Prosiding Seminar Nasional Integrasi Matematika dan Nilai Islami
Vol.3, No.1, September 2019, hlm. 37-42
p-ISSN: 2580-4596; e-ISSN: 2580-460X

Implementasi *Quasigroup Hybrid Encryption* untuk Mengamankan Soal Ujian

Muhammad Khudzaifah
Jurusan Matematika, UIN Maulana Malik Ibrahim Malang
khudzaifah@uin-malang.ac.id

Info Artikel

Riwayat Artikel:

Diterima: 21 Oktober 2019
Direvisi: 18 November 2019
Diterbitkan: 15 Januari 2020

Kata Kunci:

Quasigrup
Kriptografi
Algoritma Hibrida

ABSTRAK

Pada penelitian ini dibahas penerapan quasigrup di bidang kriptografi. Suatu operasi quasigrup didefinisikan sehingga bisa membentuk suatu algoritma kriptografi yang disebut sebagai *quasigroup cipher*. *Quasigroup cipher* merupakan algoritma kriptografi simetris. Algoritma kriptografi simetris memiliki sistem keamanan lemah karena kunci yang digunakan untuk proses *enciphering* sama dengan kunci yang digunakan untuk proses *deciphering*, sehingga pada penelitian ini algoritma *quasigroup cipher* dimodifikasi dengan menggabungkannya dengan algoritma RSA menjadi suatu algoritma hibrida yang memiki dua tingkatan kunci untuk mengamankan naskah soal ujian. Maka dihasilkan algoritma yang memiliki tingkat keamanan yang baik dan proses *enciphering* serta *deciphering* membutuhkan waktu yang singkat.

Korespondensi:

Muhammad Khudzaifah,
Jurusan Matematika,
UIN Maulana Malik Ibrahim Malang,
Jl. Gajayana No. 50 Malang, Jawa Timur, Indonesia 65144
khudzaifah@uin-malang.ac.id

1. PENDAHULUAN

Seiring dengan perkembangan teknologi informasi, ancaman peretasan keamanan data pun semakin besar. Hampir semua kegiatan dilakukan dengan serba digital, terutama kegiatan akademik mulai kegiatan belajar mengajar (*e-learning*) hingga ujian yang saat ini pemerintah menerapkan ujian dalam bentuk UNBK(Ujian Nasional Berbasis Komputer). Ujian bersifat sangat rahasia, oleh karena itu diperlukan pengamanan yang sangat ketat dari segala tindak bentuk kecurangan, terutama peretasan soal ujian yang bisa mengakibatkan soal ujian bisa bocor.

Sudah sangat banyak algoritma yang dikembangkan untuk mengamankan data, ada algoritma simetris dan asimetris, yang membedakan keduanya adalah kunci untuk mensandikan pesan dan kunci untuk menterjemahkan pesan sama untuk algoritma simetris, sedangkan pada algoritma asimetris mempunyai kunci yang berbeda.

Setiap algoritma memiliki keunggulan dan kelemahan, pada algoritma simetris keunggulannya proses penyandian pesan dan menterjemahkan pesan butuh waktu yang singkat tetapi tingkat keamanannya kurang baik karena kunci untuk mensandikan pesan dan menterjemahkan pesan sama, pada algoritma asimetris keunggulannya tingkat keamanannya baik karena kunci untuk mensandikan pesan dan menterjemahkan pesan berbeda tetapi proses penyandian pesan dan menterjemahkan pesan butuh waktu yang lama.

Jika dianalisis, algoritma simetri apabila diterapkan untuk mengamankan soal ujian maka proses penyandian soal ujian dan menterjemahkan soal ujian membutuhkan waktu yang singkat, akan tetapi tingkat keamanannya kurang baik sehingga rawan peretasan. Jika kita menggunakan algoritma asimetri untuk diterapkan untuk mengamankan soal ujian maka tingkat keamanannya baik, akan tetapi proses penyandian soal

ujian dan menterjemahkan soal ujian membutuhkan waktu yang lama sehingga akan memakan waktu ujian yang cukup banyak.

