**PENERAPAN SISTEM KRIPTOGRAFI HYBRID MENGGUNAKAN ALGORITMA ADVANCED ENCRYPTION SYSTEM  
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 ketika kita menggunakan teknologi - teknologi modern. Salah satu bidang keilmuan yang dapat menjaga keamanan kita di zaman modern ini 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." atau bisa diterjemahkan yang artinya kriptografi merupakan seni dan keilmuan mengamankan pesan.

Kriptografi merupakan ilmu yang memanfaatkan suatu rumus atau algoritma matematika dan kunci yang diterapkan pada suatu teks (*plaintext*) untuk diacak menjadi tulisan yang tidak dapat dimengerti (*ciphertext*) agar teks atau pesan tersebut tidak dapat dimengerti oleh orang - orang yang tidak berwenang untuk membaca nya dan hanya orang - orang tertentu saja yang memiliki kunci yang dapat memahami isi teks asli tersebut.

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 SYSTEM 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 System (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, sejarah kriptografi, sistem kriptografi modern, sistem kriptografi simetris juga sistem kriptografi Advanced Encryption System (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 sedikit. Kunci dari algoritma ini bersifat rahasia dan 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 yang mana 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 menkeripsi 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 penggabungan sistem kriptografi kunci simetris dan kunci asimetris. Dengan memanfaatkan kedua sistem kriptografi sebelumnya tujuan dari kritografi ini adalah mengatasi kelemahan dari masing – masing algoritma dengan memanfaatkan kelebihan dari masing – masing algoritma itu sendiri.

### 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 text ASCII ke dalam format hex. Setelah itu, hasil pesan panjang yang sudah dikonversikan tersebut akan dimasukan ke dalam array state atau matriks blocks 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 digenerate 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. Dari XOR tersebut akan dihasilkan binary yang sudah tidak dapat dimengerti lagi pesan aslinya dan akan dikonversikan kembali ke dalam bentuk hex dan dimasukan ke dalam hasil state yang baru.

Selanjutnya adalah proses shift row. Hasil state yang sudah dilakukan 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 pertama. Kegiatan ini akan diulang sebanyak 10 kali namun di ronder yang terakhir mix column tidak diperlukan lagi dan hasil nya ciphertext atau pesan yang sudah terenkripsi. Proses tadi merupakan dasar algoritma kriptografi oleh algoritma kriptografi AES 128-bit dengan kunci 128-bit sebanyak 10 ronde.

Pada Proses enkripsi awalnya teks asli dibentuk sebagai sebuah state. Kemudian sebelum ronde 1 dimulai blok teks asli dicampur dengan kunci ronde ke-0 (transformasi ini disebut AddRoundKey). Setelah itu, ronde ke-1 sampai dengan ronde ke-(Nr-1) dengan Nr adalah jumlah ronde. AES menggunakan 4 jenis transformasi yaitu:

1. *SubBytes*, sebagai transformasi subtitusi
2. *ShiftRows,* sebagai transformasi permutasi
3. *MixColumns,* sebagai transformasi pengacakan
4. *AddRoundKey,* sebagai transformasi penambahan kunci

Pada ronde terakhir, yaitu ronde ke-Nr dilakukan transformsi serupa dengan ronde lain namun tanpa transfomasi serupa dengan ronde lain namun tanpa transformasi MixColumns.

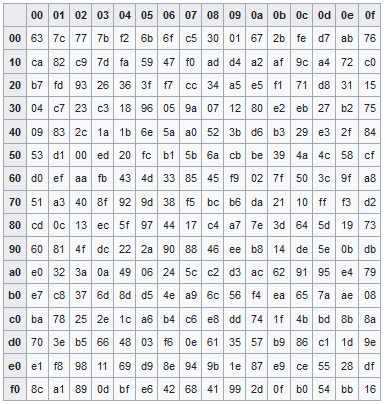
Penyandian AES membutuhkan kunci ronde untuk setiap ronde transformasi kunci ronde ini di bangkitkan (di ekspansi) dari kunci AES. Pada bagian ini di bahas bagaimana kunci ronde di bangkitkan oleh kunci AES. Kunci AES 128 bit atau 4 word menghasilkan sebuah larik sebanyak 44 word yang menjadi kunci. Berikut adalah langkah langkah mengekspansi kunci:

1. Pertama kunci AES 128 bit di organisir menjadi 4 word dan disalin ke word keluaran (W) pada 4 elemen pertama (W[0], W[1], W[2], W[3]).
2. Untuk elemen keluaran selanjutnya W[i]dengan i ={4,...,43} dihitung sebagai berikut:
3. Salin W [i-1] pada word t.
4. Jika i mod 4 = 0 ( I habis dibagi 4 ) maka lakukan W[i]= f (t,i) ⊕W[i-4] ,dengan fungsi f(t,i) adalah sebagai berikut:

f(t,i) = Subword (rotword(t)) ⊕RC[i/4]

1. Jika I mod 4 tidak sama dengan 0, lakukan W[i]= t⊕W[i-4].

Tabel Subtitution Box AES



#### Flowchart Kriptografi AES

Berikut adalah Flowchart pada algoritma Sistem Kriptografi AES:



Gambar flowchart algoritma kriptografi AES

#### Proses Enkripsi Kriptografi AES

Langkah pertama yang harus dilakukan adalah mengambil plaintext dan melakukan XOR dengan kunci, seperti contoh 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 |

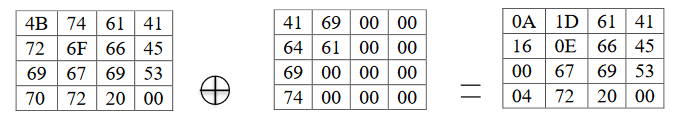
⊕

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

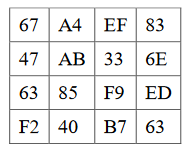
⊕ =

0100 1011

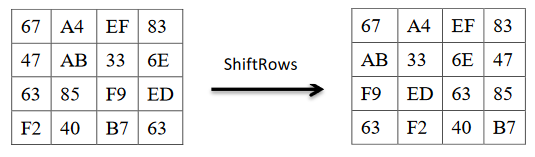


**Ronde 1**

1. Hasil Proses Subbyte ( menggunakan table s-box)



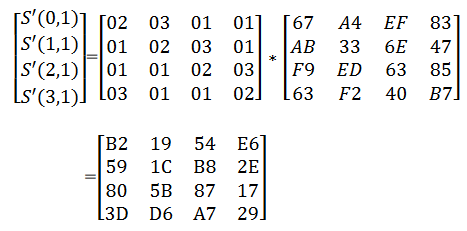
1. Transformasi ShiftRows



1. Proses MixColumns

Proses ini merupakan proses terbanyak dari pada proses proses lain setiap rounde. Kali ini penulis membagi proses mixcolumnsmenjadi 4 bagian untuk sebuah matriks atau state karena dikerjakan untuk setiap kolom sebagai berikut:

1. Proses mixcolumn untuk kolom pertama.



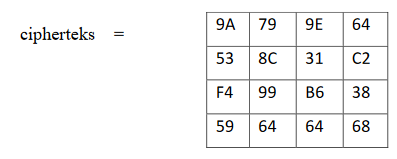
Pada ronde pertama didapat Cipherteks yang akan menjadi masukan atau input untuk ronde 2, begitu juga cipherteks yang didapat pada ronde 2 kan digunakan menjadi input pada ronde 3. Proses seperti ini berlangsung hingga ronde 10. Pada ronde 10 didapat hasil enkripsi sebagai berikut:

**Ronde ke-10**:





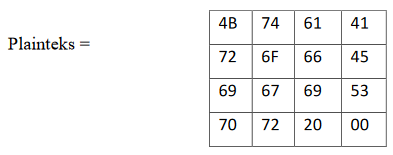
Pada ronde 10 transformasi yang dilakukan hanya 3 transformasi yaituSubbyte, ShiftRows,Addroundkey. Dan didapat cipherteks yang sesungguhnya yaitu:



jika didalam bentuk ASCII maka di dapat cipherteks: š S ô Y y Œ ™ d ž 1¶ d d Â 8 h.

