LAPORAN PRAKTIK SISTEM KEAMANAN DATA

ALGORITMA RSA

KELOMPOK 3

DISUSUN OLEH

Bagas Aditya pramudana	(V3920012)
Dion Aji cahyono	(V3920018)
Isnan Nur Ahmad Wijayakusuma	(V3920029)
Ivan Fausta Dinata	(V3920030)
Kreshna Pura Adi Wicaksana	(V3920032)

PROGRAM STUDI D-III TEKNIK INFORMATIKA

FAKULTAS SEKOLAH VOKASI

UNIVERSITAS SEBELAS MARET SURAKARTA

2021/2022

JURNAL 1

❖ **Analisa Improvisasi Algoritma RSA Menggunakan RNG LCG pada Instant Messaging Berbasis Socket TCP**

❖ **Latar Belakang**

Socket TCP adalah abstraksi yang digunakan aplikasi untuk mengirim dan menerima data melalui koneksi antar dua host dalam jaringan komputer. Seorang programmer dapat menggunakan koneksi jaringan tersebut untuk menentukan lalu lintas data untuk ditulis dan dibaca. Jaringan yang biasa kita gunakan bersifat publik yang sangat rentan akan penyadapan data. Sehingga untuk mengatasi permasalah ini dapat menggunakan algoritma kriptografi pada socket TCP, salah satunya menggunakan algoritma RSA. Algoritma RSA yang memiliki tingkat keamanan yang berfokus pada sulitnya faktorisasi bilangan besar pada nilai N menjadi 2 bilangan prima (p dan q).

❖ **Tujuan Penelitian**

Tujuan penelitian ini adalah untuk menganalisis perbandingan performa waktu proses dari awal hingga akhir antara algoritma RSA standar dengan improvisasi algoritma RSA menggunakan RNG LCG, serta metode known plaintext attack dan fermat factorization digunakan untuk melakukan pengujian terhadap perbandingan tingkat keamanan antara algoritma RSA standar dengan improvisasi algoritma RSA menggunakan RNG LCG. Penelitian ini diharapkan mampu meningkatkan keamanan dan mempercepat proses enkripsi dan dekripsi algoritma RSA.

❖ **Algoritma yang dipakai beserta alur penelitiannya**

Algoritma yang digunakan adalah algoritma RSA yang termasuk ke dalam algoritma kriptografi kunci asimetris yang memiliki dua buah kunci, kunci publik dan kunci privat. Kunci tersebut terdiri dari bilangan prima yang menjadi penentu tingkat kesulitan pada algoritma RSA. Algoritma ini sangat umum digunakan dalam proses data enkripsi maupun aplikasi digital signature. Bilangan prima menjadi penentu tingkat kesulitan pada algoritma RSA. Metode yang digunakan adalah Waterfall.

A. Implementasi Aplikasi Chat

Aplikasi yang dibangun meliputi aplikasi pada client dan pada server.pesan yang akan dikirim akan dilakukan proses enkripsi terlebih dahulu dengan algoritma RSA atau algoritma RSA improvisasi dan User akan terhubung jika sudah memasukan ip server dan juga username lalu menekan tombol Connect.

B. Algoritma RSA

a) Pembangkitan RSA

Pengkodean RSA membutuhkan dua kunci yang berbeda untuk enkripsi dan dekripsi. Bilangan yang dipilih sebagai kunci adalah bilangan prima yang besar, dengan alasan pemfaktoran sebuah bilangan hasil perkalian dari dua bilangan prima yang besar menjadi dua bilangan prima yang sesuai akan sangat sulit. Berikut Langkah-langkahnya:

1) Membangkitkan bilangan prima p dan q bernilai acak
2) Menghitung nilai N = p $*$ q
3) Menghitung nilai $\Phi(N) = (p - 1) * (q - 1)$
4) Menentukan nilai e dengan rumus gcd $(e, \Phi(N)) = 1$
5) Mencari nilai d yang sesuai dengan $e*d \equiv 1 \bmod \Phi(N)$
6) Hasil dari pembangkitan kunci yaitu kunci publik (e,N) dan kunci privat (d,N).
7) Menghitung ciphertext untuk mengenkripsi plaintext dengan rumus 1 ($C = Me \bmod N$)
8) Menghitung nilai Untuk melakukan dekripsi ciphertext,.($M = Cd \bmod N$)