Oleh karena itu dalam penelitian ini saya ingin mengkombinasikan algoritma simetris dan asimetris, agar menghasilkan algoritma yang efektif dan efisien maka algoritma simetris digunakan untuk menyandikan soal ujian dan algoritma asimetris digunakan untuk mengamankan kunci dari algoritma simetris. Sehingga dihasilkan algoritma hibrida yang memiliki tingkat keamanan yang baik dan proses penyandian soal ujian dan penerjemahan soal ujian membutuhkan waktu yang singkat.

2. METODE PENELITIAN

Metode yang digunakan dalam penelitian ini adalah metode deskriptif melalui literature dan praktikum. Adapun langkah-langkah yang akan dilakukan dalam pelaksanaan penelitian ini adalah Mengkontruksi Kunci Rahasia dalam Bentuk *Quasigroup Encryption*, Mengkontruksi Algoritma (Quasigroup Encryption -RSA) dan Mengkontruksi Program *Quasigroup Encryption* untuk mengenkripsi soal ujian.

3. HASIL DAN PEMBAHASAN

Akan dibahas suatu metode pembentukan kunci rahasia yang menerapkan sifat *quasigroup order* $p-1$ untuk proses *encipher* dan *decipher*. Teori mengenai *quasigroup order* $p-1$ berikut dikutip dari [1][2]

Quasigroup $(Q,*)$ dan k-tuple $(a_1, a_2, \dots, a_k)$ dari leader $a_i \in Q$, sistem $((Q,*),(a_1,a_2,\dots,a_k),E_{a_1\dots a_k},D_{a_k\dots a_1})$ terdefinisi sebagai quasigroup stream cipher atas string di Q^+.

Untuk suatu p bilangan prima dan bilangan K yang memenuhi $1 \le K \le p-2$, fungsi $f_K(j) = \frac{1}{1+(K+j)mod(p-1)} mod\ p$ adalah permutasi dari element di $\mathbb{Z}_p^*$.

Operasi biner $*$ pada himpunan $Q = \{1,2,\dots,p-1\}$ didefinisikan sebagai

$$i * j = i \times f_K(j) mod\ p$$

membentuk *quasigroup* $(Q,*)$.

Jika didefinisikan fungsi $g(i,j,K) = \left((i \times j^{-1} mod\ p) - 1 - K\right) mod\ (p-1)$yang mengambil parameter i,j,K dari himpunan $Q = \{1,2,\dots,p-1\}$, yang memetakan himpunan $A = \{1,2,\dots,p-1\}^3$ ke himpunan $B = \{1,2,\dots,p-2\}$ maka pembagi kiri $(Q,\backslash)$ dari *quasigroup* $(Q,*)$ didefinisikan sebagai :

$$i \backslash j = \begin{cases} g(i,j,K), \text{jika } g(i,j,K) \neq 0 \\ p-1, \text{jika } g(i,j,K) = 0 \end{cases}$$

Mengkonstruksi Algoritma *Quasigroup Cipher*

Dari teori *quasigroup* dibentuk suatu algoritma kriptografi *quasigroup cipher* [3] sebagai berikut

Algoritma Enkripsi

1. Pilih sebarang bilangan bulat K, $1 \le K \le p-1$ yang mana *quasigroup* $(Q,*)$ terdefinisi untuk elemen $\{1,2,\dots,p-1\}$ dengan persamaan $f_K(j)$, dengan p adalah sebarang bilangan prima yang dipilih.
2. Pilih $k \ge 3$ bilangan bulat acak $a_i, i = 1,2,\dots,k$, $1 \le a_i \le p-2$ untuk menjadi *leader* untuk *quasigroup cipher*.
3. Ubah setiap karakter pada pesan m_μ menjadi bilangan bulat pada range $\{1,2,\dots,p-1\}$, dengan μ adalah indeks setiap karakter dari pesan.
4. Secara berulang hitung $m_\mu^i = a_i * m_\mu^{i-1}$, dimana $m_\mu^0 = m_\mu$, $i = 1,\dots,k$ dan $*$ adalah operasi *quasigroup* yang terdefinisi pada $f_K(j)$.
5. $c_\mu = m_\mu^k$ dan update nilai *leader* dengan $a_i = m_\mu^i, i = 1,\dots,k-1$ dan $a_k = 1 + (\sum_{i=1}^{k} m_\mu^i\ mod\ (p-1))$.
6. Didapatkan pesan yang terenkripsi c_μ (*ciphertext*).

