiap bit sehingga hasil nya adalah sebagai berikut:

|  |  |  |  |
| --- | --- | --- | --- |
| 08 | 1D | 50 | 59 |
| 07 | 2E | 46 | 54 |
| 17 | 22 | 2A | 6B |
| 13 | 21 | 00 | 79 |

Selanjutnya adalah tahap substitution box dimana tiap tiap bit di dalam state akan diubah sesuai dengan tabel substitution box. Bila dilihat pada table substitution box maka 08 akan berubah menjadi 30. Hasil dari substitution box dari keseluruhan state adalah:

|  |  |  |  |
| --- | --- | --- | --- |
| 30 | A4 | 53 | CB |
| C5 | 31 | 5A | 20 |
| F0 | 93 | E5 | 7F |
| 7D | FD | 63 | B6 |

Setelah proses substitution box selesai selanjutnya adalah proses shift row dimana tiap – tiap byte dari state akan digeser sesuai baris nya masing – masing. Baris pertama tidak ada pergeseran byte. Baris ke dua bergeser 1 byte. Baris ke 3 bergeser 2 byte dan baris ke 4 bergeser 3 byte. Hasil adalah sebagai berikut:

|  |  |  |  |
| --- | --- | --- | --- |
| 30 | A4 | 53 | CB |
| 31 | 5A | 20 | C5 |
| E5 | 7F | F0 | 93 |
| B6 | 7D | FD | 63 |

Selanjutnya adalah proses mix column. Dalam proses ini kita meng XOR kan state dengan predefined matrix dengan melihat ke tabel galois.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tabel Mix Column (Prdefined Matrix)** | | | |
| 02 | 03 | 01 | 01 |
| 01 | 02 | 03 | 01 |
| 01 | 01 | 02 | 03 |
| 03 | 01 | 01 | 02 |

State 30 dan predefined matrix 02 jika dilihat ke tabel galois maka akan menghasilkan 60. Dilanjutkan dengan state 31 dengan predefined matrix 03 menghasilkan 53. State E5 dengan predefined matrix 01 bila dilihat pada tabel galois akan menghasilkan E5 dan state B6 dengan predefined matrix 01 akan menghasilkan B6. Hasil nya dapat dilihat sebagai berikut:

30 ⊕ 02 = 60

31 ⊕ 03 = 53

E5 ⊕ 01 = E5

B6 ⊕ 01 = B6

Setelah itu hasil – hasil tersebut masih harus diproses dengan melakukan XOR dengan tabel Hex XOR Value tanpa harus dikonversikan ke dalam bentuk binary.

Maka yang harus dilakukan adalah melakukan XOR dengan hasil yang sudah didapat sebelumnya dengan tabel Hex XOR Value yaitu:

60 ⊕ 53 = 33

Penjelasan:

Bilangan 6 di XOR kan dengan bilangan 5, sehingga:

6 ⊕ 5 = 3

Bilangan 0 di XOR kan dengan bilangan 3, sehingga:

0 ⊕ 3 = 3

Maka hasil nya adalah 33

Selanjutnya 33 akan di XOR kan dengan hasil bilangan selanjutnya yaitu E5 dan proses nya sama seperti sebelum nya sehingga menghasilkan bilangan D6. Lalu D6 akan diproses dengan bilangan B6 sehingga hasil akhirnya akan menghasilkan 60.

30 ⊕ 02 = 60

31 ⊕ 03 = 53 = 33

E5 ⊕ 01 = E5 = D6

B6 ⊕ 01 = B6 = 60

Lakukan proses mix column pada setiap byte state dan akan menghasilkan sebagai berikut:

|  |  |  |  |
| --- | --- | --- | --- |
| 60 | BF | CB | 29 |
| D0 | EC | E5 | 97 |
| 11 | 87 | 94 | 96 |
| F3 | 28 | C4 | D6 |

Dan terakhir lakukan lagi proses add round key dengan key selanjutnya.

