IMPLEMENTASI KODE HAMMING PADA ALGORITMA McELIECE UNTUK MENGAMANKAN PESAN

SKRIPSI

OLEH
M. FAJRUL FALAKH
NIM. 17610114

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2023

IMPLEMENTASI KODE HAMMING PADA ALGORITMA McELIECE UNTUK MENGAMANKAN PESAN

SKRIPSI

Diajukan Kepada
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang
untuk Memenuhi Salah Satu Persyaratan dalam
Memperoleh Gelar Sarjana Matematika (S.Mat)

Oleh
M. FAJRUL FALAKH
NIM. 17610114

PROGRAM STUDI MATEMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2023

IMPLEMENTASI KODE HAMMING PADA ALGORITMA McELIECE UNTUK MENGAMANKAN PESAN

SKRIPSI

Oleh
M. Fajrul Falakh
NIM. 17610114

Telah Diperiksa dan Disetujui Untuk Diuji
Malang, 27 Desember 2023.

Dosen Pembimbing I	Dosen Pembimbing II
Muhammad Khudzaifah, M. Si.	Mohammad Nafie Jauhari, M.Si.
NIP. 19900511 202321 1 029	NIP. 19870218 202321 1 018

Mengetahui
Ketua Program Studi Matematika

Dr. Elly Susanti, M.Sc.
NIP. 19741129 200012 2 005

IMPLEMENTASI KODE HAMMING PADA ALGORITMA McELIECE UNTUK MENGAMANKAN PESAN

SKRIPSI

Oleh
M. Fajrul Falakh
NIM. 17610114

Telah Dipertahankan di Depan Dewan Penguji Skripsi
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Matematika (S.Mat)
Tanggal 30 Desember 2023

Ketua Penguji	: Prof. Dr. H.Turmudi, M.Si, Ph.D.	.....
Anggota Penguji 1	: Hisyam Fahmi, M.Kom.	.....
Anggota Penguji 2	: Muhammad Khudzaifah, M. Si.	.....
Anggota Penguji 3	: Mohammad Nafie Jauhari, M. Si.	.....

Mengetahui,
Ketua Program Studi

Dr. Elly Susanti, M.Sc.
NIP. 19741129 200012 2 005

PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan dibawah ini:

Nama	: M. Fajrul Falakh
NIM	: 17610114
Program Studi	: Matematika
Fakultas	: Sains dan Teknologi
Judul	: Implementasi Kode Hamming pada Algoritma McEliece untuk Mengamankan Pesan.

Menyatakan dengan sebenarnya bahwa skripsi yang saya tulis ini benar-nenar merupakan hasil karya saya sendiri, bukan merupakan pengambil alihan data, tulisan atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila dikemudian hari terbukti atau dapat dibuktikan skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 30 Desember 2023

Yang Membuat Pernyataan

M. Fajrul Falakh

17610114

MOTO

العِلْمُ بِلاَ عَمَلٍ كَالشَّجَرِ بِلاَ ثَمَرٍ

“Ilmu yang tidak diamalkan bagaikan pohon yang tidak berbuah”

-Mahfudzot-

“Kunci selesai adalah mengerjakan, maka jika ingin cepat menyelesaikan skripsi selalu kerjakanlah”

-Jeckyoii-

PERSEMBAHAN

Bismillahirrohmanirrohim

Puji syukur kepada Allah SWT atas segala nikmat yang telah diberikan. Banyak kemudahan dan kelancaran dalam menyelesaikan skripsi ini.

Tak lupa shalawat salam kita ucapkan kepada Nabi Muhammad SAW yang telah menjadi suri tauladan bagi saya dan seluruh umat Islam di dunia.

Dengan menyandarkan harap pada Allah SWT yang Maha Pengasih dan Penyayang, segala puji hanya bagi-Nya atas segala kemudahan dan keberkahan yang telah Dia limpahkan. Shalawat serta salam senantiasa tercurah kepada junjungan kita Nabi Muhammad SAW, teladan utama bagi perjalanan hidup ini. Karya ini kupersembahkan dengan penuh rasa terima kasih dan cinta kepada kedua orang tua tercinta, Bapak Nur Alim Zaini dan Ibu Umi Astutik. Terima kasih tak terhingga atas kasih sayang, doa, dan dukungan tanpa henti yang telah memberikan arah dan kekuatan pada langkah-langkahku. Tidak lupa juga kepada kakak tercinta, Ahmad Zainul Alfan, serta adikku tercinta Akmal Izzul A.M, yang selalu menjadi sumber inspirasi dan dukungan penuh semangat. Tak lupa juga pada Layla Faidatul Hasanah yang telah memberikan bantuan dan dukungan tak ternilai harganya, terima kasih atas segala bentuk bantuannya. Segala doa, nasihat, dan cinta yang kalian curahkan telah membentuk fondasi kuat dalam perjalanan ini. Terima kasih, atas segalanya.