Algoritma Dekripsi

1. Inputkan K untuk membuat $(Q,\backslash)$ dan didapatkan sejumlah k *leader*.

2. Secara berulang hitung $c_\mu^k = a_k \backslash c_\mu, c_\mu^i = a_i \backslash c_\mu^{i+1}, i = k-1, \dots, 1$ dan $\backslash$ adalah operasi $quasigroup$.
3. $m_\mu = c_\mu^1$ dan update nilai *leader* dengan $a_i = c_\mu^{i+1}, i = 1, \dots, k-1$ dan $a_k = 1 + (c_\mu + \sum_{i=2}^{k} c_\mu^i \ mod\ (p-1))$.
4. Didapatkan *plaintext* m_μ.

Mengkonstruksi Algoritma Hibrida *(RSA- Quasigroup Cipher)*

Ilustrasi proses *encipher* dan *decipher*

1. A membangkitkan kunci publik dan kunci privat dengan algoritma RSA yang mana kunci publik akan dikirimkan ke B.
2. B mengenkripsi pesan dengan algoritma *quasigroup cipher*, dan mengenkripsi *session key* dari *quasigroup cipher* dengan kunci publik yang diberikan oleh A dengan menggunakan algoritma RSA. Pesan dan key yang telah terenkripsi dikirim ke A.
3. A mendekripsi *session key* dari B dengan menggunakan kunci privat algoritma RSA, lalu mendekripsi pesan dari B dengan menggunakan *session key* yang sudah terdekripsi dengan algoritma *quasigroup cipher* .

Algoritma Hibrida *(RSA- Quasigroup Cipher)*

Algoritma Enkripsi

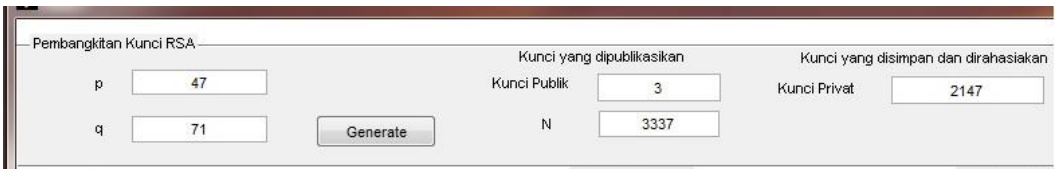
1. Pilih sebarang bilangan bulat K, $1 \le K \le p-1$ yang mana *quasigroup* $(Q,*)$ terdefinisi untuk elemen $\{1,2, \dots, p-1\}$ dengan persamaan pada $f_K(j)$, dengan p adalah sebarang bilangan prima yang dipilih.
2. Enkripsi K dengan algoritma RSA.
3. Pilih $k \ge 3$ bilangan bulat acak $a_i, i = 1,2, \dots, k$, $1 \le a_i \le p-2$ untuk menjadi *leader* untuk *quasigroup cipher* dan enkripsikan dengan algoritma RSA.
4. Ubah setiap karakter pada pesan m_μ menjadi bilangan bulat pada range $\{1,2, \dots, p-1\}$, dengan μ adalah indeks setiap karakter dari pesan.
5. Secara berulang hitung $m_\mu^i = a_i * m_\mu^{i-1}$, dimana $m_\mu^0 = m_\mu$, $i = 1, \dots, k$ dan $*$ adalah operasi *quasigroup*.
6. $c_\mu = m_\mu^k$ dan update nilai *leader* dengan $a_i = m_\mu^i, i = 1, \dots, k-1$ dan $a_k = 1 + (\sum_{i=1}^{k} m_\mu^i \ mod\ (p-1))$.
7. Didapatkan *session key* terenkripsi dan pesan yang terenkripsi c_μ (*ciphertext*).