#### Proses Dekripsi Kriptografi AES

Untuk mengubah kembali cipherteks menjadi plaintext maka dilakukan proses dekripsi dengan menggunakan transformasi invers semua transformasi dasar yang digunakan pada algoritma enkripsi AES. Setiap transformasi dasar AES memiliki transformasi invers, yaitu L: invsubbytes, invshiftrows, dan invmixcolumns.Dari proses dekripsi yang dilaksanakan 10 ronde didapat:



Plaintext yang di konversi menjadi bentuk ASCII menjadi: “Kriptografi AES”. Maka pada Algoritma kriptografi AES untuk Plaintext= “Kriptografi AES”dan kunci= “Aditia” didapat Cipherteks= “š S ô Y yŒ ™ d ž 1¶ d d Â 8 h”.

### Algoritma Kriptografi RSA

RSA merupakan algoritma kriptografi asimetris karena menggunakan dua kunci, yaitu kunci publik dan kunci pribadi. Ada tiga algoritma dalam kriptografi RSA, yaitu pembangkitan kunci, proses enkripsi, dan proses dekripsi. Algoritma ini memiliki tingkat keamanan yang terletak pada sulitnya memfaktorkan sebuah bilangan besar menjadi dua buah bilangan prima. Kelemahan dari algoritma kriptografi RSA adalah waktu yang dibutuhkan untuk melakukan proses pembangkitan kunci, enkripsi dan dekripsi lambat. Sedangkan kelebihannya terletak pada sulitnya memecahkan kunci dan penggunaan kunci yang lebih efektif.

#### Prosedur Kriptografi RSA

Sistem kriptografi RSA terdiri dari 3 prosedur: pembangkit kunci publik dan kunci privat, enkripsi dan dekripsi. Berikut ini adalah algoritma untuk prosedur pembangkit Kunci RSA:

Pertama – tama kita harus memilih 2 bilangan prima berjumlah saangat besar p & q masing – masing sebesar 1024-bits

Algoritma pembangkit kunci pada sistem kriptografi RSA membuat 2 buah kunci yaitu kunci publik dan kunci privat. Algoritma pembangkit kunci RSA:

1. Pilih dua bilangan prima *p* dan *q*. kedua bilangan *p* dan *q* ini tidak boleh sama. Hitung *N* = *p* x *q*. *N* hasil perkalian dari *p* dikalikan dengan *q.*
2. Hitung φ = (*p*-1)(*q*-1).
3. Pilih bilangan bulat (*integer*) antara satu dan φ (1 < *e* < φ) yang juga merupakan koprima dari φ.
4. Hitung *d* hingga *d e* ≡ 1 (mod φ).

Pada kunci publik terdiri dari:

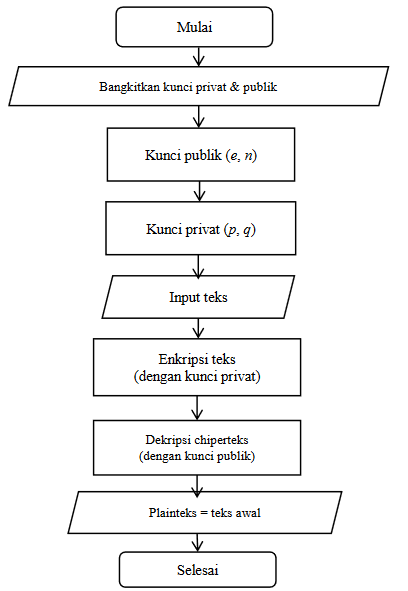
* *N*, modulus yang digunakan
* *e*, eksponen publik (untuk menenkripsi) tidak rahasia dan boleh diketahui

Pada kunci privat terdiri dari:

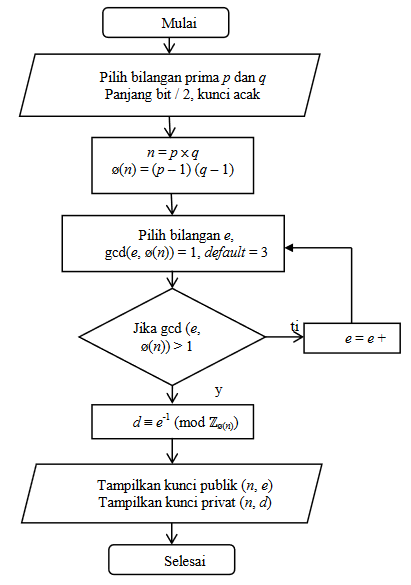
* *N*, modulus yang digunakan dan digunakan juga pada kunci publik
* *d*, eksponen privat (untuk mendekripsi) dan harus dijaga kerahasiaannya