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh.

Segala puji bagi Allah SWT yang telah memberikan berkah dan kemudahan sehingga proposal skripsi ini dapat diselesaikan. Shalawat dan salam senantiasa tercurah kepada Nabi Muhammad SAW, yang telah membimbing umat manusia dari kegelapan menuju cahaya agama Islam. Tidak dapat disangkal bahwa penyelesaian skripsi ini memerlukan upaya keras. Namun demikian, pencapaian ini tidak akan terwujud tanpa dukungan dan bantuan dari berbagai pihak, antara lain:

1. Prof. Dr. H. Zainuddin, M.A., sebagai rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Prof. Dr. Hj. Sri Harini, M. Si., sebagai dekan Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Dr. Elly Susanti, M. Sc., sebagai ketua Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Muhammad Khudzaifah, M.Si., sebagai dosen pembimbing I, yang telah memberikan bimbingan dan arahan berharga dalam proses penyusunan skripsi ini.
5. Mohammad Nafie Jauhari, M. Si., sebagai pembimbing II, yang telah memberikan panduan dan dukungan sepanjang penulisan skripsi.
6. Prof. Dr. H.Turmudi, M.Si, Ph.D., sebagai ketua penguji skripsi, memberikan kontribusi penting dalam proses ujian skripsi.
7. Hisyam Fahmi, M.Kom., sebagai anggota penguji skripsi, turut serta memberikan sumbangan berarti dalam kelancaran proses ujian skripsi.

8. Seluruh staf pengajar Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang, yang telah berbagi pengetahuan dan wawasan.
9. Orang tua penulis, Bapak Nur Alim Zaini dan Ibu Umi Astutik, serta keluarga tercinta, yang memberikan semangat dan dukungan tanpa henti sepanjang perjalanan penulisan skripsi.
10. Teman-teman seangkatan penulis, "MAGENTA" tahun 2017, yang telah memberikan waktu dan pikiran untuk berdiskusi dan berbagi pengetahuan.
11. Semua pihak yang telah membantu langsung maupun tidak langsung dalam menyelesaikan skripsi ini, yang tidak bisa disebutkan satu per satu.

Akhir kata, diharapkan skripsi ini dapat memberikan manfaat dan menambah pemahaman ilmiah bagi pembaca

Wassalamu'alaikum Warahmatullahi Wabarakatuh.

Malang, 30 Desember 2023

Penulis

DAFTAR ISI

DAFTAR GAMBAR

ABSTRAK

Falakh, Muhammad Fajrul. 2023. **Implementasi Kode Hamming pada Algoritma McEliece untuk Mengamankan Pesan**. Skripsi. Program Studi Matematika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing: (I) Muhammad Khudzaifah, M. Si. (II) Mohammad Nafie Jauhari, M.Si.

Kata Kunci : Algoritma McEliece, Kode Hamming, Pembangkit kunci, Enkripsi, Dekripsi.

Kehadiran internet membawa risiko penyadapan yang signifikan, terutama ketika informasi yang dikomunikasikan bersifat rahasia. Jenis informasi ini dapat termasuk pesan teks, gambar, atau data sensitif lainnya. Penggunaan kode Hamming dalam implementasi Algoritma McEliece untuk meningkatkan keamanan pesan dalam komunikasi. Algoritma McEliece adalah salah satu teknik kriptografi kunci publik yang sangat tangguh dan tahan terhadap serangan komputasi kuantum. Sementara kode Hamming adalah salah satu jenis kode koreksi kesalahan yang mampu mendeteksi dan memperbaiki kesalahan pada data yang dikirimkan. terdapat tiga tahapan dalam algoritma McEliece yaitu pembentukan kunci, enkripsi dan dekripsi. Pembangkitan kunci dalam Algoritma McEliece akan dimodifikasi dengan memasukkan mekanisme kode Hamming. Proses enkripsi pesan dilakukan dengan menggunakan algoritma McEliece dengan kode Hamming. Selanjutnya, proses dekripsi pada algoritma McEliece dan koreksi *error* dengan menggunakan kode Hamming untuk mengembalikan pesan ke bentuk aslinya. Dengan menggabungkan Kode Hamming ke dalam Algoritma McEliece, dapat menunjukkan peningkatan keamanan pesan yang dikirimkan dengan memadukan kode Hamming dalam algoritma McEliece.