Pseucode pembangkitan kunci algoritma RSA

Input: panjang bit bilangan prima (α), panjang bit eksponen enkripsi (β)

Output: kunci publik dan kunci privat

1. Bangkitkan dua bilangan prima berukuran α bit (p dan q)
2. $N = p \times q$
3. $\Phi(N) = (p - 1) \times (q - 1)$
4. Pilih bilangan bulat e berukuran β bit dengan gcd(e, $\Phi(N)$) = 1, $1 < e < \Phi(N)$
5. $d = e^{-1} \bmod \Phi(N)$

b) Proses Enkripsi RSA

Menghitung ciphertext untuk mengenkripsi plaintext dengan rumus $C = M^e \bmod N$. Berikut pseucode enkripsi algoritma RSA:

Input: Kunci publik = (e, N), plain text (M)

Output: *ciphertext* (C)

1. for i ← 0 to panjang *plaintext* − 1 do
2. Konversi Mi ke nilai desimal
3. Ci ← (Mi)e mod N
4. end for
5. Return C

c) Proses Dekripsi RSA

Menghitung nilai M untuk mendekripsi ciphertext dengan rumus $M = C^d \bmod N$. Berikut pseucode ekripsi algoritma RSA:

Input: Kunci privat = (d, N), *ciphertex*t (C)

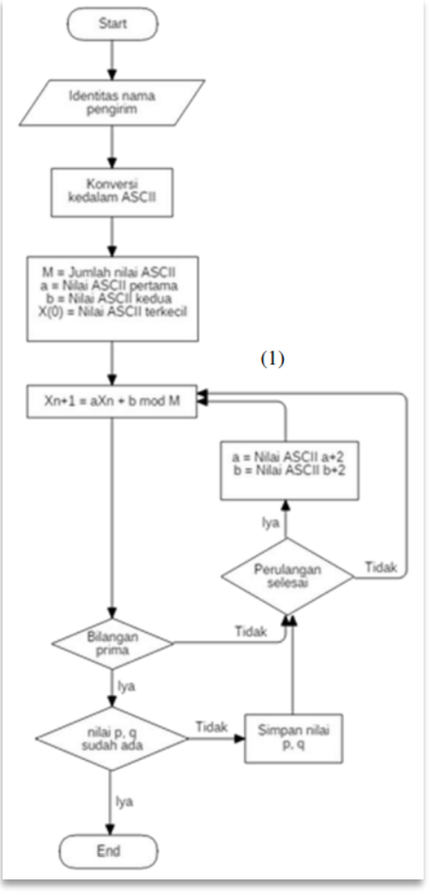
Output: *plaintext* (M)

1. for i ← 0 to panjang *Ciphertext* − 1 do
2. Mi ← (Ci)d mod N
3. Konversi nilai Mi ke karakter
4. end for
5. Return M

C. RNG LCG

Perubahan dilakukan pada isi dari variabel M yang semula didapat dari nilai ASCII terbesar, namum pada penelitian ini variabel M diubah dengan total penjumlahan nilai ASCII dari nama pengirim. Improvisasi dilakukan karena dari analisa hasil yang didapat pada penelitian sebelumnya nilai M yang

kecil menghasilkan bilangan output yang kecil juga. Nilai M pada penelitian ini ditingkatkan jumlahnya yang diharapkan akan meningkatkan hasil dari output yang dihasilkan. Berikut Flowchart dari RNG LCG:



D. Algoritma RSA Improvisasi

RSA improvisasi mendapatkan bilangan prima p dan q dari RNG LCG. Perbedaan lain nya terdapat improvisasi pada rumus $\Phi(N)$. Pseudocode dari proses pembangkitan kunci algoritma improvisasi algoritma RSA menggunakan RNG LCG sebagai berikut:

Input: p, q, panjang bit eksponen enkripsi (β)
Output: Kunci publik, Kunci privat