#### Flowchart Kriptografi RSA

Berikut adalah Flowchart untuk Algoritma Sistem Kriptografi RSA:



Berikut adalah Flowchart pembangkitan kunci Sistem Kritografi RSA:



#### Proses Enkripsi Kriptografi RSA

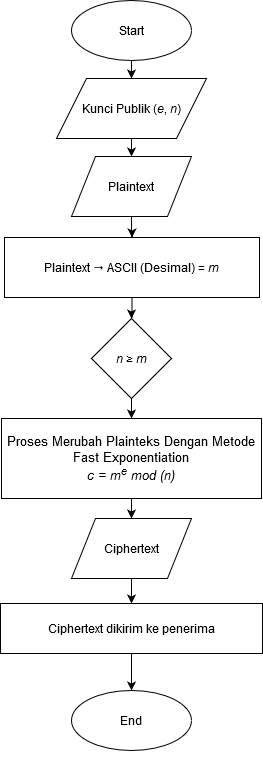
Misalkan Bob ingin mengirim pesan *m* ke Alice. Bob mengubah *m* menjadi angka *n* < *N*, menggunakan protokol yang sebelumnya telah disepakati dan dikenal sebagai *padding scheme.*

Maka Bob memiliki *n* dan mengetahui *N* dan *e*, yang telah diumumkan oleh Alice. Bob kemudian menghitung *ciphertext* *c* yang terkait pada *n:*

*c* = *ne* mod *N*

Perhitungan tersebut dapat diselesaikan dengan cepat menggunakan metode *exponentiation by squaring*. Bob kemudian mengirimkan *c* kepada Alice.

Berikut adalah Flowchart proses enkripsi pada algoritma Sistem Kriptografi RSA:



#### Proses Dekripsi Kriptografi RSA

Alice menerima *c* dari Bob, dan mengetahui *private key* yang digunakan oleh Alice sendiri. Alice kemudian memulihkan *n* dari *c* dengan langkah-langkah berikut:

*n* = *cd* mod *N*

Perhitungan di atas akan menghasilkan *n*, dengan begitu Alice dapat mengembalikan pesan semula *m*. Prosedur dekripsi bekerja karena

*cd* ≡ (*ne*)*d* ≡ *ned* (mod *N*)

Kemudian, dikarenakan *ed* ≡ 1 (mod p-1) dan *ed* ≡ 1 (mod q-1), hasil dari *Fermat’s little theorem.*

*ned* ≡ *n* (mod *p*)

Dan

*ned* ≡ *n* (mod *q*)

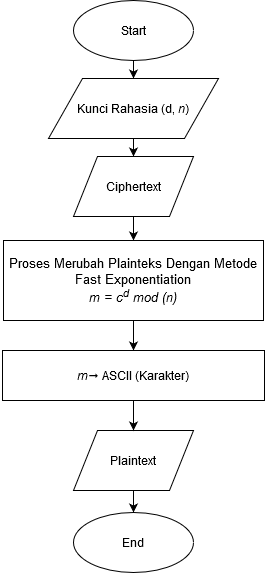
Dikarenakan *p* dan *q* merupakan bilangan prima yang berbeda, mengaplikasikan Chinese Remainder Theorem akan menghasilkan dua macam kongruen

*ned* ≡ *n* (mod *pq*)

Serta

*cd* ≡ *n* (mod *N*)

Berikut adalah Flowchart proses dekripsi pada algoritma Sistem Kriptografi RSA:

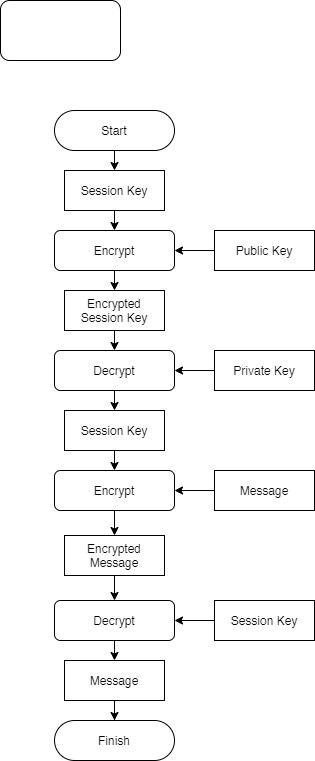


### Algoritma Kriptografi Hybrid

Kriptografi hybrid melakukan enkripsi secara berlapis. Hal ini secara teori akan membuat pemecahannya menjadi semakin kompleks. Algoritma kriptografi hybrid (hybrid cryptosystem) merupakan gabungan antara kriptografi simetris dan kriptografi asimetris. Permasalahan yang menarik pada bidang keamanan informasi adalah adanya trade off antara kecepatan dengan kenyamanan. Salah satu contohnya adalah bidang kriptografi. Tetapi hal ini dapat diatasi dengan penggunaan kriptografi hybrid. Kriptografi hybrid sering dipakai karena memanfaatkan keunggulan kecepatan pemrosesan data oleh algoritma kriptrografi simetrik dan kemudahan transfer kunci menggunakan algoritma kriptografi asimetrik. Hal ini mengakibatkan peningkatan kecepatan tanpa mengurangi kenyamanan serta keamanan.

#### Konsep Kriptografi Hybrid

#### Flowchart Kriptografi Hybrid



Gambar flowchart sistem kriptografi hybrid

#### Proses Kriptografi Hybrid

1. **Prosedur pada Sistem Kriptografi Hybrid**

Kriptografi hybrid menggabungkan kedua buah sistem kriptografi simetris dan sistem kriptografi asimetris. Katakan ada dua pihak yang akan melakukan komunikasi yaitu Alice (Pengirim Pesan) dan Bob (Penerima Pesan). Pertama-tama, Bob sang penerima pesan akan membuat (*Generate*) kunci RSA terlebih dahulu yang terdiri dari kunci publik dan kunci privat. Lalu Alice sebagai pengirim pesan akan membuat sebuah Session Key menggunakan sistem kriptografi simetris AES. Tahap berikut nya Alice akan mengenkripsi Session Key yang sudah dibuat tadi dengan Public Key milik Bob yang mana siapapun boleh mengetahuinya. Lalu Alice mengirim Session Key yang sudah dienkripsi tadi oleh Public Key milik Bob dan mengirim nya kepada Bob. Setelah Bob menerima Session Key yang sudah dienkripsi oleh Alice tadi Bob mendekripsi Session Key tersebut dengan Private Key miliknya sehingga Bob dapat mengetahui Session Key yang akan digunakan untuk mengirim pesan nanti. Setelah kedua belah pihak Alice dan Bob sama-sama sudah memiliki Session Key yang akan digunakan maka Alice dan Bob sudah dapat melakukan komunikasi, bertukar pesan atau informasi rahasia secara aman dan cepat dengan memanfaatkan kedua sistem kriptografi simetris dan sistem kriptografi asimetris tadi atau sebutan nya yaitu Sistem Kriptografi Hybrid.

### Bahasa Pemrograman Python

1. 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.

### Aplikasi Desktop

### Exclusive or operation (XOR)

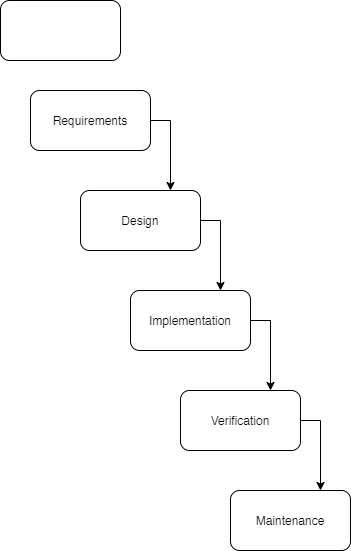
## Metodologi Yang Digunakan

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

Sistem baru yang akan dibuat merupakan implementasi Sistem kriptografi hybrid yang merupakan sistem kriptografi modern terbaru dari 2 sistem sebelum nya yaitu sistem kriptografi simetris dan asimetris. Sistem kriptografi hybrid adalah penggabungan dari kedua sistem sebelumnya yang tujuan nya adalah memanfaatkan kelebihan kedua sistem sebelumnya untuk mengatasi kelemahan kedua sistem tersebut. Sistem ini menggunakan algoritma kriptografi kunci simetris AES (Advanced Encryption System) dan algoritma kriptografi kunci asimetris RSA (Rivest Shamir Adleman).