Algoritma Dekripsi

1. Dekripsi *Session key* dengan algoritma RSA, maka didapatkan K untuk membuat $(Q,\backslash)$ dan didapatkan sejumlah k *leader*.
2. Secara berulang hitung $c_\mu^k = a_k \backslash c_\mu, c_\mu^i = a_i \backslash c_\mu^{i+1}, i = k-1, \dots, 1$ dan $\backslash$ adalah operasi $quasigroup$.
3. $m_\mu = c_\mu^1$ dan update nilai *leader* dengan $a_i = c_\mu^{i+1}, i = 1, \dots, k-1$ dan $a_k = 1 + (c_\mu + \sum_{i=2}^{k} c_\mu^i \ mod\ (p-1))$.
4. Didapatkan *plaintext* m_μ.

Implementasi

Admin jurusan matematika UIN butuh soal Ujian Akhir Semester(UAS). Agar soal tidak bocor maka soal uas akan dikirim dari dosen ke Admin dalam pesan terenkripsi, maka Admin membangkitkan kunci algoritma RSA dan memberitahukan kunci publik kepada para dosen yang digunakan untuk mengenkripsi kunci dari algoritma *quasigroup cipher.* Untuk membangkitkan kunci RSA dipilih sebarang bilangan prima p dan q, pada contoh ini dipilih $p = 47, q = 71$, seperti pada Gambar 1.



Gambar 1. Tampilan program saat pembangkitan kunci algoritma RSA.

Setelah kunci publik diterima, maka para dosen mengenkripsikan soal UAS dengan algoritma *quasigroup cipher* lalu mengenkripsi kan kunci dari *quasigroup cipher* tadi dengan algoritma RSA. Untuk mengenkripsi soal UAS dengan algoritma *quasigroup cipher* pilih sebarang bilangan bulat K, $1 \leq K \leq p-1$, dengan p adalah sebarang bilangan prima yang dipilih (karena *char* yang terdefinisi pada MATLAB sebanyak 127, maka pada program ini menggunakan bilangan prima 127), dan pilih $k \geq 3$ bilangan bulat acak a_i, $i = 1,2,\dots,k$, $1 \leq a_i \leq p-2$ untuk menjadi *leader* untuk algoritma *quasigroup cipher*. Pada contoh ini dipilih $K = 67, k_1 = 78, k_2 = 89, k_3 = 98$. Misalkan soal UAS yang dienkripsi adalah "Jelaskan definisi Grup, Ring, Field dan Daerah Integral beserta contohnya!".

Karakter "J" yang merupakan huruf awal dari kalimat di atas memiliki nilai ASCII 74, seperti perhitungan pada $f_K(j)$

enkrip=$\left(leader(i) * \frac{1}{1+(K+pesan(n))mod(p-1)} mod\ p\right) mod\ p$

- *Leader* ke-1

enkrip=$\left(78 * \frac{1}{1+(67+74)mod(126)} mod\ 127\right) mod\ 127 = 116$

- *Leader* ke-2

enkrip=$(89 * \frac{1}{1+(67+116)mod(126)} mod\ 127) mod\ 127 = 30$

- *Leader* ke-3

enkrip=$(98 * \frac{1}{1+(67+30)mod(126)} mod\ 127) mod\ 127 = 1$

Nilai 1 pada ASCII adalah karakter " "(spasi). Update *leader* 1=116, *leader* 2=30, dan *leader* 3=$1 + \left((116 + 30 + 1) mod\ (126)\right) = 22$