1. Input 2 bilangan prima p dan q dari proses RNG LCG
2. $N = p \times q$
3. $\Phi(N) = (p^2 - 1) \times (q^2 - 1)$
4. Pilih bilangan bulat e berukuran β bit dengan gcd(e, $\Phi(N)$) = 1, $1 < e < \Phi(N)$
5. $d = e^{-1} \bmod \Phi(N)$
6. Kpublik = (e, N), Kprivat = (d, N)

E. Known Plaintext Attack

Attacker mengetahui kunci publik dan pesan yang terenkripsi. Lalu, attacker akan membentuk baris himpunan ASCII untuk plaintext (P) dan ciphertext (C). Kemudian, Attacker mencocokan ciphertext dengan plaintext apakah terdapat plaintext yang berkorespondensi. Kemudian, Attacker akan menyimpan plaintext jika terdapat plaintext berkorespondensi. Contohnya, Andi memiliki kunci publik (130931,736255235558669442766648479002934564l7) yang dibagikan ke Budi dan Caca. Caca

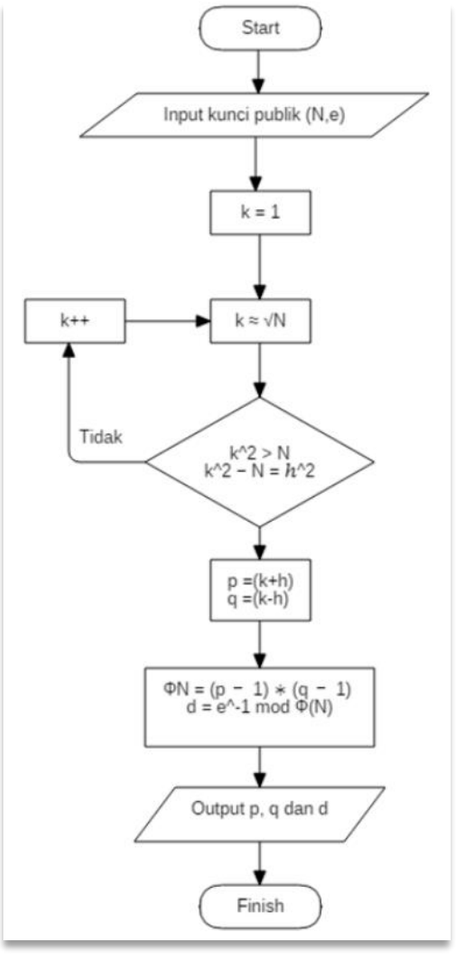
kemudian mengenkripsi himpunan ASCII desimal ke a-z (97 - 122) dengan kunci publik Andi, maka Caca memiliki himpunan data sebagai berikut:

P	C	P	C	P	C
97	20428	106	21175	115	66588
98	99834	107	67021	116	51266
99	130615	108	41582	117	21074
100	60004	109	10647	118	59861
101	100743	110	54014	119	43787
102	114041	111	100756	120	18997
103	246	112	46134	121	28233
104	87120	113	73443	122	58327
105	121279	114	89167		

Kemudian Budi menyadap pesan yang dikirimkan oleh Andi dan Caca dengan mencocokan nilai ciphertext(C) yang dikirimkan dengan tabel diatas. Misal, pesan yang dikirimkan dalam bentuk ciphertext adalah 87120 20428 121279. Ciphertext tersebut dapat dicocokan dengan tabel diatas dimana 87120 = 104 (huruf “h”), 20428 = 97 (huruf “a”) dan 121279 = 105 (huruf “i”) menghasilkan plaintext “hai”.

F. Fermat Factorization

Apabila attacker mengetahui faktor dari nilai N pada Kunci, maka dia akan mengetahui nilai eksponen d yang terdapat pada kunci privat (N,d) yang didapat dari nilai kunci publik (N,e). Lalu, pesan yang telah dienkripsi akan dengan mudah didekripsi oleh attacker tersebut. Berikut flowchart fermat factorization:



❖ **Hasil penelitian pada jurnal tersebut dan kesimpulannya**

RSA menggunakan RNG LCG dapat diimplementasikan pada aplikasi instant messaging socket TCP, improvisasialgoritma RSA menggunakan RNG LCG memiliki performa waktu yang lebih baik disbanding algoritma RSA standar pada proses pembangkitan kunci, enkripsi dan dekripsi, improvisasi algoritma RSA menggunakan RNG LCG lebih aman disbanding algoritma RSA dari serangan known plaintext attack serangan menggunakan metode fermat factorization.