## 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 terakhir ada 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 menenkrip kunci sesi yang sudah dibuat dan tombol decrypt session key untuk mendekrip kunci sesi yang sudah dienkrip 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

|  |  |  |
| --- | --- | --- |
| No | Aktivitas yang dilakukan | Penjelasan |
| 1. | Analisis kebutuhan perancangan | Pada tahap ini menentukan kebutuhan yang diperlukan bagi pengembangan aplikasi. |
| 2. | Pengujian algoritma kriptografi simetris dan asimetris | Melakukan pengujian pada tiap – tiap algoritma kriptografi |
| 3. | Penggabungan dan perancangan alur sistem | Pada tahap ini melakukan penggabungan kedua algoritma kriptografi dan perancangan alur sistem |
| 4. | Perancangan sistem kriptografi hybrid | Pada tahap ini melakukan perancangan sistem kriptografi hybrid |
| 6. | Penulisan kode program | Pada tahap ini melakukan pengkodean sesuai dengan hasi perancangan |
| 7. | Testing dan perbaikan kode aplikasi | Pada tahap ini melakukan pengecekan terhadap fungsi – fungsi yang terdapat pada aplikasi |
| 8. | Implementasi penggunaan  sistem | Pada tahap ini melakukan implementasi penggunaan sistem kriptografi hybrid |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ke g | Maret | | | | April | | | | Mei | | | | Juni | | | | Juli | | | | Agustus | | | |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| 1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

### 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



Tampilan utama Aplikasi Desktop Hybrid Cryptography berbasis Python



Tampilan menu Session Key Exchange (Pertukaran kunci sesi)



Tampilan menu Generate Session Key (Buat kunci sesi)

Tombol Generate Session Key akan mengenerate kunci sesi baru secara otomatis dan acak.



Tampilan menu Generate Public & Private Key (Buat kunci public dan kunci privat)

Tombol Generate Public & Private Key akan membuat kunci public & kunci privat baru secara otomatis dan acak.



Tampilan Encrypt Session Key (Enkrip kunci sesi)

Masukan kunci sesi ke dalam entry widget session key dan masukan kunci publik milik calon penerima pesan kedalam text widget public key. Klik tombol encrypt maka kunci sesi akan dienkripsi dengan kunci publik secara kriptografi RSA untuk dikirim kepada calon penerima pesan.



Tampilan menu Decrypt Session Key (Dekrip kunci sesi)

Masukan kunci sesi yang telah dienkripsi dengan kunci publik tadi ke dalam text widget encrypted session key dan masukan kunci privat milik penerima pesan ke dalam text widget private key, untuk mendekripsi kunci sesi yang sudah dienkripsi tadi klik tombol decrypt dan hasil kunci sesi yang sudah didekripsi akan ditampilkan di text widget decrypted session key.



Tampilan menu Message Encrypt & Decrypt (Enkripsi dan dekripsi pesan)

Tombol encrypt message akan membuka menu untuk mengenkripsi pesan, sedangkan tombol decrypt message akan membuka menu untuk mendekripsi pesan.



Tampilan menu message encryption (Enkripsi pesan)

Untuk mengenkripsi pesan maka kunci sesi harus dimasukan terlebih dahulu ke dalam entry session key. Setelah itu baru masukan pesan yang akan dienkripsi dan klik tombol encrypt untuk mengenkrip pesan dengan kunci sesi dengan sistem kriptografi AES 128-bit. Hasil enkripsi akan ditampilkan pada text widget encrypted message.



Tampilan menu message decryption (dekripsi pesan)

Untuk mendekripsi pesan maka pertama – tama kunci sesi harus dimasukan terlebih dahulu ke dalam entry widget session key. Lalu pesan yang sudah dienkripsi tadi akan didekripsi dengan kunci sesi dengan sistem kriptografi AES 128-bit ketika tombol decrypt diklik. Hasil pesan yang didekripsi akan ditampilkan pada text widget decrypted message dan pesan asli akan dapat dibaca oleh penerima pesan.

## 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 bahasa Jepang, bahasa 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