|  |  |  |  |
| --- | --- | --- | --- |
| 07 | 6E | 5F | 2B |
| BE | FF | C9 | AC |
| ED | A8 | EA | B3 |
| F1 | A2 | DB | 9A |

Sehingga menghasilkan enkripsi ronde ke pertama:

|  |  |  |  |
| --- | --- | --- | --- |
| 67 | D1 | 94 | 02 |
| 6E | 13 | 2C | 3B |
| FC | 2F | 7E | 25 |
| 02 | 8A | 1F | 4C |

Lakukan sebanyak 10 ronde sehingga menghasilkan state:

|  |  |  |  |
| --- | --- | --- | --- |
| 25 | 93 | DE | 62 |
| 67 | 09 | 77 | 4E |
| 89 | 90 | DD | 95 |
| 66 | CF | C1 | 20 |

Konversikan ke dalam bentuk ASCII maka akan menghasilkan ciphertext:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| HEX | 25 | 67 | 89 | 66 | 93 | 09 | 90 | CF | DE | 77 | DD | C1 | 62 | 4E | 95 | 20 |
| CIPHERTEXT | % | g | ‰ | f | “ |  |  | Ï | Þ | w | Ý | Á | b | N | • |  |

### Tahap Dekripsi Pesan dengan Algoritma AES (Simetris)

Untuk melakukan proses dekripsi digunakan metode invers dari setiap langkah yang sudah dilakukan. Pada setiap proses enkripsi terdapat pula tabel dan metode invers nya yaitu invsubbytes, invshiftrows dan invmixcolumns. Dengan menginput ciphertext dan kunci yang sama maka proses akan dibalikan sehingga kita mendapatkan pesan semula yaitu Cryptography-128 dengan menggunakan kunci KunciAES16ByteYA.

### Lingkup dan Batasan

Lingkup dan batasan dari perangkat lunak yang telah penulis buat adalah sebagai berikut :

1. Perangkat lunak dibuat dengan menggunakan sistem kriptografi hybrid.

### Kebutuhan Sumber Daya

Untuk menggunakan perangkat lunak ini dibutuhkan spesifikasi yang mampu mendukung pengopersiannya, beberapa komponen yang dibutuhkan adalah sebagai berikut :

1. Kebutuhan Minimum Hardware

Untuk menjalankan perangkat lunak yang telah dibuat, dibutuhkan beberapa spesifikasi kebutuhan hardware sebagai berikut :

1. Processor : Quadcore 3,0 GHz
2. RAM : 4 GB
3. SSD : 256 GB
4. VGA : 2 GB
5. Monitor : 22’ inch FHD
6. Keyboard : Keyboard
7. Mouse : Mouse
8. Kebutuhan Minimum Software

Perangkat Lunak yang telah dibuat juga membutuhkan software yang mendukung dalam proses pembuatan maupun dalam penggunaan perangkat lunak tersebut.

Kebutuhan akan software atau perangkat lunak untuk mengembangkan aplikasi ini adalah sebagai berikut :

1. Sistem operasi : Windows 10 32/64 Bit
2. Text Editor : Visual Studio Code
3. Command Line : Command Prompt
4. Bahasa Pemrograman : Python 3.7.3
5. Modul : TKinter, Cryptography, Cryptodome
6. Spesifikasi Brainware

Spesifikasi kebutuhan brainware untuk mengembangkan aplikasi ini yaitu :

1. Memahami dasar python
2. Memahami dasar matematika
3. Memahami dasar kriptografi simetris
4. Memahami dasar kriptografi asimetris
5. Memahami dasar penggunaan terminal atau command prompt

### Implementasi Aplikasi



**Gambar** Tampilan Hybrid Cryptography Program Main Menu

Aplikasi dibuat menggunakan bahasa pemrograman python dengan menggunakan modul Tkinter sebagai GUI nya. Di dalam menu utama terdapat 3 tombol. Tombol Session Key Exchange adalah untuk kegiatan pertukaran kunci sesi dengan menggunakan algoritma kunci simetris untuk mengamankan kunci sesi yang akan digunakan untuk pertukaran pesan. Tombol Message Encryption & Decryption merupakan kegiatan pertukaran pesan setelah kegiatan pertukaran kunci sesi berhasil dilakukan. Tombol Quit merupakan tombol untuk keluar dari aplikasi.