Jaringan yang biasa kita gunakan bersifat publik yang sangat rentan akan penyadapan data. Masalah ini dapat teratasi dengan menggunakan algoritma kriptografi pada socket TCP, salah satunya menggunakan algoritma RSA. Tingkat keamanan algoritma RSA standar memiliki celah keamanan pada kunci public ataupun privat yang berasal dari inputan 2 bilangan prima saat pembangkitan kunci. Beberapa penelitian telah dilakukan untuk mengembangkan algoritma RSA, namun hasil dari penelitian tersebut membuat performa dari algoritma RSA menjadi lebih lambat. Peningkatkan performa dapat menggunakan RNG LCG pada pembangkitan kunci RSA. RNG LCG memiliki kelebihan yang utama pada segi kecepatannya. RNG LCG dapat menghasilkan bilangan prima yang berasal dari inputan nama yang tidak ditemukan pada RNG lainnya. Hasil pengujian performa waktu pembangkitan kunci, enkripsi, dekripsi dengan panjang karakter mulai dari 40 hingga 81920 menunjukkan bahwa algoritma improvisasi RSA menggunakan RNG LCG lebih baik dibandingkan algoritma RSA. Pengujian kemanan menggunakan known plaintext attack dan fermat factorization menunjukkan bahwa algoritma improvisasi RSA menggunakan RNG LCG lebih baik dibandingkan algoritma RSA.

❖ **Kelebihan dan kekurangan masing-masing jurnal**

- **Kelebihan**
  - Menjelaskan alur dari program yang digunakan dengan membuat flowchart dari program tersebut.
  - Memberikan gambaran hasil dan proses saat melakukan enkripsi dengan menampilkan gambar proses.
  - Memberikan perbandingan nilai antara algoritma RSA biasa dengan algoritma RSA menggunakan RNG LCG.

- **Kekurangan**
  - Rumus yang digunakan tidak dijabarkan secara sederhana.
  - Tidak menjelaskan pembuatan atau penggunakan RNG LCG yang lebih detail seperti perhitungannya manual atau penjelasan yang lebih detail

JURNAL 2

❖ **IMPLEMENTASI KEAMANAN PESAN TEKS MENGGUNAKAN KRIPTOGRAFI ALGORITMA RSA DENGAN METODE WATERFALL BERBASIS JAVA**

❖ **Latar Belakang**

Teknologi komunikasi dan informasi sangat berkembang dengan pesat dan memberikan pengaruh besar bagi seluruh kehidupan manusia.proses pengiriman data (pesan) terdapat beberapa hal yang harus diperhatikan, yaitu kerahasiaan, integritas data, autentikasi dan non repudiasi. Oleh karena itu membutuhkan suatu proses penyandian atau pengkodean pesan sebelum dilakukannya proses pengiriman atau delivery.

Kriptografi (Cryptography) berasal dari bahasa Yunani, terdiri dari dua suku kata yaitu kripto dan graphia. Kripto artinya menyembunyikan, sedangkan graphia artinya tulisan. Kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi.

RSA merupakan proses penyandian kunci asimetrik (asymmetric key). Proses perumusan RSA didasarkan pada Teorema Euler, sedemikian sehingga menghasilkan kunci umum dan kunci pribadi yang saling berkaitan. Sehingga meskipun proses enkripsi dan dekripsi menggunakan dua kunci yang berbeda hasilnya akan tetap sama. Kunci umum dan kunci pribadi yang digunakan adalah suatu bilangan prima, dan disarankan bilangan prima yang besar. Hal ini digunakan untuk pencegahan usaha pemecahan teks rahasia, karena semakin besar bilangan prima yang digunakan sebagai kunci maka semakin sulit mencari bilangan besar sebagai faktornya.