Kemudian huruf setelah "J" adalah huruf "e" yang memiliki nilai ASCII 101

- *Leader* ke-1

enkrip=$\left(116 * \frac{1}{1+(67+101)mod(126)} mod\ 127\right) mod\ 127 = 47$

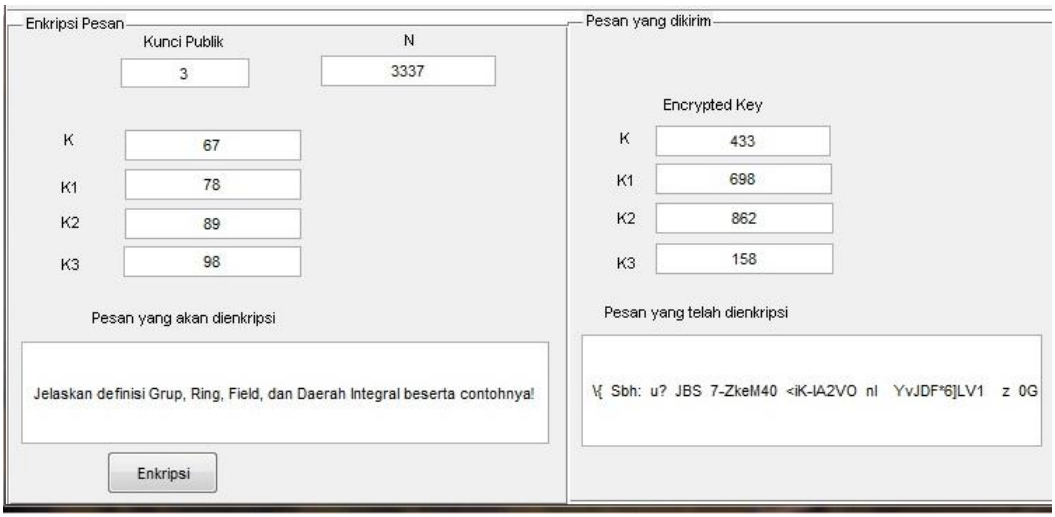
- *Leader* ke-2

enkrip=$(30 * \frac{1}{1+(67+47)mod(126)} mod\ 127) mod\ 127 = 61$

- *Leader* ke-3

enkrip=$(22 * \frac{1}{1+(67+61)mod(126)} mod\ 127) mod\ 127 = 92$

Nilai 92 pada ASCII adalah karakter "\", proses selanjutnya sama hingga karakter terakhir, sehingga didapatkan *ciphertext* seperti pada Gambar 2 dibawah ini.



Gambar 2 Tampilan program saat proses enkripsi.

Setelah soal UAS terenkripsi diterima Admin, maka kunci *quasigroup cipher* dienkripsi terlebih dahulu, lalu digunakan untuk mengenkripsi soal UAS.
Karakter awal dari *ciphertext* spasi memiliki nilai ASCII 1.