**Gambar** Tampilan Session Key Exchange Menu

Menu Session Key Exchange memiliki 4 tombol. Tombol Generate Session Key adalah untuk membuka layar menu untuk kegiatan pembuatan kunci sesi. Tombol Generate Publik & Private Key adalah untuk membuka layar menu untuk membuat kunci publik dan kunci privat. Tombol Encrypt Session Key adalah untuk membuka layar menu untuk kegiatan enkripsi kunci sesi dengan menggunakan algoritma kunci asimetris. Tombol Decrypt Session Key adalah untuk membuka layar menu untuk kegiatan dekripsi kunci sesi yang terenkripsi.



**Gambar** Tampilan Generate Session Key Menu

Pada menu Generate Session Key, terdapat tombol Generate Session Key untuk membuat kunci sesi baru. Ketika tombol diklik maka kunci sesi akan dibuat secara acak dan otomatis dan langsung ditampilkan pada Entry Widget Generated Session Key.



**Gambar** Tampilan Generate Public & Private Key Menu

Pada menu Generate Public & Private Key, Terdapat 1 tombol bernama Generate Public & Private Key untuk membuat kunci publik dan kunci privat baru. Ketika tombol diklik maka kunci publik dan kunci privat baru akan ditampilkan pada Text Widget Generated Public Key dan Generated Private Key.



**Gambar** Tampilan Encrypt Session Key Menu

Pada menu Encrypt Session Key kegiatan pertukaran kunci dimulai. Setelah pengirim pesan membuat kunci sesi baru, kunci akan dienkripsi dengan kunci publik milik penerima pesan. Dengan memasukan kunci sesi pada entry widget session key dan memasukan pulic key kedalam text widget public key, klik tombol Encrypt untuk mengenkripsi kunci sesi dengan kunci publik menggunakan algoritma kunci asimetris. Hasil enkripsi akan ditampilkan pada text widget Encrypted Session Key.



**Gambar** Tampilan Decrypt Session Key Menu

Kegiatan dekripsi kunci sesi akan dilakukan di menu Decrypt Session Key. Penerima pesan akan menerima kunci sesi yang terenkripsi dengan kunci publik milik nya. Kunci sesi yang terenkripsi akan didekripsi dengan kunci privat milik penerima pesan. Penerima pesan akan memasukan kunci sesi yang terenkripsi ke dalam text widget encrypted session key lalu memasukan kunci privat milik nya ke dalam text widget private key. Setelah kedua input dimasukan, klik tombol Decrypt untuk mendekripsi kunci sesi yang terenkripsi untuk dikembalikan ke bentuk awal. Hasil dekripsi akan ditampilkan pada text widget Decrypted Session Key dan kunci sesi sudah bisa digunakan oleh penerima pesan.



**Gambar** Tampilan Message Encrypt & Decrypt Menu

Kegiatan enkripsi dan dekripsi pesan dengan menggunakan algoritma kunci simetris akan dilakukan pada Message Encrypt & Decrypt Menu. Terdapat 2 tombol, untuk membuka menu enkripsi pesan tekan tombol Encrypt Message, untuk mendekripsi pesan yang terenkripsi tekan tombol Decrypt Message.



**Gambar** Tampilan Message Encryption Menu

Pada menu Message Encryption, pengirim pesan akan memasukan kunci sesi pada enctry widget session key dan menuliskan pesan yang akan dienkripsi pada text widget enter the message. Setelah kunci dan pesan sudah diinput maka langkah selanjutnya adalah mengklik tombol Encrypt untuk mengenkripsi pesan dengan menggunakan algoritma kunci simetris dan hasil pesan yang dienkripsi akan ditampilkan pada text widget encrypted message.