❖ **Tujuan Penelitian**

Tujuan dari penelitian ini adalah membuat sebuah program percobaan enkripsi dan dekripsi pesan menggunakan teknik RSA menggunakan bahasa pemrograman Java yang harapannya pesan teks yang asli tidak akan berubah meskipun dalam proses enkripsi dan dekripsi menggunakan kunci yang berbeda.

❖ **Algoritma yang dipakai beserta alur penelitiannya**

Algoritma yang digunakan adalah algoritma RSA yang termasuk ke dalam algoritma kriptografi kunci asimetris yang memiliki dua buah kunci, kunci publik dan kunci privat. Kunci tersebut terdiri dari bilangan prima yang menjadi penentu tingkat kesulitan pada algoritma RSA.

Alur penelitian:

Algoritma RSA ini memiliki 3 tahapan diantaranya pembangkitan kunci, proses enkripsi, proses dekripsi. Berikut ini penjelasan tiap tahapannya:

1. **Pembangkitan Kunci**

Pengkodean RSA membutuhkan dua kunci yang berbeda untuk enkripsi dan dekripsi. Bilangan yang dipilih sebagai kunci adalah bilangan prima yang besar, dengan alasan pemfaktoran sebuah bilangan hasil perkalian dari dua bilangan prima yang besar menjadi dua bilangan prima yang sesuai akan sangat sulit. Berikut Langkah-langkahnya:

1) Membangkitkan bilangan prima p dan q bernilai acak
2) Menghitung nilai N = p $*$ q
3) Menghitung nilai Φ(N) = (p — 1) $*$ (q — 1)
4) Menentukan nilai e dengan rumus gcd (e,Φ(N)) = 1
5) Mencari nilai d yang sesuai dengan e$*$d ≡ 1 mod Φ(N)
6) Hasil dari pembangkitan kunci yaitu kunci publik (e,N) dan kunci privat (d,N).
7) Menghitung ciphertext untuk mengenkripsi plaintext dengan rumus 1 (C = Me mod N)
8) Menghitung nilai Untuk melakukan dekripsi ciphertext,.(M = Cd mod N)

2. **Proses Enkripsi**

Berikut prosesnya:

1) Plaintext diubah ke dalam bentuk bilangan. Untuk mengubah plaintext yang berupa huruf menjadi bilangan dapat digunakan kode ASCII dalam sistem bilangan decimal.

2) Plaintext m dinyatakan menjadi blok-blok x1,x2,x3,...., sedemikian sehingga setiap blok merepresentasikan nilai di dalam selang [0, n-1] , sehingga transformasinya menjadi satu ke satu.

3) Setiap blok mi dienkripsi menjadi blok ci dengan rumus $y_i = {x_i}^{PK} \ mod \ r$

3. Proses Dekripsi

Berikut prosesnya:

1) Setiap blok ciphertext yi didekripsi kembali menjadi blok xi dengan rumus $x_i = {y_i}^{SK} \ mod \ r$

2) Kemudian blok-blok m1,m2,m3,...., diubah kembali ke bentuk huruf dengan melihat kode ASCII hasil dekripsi.

❖ **Hasil penelitian pada jurnal tersebut dan kesimpulannya**

Kesimpulan dan hasil penelitian pada Analisa Improvisasi Algoritma RSA Menggunakan RNG LCG pada Instant Messaging Berbasis Socket TCP bahwa a kriptografi RSA ini cukup aman, karena kunci RSA tidak asal dibuat. Hasil dari kriptografi ini juga cukup membingungkan bagi pembaca yang tidak mempunyai kunci dekripsi, karena yang dihasilkan hanya berupa bilangan bulat.

❖ **Kelebihan dan kekurangan masing-masing jurnal**

- **Kelebihan**
  - Memberikan gambaran langkah-langkah dalam proses melakukan enkripsi dengan menggunakan sebuah software.
  - Memberikan contoh perhitungan kunci yang sudah dipilih menggunakan perhitungan matematika manual.

- **Kekurangan**
  - Penjelasan skema alur skema program enkripsi dan dekripsi dengan Algoritma Kriptografi Asimetris belum terlalu detail dan jelas.