- *Leader* ke-1

dekrip=$\big((78 \times 1^{-1} mod\ p) - 1 - 67\big)\ mod\ (126) = 30$

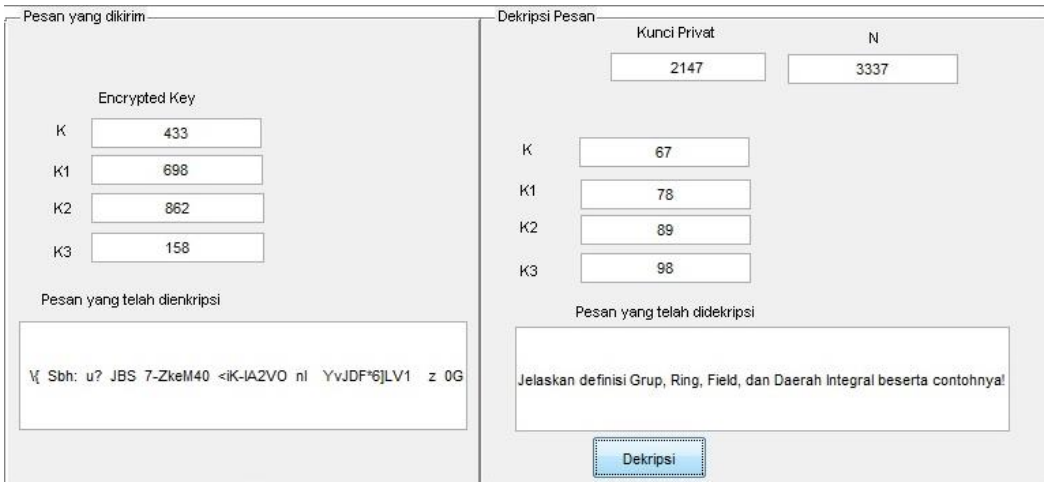
- *Leader* ke-2

dekrip=$\big((87 \times 30^{-1} mod\ p) - 1 - 67\big)\ mod\ (126) = 116$

- *Leader* ke-3

dekrip=$\big((78 \times 1^{-1} mod\ p) - 1 - 67\big)\ mod\ (126) = 74$

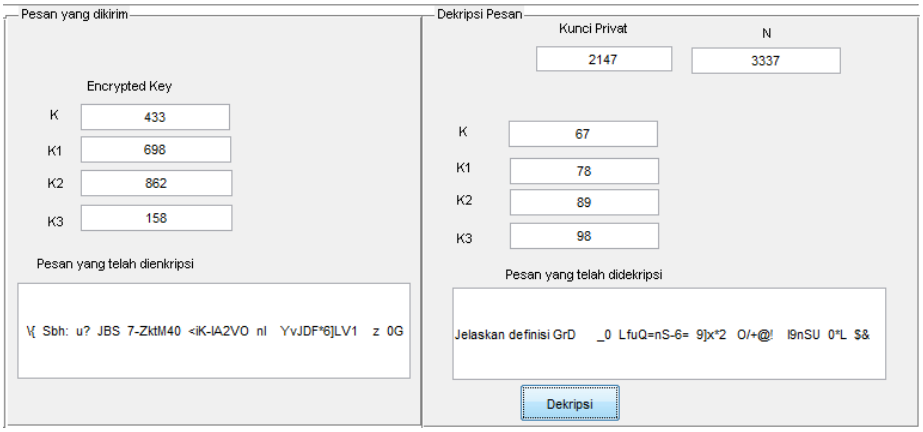
Nilai 74 pada ASCII adalah karakter "J", proses selanjutnya sama hingga karakter terakhir. Sehingga didapatkan *plaintext* seperti pada Gambar 3 dibawah ini.



Gambar 3

Tampilan program saat proses dekripsi.

Algoritma *quasigroup cipher* ini memiliki sistem keamanan cukup bagus, misalkan software pemecah kode memiliki kesalahan baca pada satu huruf saja. Contohnya huruf "e" pada pesan yang terbaca menjadi huruf "t" maka pesan menjadi tak terbaca, seperti pada Gambar 4.



Gambar 4 Tampilan program saat pemecah kode gagal mendekripsi pesan.

4. KESIMPULAN

Berdasarkan hasil pembahasan, maka dapat diambil kesimpulan sebagai berikut.

1. Algoritma kriptografi yang didasarkan dari *quasigroup* yaitu *quasigroup cipher* memiliki keamanan cukup baik. Hal ini dibuktikan pada contoh ketika *software* pemecah kode salah mendekripsi satu huruf saja maka pesan tidak bisa terbaca.
2. Keamanan algoritma *quasigroup cipher* terletak pada enkripsi tiap karakter pesan dilakukan sebanyak *k leader*, dan dilakukan *update leader* setiap pergantian enkripsi ke karakter pesan berikutnya.
3. Kelemahan algoritma *quasigroup cipher* adalah karena kuncinya adalah kunci simetris, sehingga bila kuncinya bocor kepada orang lain, maka pesan bisa dibaca orang lain. Dengan diperkuat algoritma RSA yang memiliki kunci asimetris, maka algoritma *quasigroup cipher* menjadi algoritma hibrida yang mempunyai tingkat keamanan lebih tinggi, karena memiliki 2 tingkatan kunci.

DAFTAR PUSTAKA

[1] S. Markovski, S. Markovski, D. Gligoroski, and S. Andova, "Using Quasigroups for One-One Secure Encoding," *PROC. VIII CONF. Log. Comput. Sci. "LIRA '97", NOVI SAD*, pp. 157--162, 1997.

[2] D. Gligoroski, "Stream cipher based on quasigroup string transformations in Z_p^*," Mar. 2004.

[3] M. Khudzaifah, "Aplikasi quasigroup dalam pembentukan kunci rahasia pada algoritma hibrida," *CAUCHY; Vol 3, No 2 CAUCHYDO - 10.18860/ca.v3i2.2573*, May 2014.