**Gambar** Tampilan Message Decryption Menu

Pada Message Decryption Menu, penerima pesan akan memasukan kunci sesi ke dalam entry widget session key. Selanjutnya pesan yang telah dienkripsi akan dimasukan ke dalam text widget encrypted message. Lalu untuk mendekripsi pesan klik tombol Decrypt dan pesan pun akan didekripsi dan ditampilkan pada text widget decrypted message.

## Pengujian

### Lingkup dan Lingkungan

Lingkup dan lingkungan dari perangkat lunak yang telah penulis buat adalah sebagai berikut :

1. Aplikasi kriptografi hybrid ini hanya untuk mengenkripsi dan mendekripsi kunci sesi dan pesan, tidak menyediakan layanan untuk berkomunikasi seperti email, chatting dll.
2. Aplikasi kriptografi hybrid ini menggunakan algoritma RSA 1024-bit untuk kriptografi asimetris nya dan AES 128-bit untuk kriptografi simetris nya.
3. Kunci sesi adalah fixed dan tidak bisa sembarang membuat dan menggunakan kunci, kunci harus mengikuti dari yang sudah disediakan program.
4. Jenis teks yang dapat dienkripsi hanyalah text ASCII UTF-8 dan sebatas simbol – simbol matematika dasar. Tidak dapat mendekrip huruf Jepang, huruf Arab dll.

### Kebutuhan Sumber Daya

Perangkat Lunak yang telah dibuat juga membutuhkan software yang mendukung dalam proses pembuatan maupun dalam penggunaan perangkat lunak tersebut.

Kebutuhan akan software atau perangkat lunak untuk mengembangkan aplikasi ini adalah sebagai berikut :

1. Sistem operasi : Windows 10 32/64 Bit
2. Text Editor : Visual Studio Code
3. Command Line : Command Prompt
4. Bahasa Pemrograman : Python 3.7.3

### Hasil Pengujian

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Fungsi yang diuji | Cara pengujian | Hasil yang diharapkan | Hasil pengujian |
| 1. | Generate Session Key | Memanggil fungsi generate\_session\_key dengan mengklik tombol Generate Session Key pada Generate Session Key Menu | Membuat dan menampilkan kunci sesi baru secara otomatis dan secara acak pada entry widget generated session key | Berhasil |
| 2. | Generate Public & Private Key | Memanggil fungsi generate\_public\_private\_key dengan mengklik tombol Generate Public & Private Key pada Generate Public & Private Key Menu | Membuat dan menampilkan kunci publik dan privat baru pada masing - masing text widget | Berhasil |
| 3. | Encrypt Session Key | Mengenkrip kunci sesi dengan kunci publik dengan mengklik tombol encrypt untuk memanggil fungsi encrypt\_session\_key | Kunci sesi terenkripsi dan hasil nya tampil di text widget encrypted session key | Berhasil |
| 4. | Decrypt Session Key | Mendekrip kunci sesi dengan menggunakan kunci privat dengan mengklik tombol decrypt untuk memanggil fungsi decrypt\_session\_key | Kunci sesi yang terenkrip didekrip dengan pasangan kunci privat dan hasil nya ditampilkan pada text widget decrypted session key | Berhasil |
| 5. | Encrypt Message | Mengenkripsi pesan dengan memasukan kunci sesi dan pesan yang akan enkrip lalu klik tombol encrypt untuk memanggil fungsi encrypt\_message | Pesan akan terenkripsi dan hasil nya akan ditampilkan pada text widget encrypted message | Berhasil |
| 6. | Decrypt Message | Mendekrip pesan yang terenkripsi dengan memasukan kunci sesi dan pesan yang terenkripsi lalu klik tombol decrypt untuk memanggil fungsi decrypt\_message | Pesan yang terenkripsi akan didekrip dengan kunci sesi dan hasil nya akan ditampilkan pada text widget decrypted message | Berhasil |
| 7. | Quit | Mengklik tombol quit untuk keluar dari program dengan memanggil command tkinter root.destroy | Program akan berhenti dan window GUI akan hilang | Berhasil |
| 8. | Mengenkrip sebanyak mungkin kata yang dapat dienkrip | Mengenkrip 5,798 karakter ASCII UTF-8 (5,798 bytes) dan mendekrip nya sehingga kembali lagi ke teks awal | Pesan berhasil terenkrip dan terdekrip kembali ke teks awal | Berhasil |
| 9. | Mengenkrip simbol matematika lanjut, teks Jepang atau Arab | Memasukan simbol matematika lanjut, text Jepang atau Arab dan mengenkripsi nya dengan aplikasi kriptografi hybrid | Pesan dapat terdekrip kembali ke text asal nya (Simbol/Jepang/Arab) | Gagal |

# BAB V KESIMPULAN DAN SARAN

## Kesimpulan

Dari penelitian dan implementasi kriptografi hybrid tersebut maka dapat diambil kesimpulan sebagai berikut:

1. Penerapan sistem kriptografi hybrid untuk mengatasi kelemahan sistem kriptografi simetris dan asimetris dapat diaplikasikan dan diimplementasikan ke dalam aplikasi desktop berbasis python.

## Saran

Setelah melakukan implementasi kriptografi hybrid, penulis memberikan saran yang ditujukan untuk tahap pengembangan ke depannya. Berikut poin-poin saran yang disampaikan:

1. Menambahkan fitur untuk mengenkripsi dan mendekripsi tidak hanya sebuah pesan teks tetapi data – data seperti dokumen, gambar dll.
2. Meningkatkan teks yang dapat dienkripsi atau didekripsi seperti simbol – simbol matematika lanjutan, teks Jepang ataupun teks Arab.
3. Penambahan fitur untuk memilih tingkat jumlah kunci baik kunci simetris (kunci sesi) maupun asimetris (kunci publik dan kunci privat).
4. Bisa melakukan enkripsi dengan kunci privat dan mendekrip dengan kunci publik.

# DAFTAR PUSTAKA

1. S. Newman, *Building Microservices @ Squarespace*. O’Reilly Media, 2015.
2. Suhandinata et al, “Analisis Performa Kriptografi Hybrid Agoritma Blowfish dan Algoritma RSA,” vol. vi, no 1, pp. 1–10, 2019.
3. R. A. Putra, “Analisa Implementasi Arsitektur Microservices Berbasis Kontainer Pada Komunitas Pengembang Perangkat Lunak Sumber Terbuka (

Opendaylight Devops Community ),” vol. 9, pp. 150–162, 2019.

1. G. Munawar and A. Hodijah, “Analisis Model Arsitektur Microservice Pada Sistem Informasi DPLK,” *Sink. J. dan Penelit. Tek. Inform.*, vol. 3, no. 1, pp.

232–239, 2018.

1. K. P. T. Xyz, “RESTFul Web b Service Untuk Sistem m Pencatatan Transaksi St Studi,” vol. 2, no. April, 2016.
2. P. Ibeng, “Pengertian Implementasi, Tujuan, Contoh, Menurut Para Ahli.” 2020, [Online]. Available: https://pendidikan.co.id/. 10 Agustus 2020
3. M. A. Rahman, I. Kuswardayan, and R. Hariadi, “Perancangan dan Implementasi RESTful Web Service untuk Game Sosial Food Merchant Saga pada Perangkat Android,” vol. 2, no. 1, pp. 2–5, 2013.
4. I. G. Handika and A. Purbasari, “Pemanfaatan Framework Laravel Dalam Pembangunan Aplikasi E-Travel Berbasis Website,” pp. 8–9, 2018.
5. P. Sulistyorini, “Pemodelan Visual dengan Menggunakan UML dan Rational Rose,” *J. Teknol. Inf. Din. Vol.*, vol. XIV, no. 1, pp. 23–29, 2009.

# LAMPIRAN