ABSTRACT

Falakh, Muhammad Fajrul. 2023. **Implementation of the Hamming Code in the McEliece Algorithm to Secure Messages.** Thesis. Mathematics Study Program, Faculty of Science and Technology, Universitas Islam Negeri Maulana Malik Ibrahim Malang. Supervisor: (I) Muhammad Khudzaifah, M. Si. (II) Mohammad Nafie Jauhari, M.Si.

Keywords: McEliece Algorithm, Hamming Code, Key Generator, Encryption, Decryption.

The internet poses a significant eavesdropping risk, especially when communicating sensitive information such as text messages, images, or other confidential data. The use of Hamming code in implementing the McEliece Algorithm aims to enhance message security in communication. The McEliece Algorithm is a robust public-key cryptography technique that is resistant to quantum computing attacks. Meanwhile, Hamming code is a type of error correction code capable of detecting and rectifying errors in transmitted data. The McEliece algorithm consists of three stages: key generation, encryption, and decryption. Key generation in the McEliece Algorithm will be modified by incorporating the Hamming code mechanism. The message encryption process uses the McEliece algorithm with Hamming code. Subsequently, the decryption process in the McEliece algorithm involves error correction using Hamming code to restore the message to its original form. Integrating Hamming code into the McEliece Algorithm can demonstrate an enhancement in the security of transmitted messages by combining Hamming code within the McEliece algorithm.

مستخلص البحث

محمّد فجر الفلاح. ٢٠٢٣. تنفيذ كود هامينج في خوارزمية *McEliece* لأَمْن الرسائل. البحث الجامعي. قسم دراسة الرياضيات، كلية العلوم والتكنولوجيا، جامعة مولانا مالك إبراهيم الإسلامية الحكومية بمالانج. المشرف: (١) محمد خذيفة، الماجستير. (٢) محمد نافع جوهري، الماجستير.

الكلمات المفتاحية: خوارزمية *McEliece*،كود هامينج، توليد المفاتيح، التشفير، فك التشفير.

إن وجود الإنترنت يجلب إلى الخطيرة الكبيرة في التصنيف، وخاصة عندما تكون المعلومات التي يتم نقلها سرية. قد يتضمن هذا النوع من المعلومات المهمّة منها الرسائل النصية أو الصور أو البيانات الحساسة الأخرى. استخدام كود هامينج في تنفيذ خوارزمية *McEliece* لزيادة أمْن الرسائل في الاتصالات. خوارزمية *McEliece* هي أداة لترقية تقنية تشفير في التواصل في قويّة جدًّا ومقاومة لهجمات الحوسبة الكمومية. وأمّا كود هامينج هو نوع من أكواد لتصحيح الأخطاء القادرة على اكتشاف الأخطاء وتصحيحها في بيانات المراسلة. هناك ثلاث مراحل في خوارزمية *McEliece*، وهي تشكيل المفتاح والتشفير وفك التشفير. سيتم تعديل إنشاء المفاتيح في خوارزمية *McEliece* من خلال تضمين آلية كود هامينج .تتم عملية تشفير الرسائل باستخدام خوارزمية *McEliece* مع كود هامينج. وبعد ذلك، تستخدم عملية فك التشفير خوارزمية *McEliece* وتصحيح الأخطاء باستخدام كود هامينج لإعادة الرسالة إلى شكلها الأصليّة. من خلال دمج كود هامينج في خوارزمية *McEliece* ، يمكن إثبات زيادة أمْن الرسائل من خلال دمج كود هامينج في خوارزمية *McEliece*.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Manusia merupakan makhluk sosial yang memerlukan interaksi untuk berkomunikasi satu sama lain. Proses komunikasi manusia telah mengalami evolusi seiring dengan kemajuan teknologi. Pada masa lalu, manusia hanya dapat berkomunikasi secara langsung dalam jarak dekat dengan menggunakan bentuk komunikasi yang sederhana. Namun, dengan hadirnya internet, batasan jarak dan waktu tidak lagi menjadi halangan dalam pertukaran pesan. Internet merupakan suatu jaringan komputer global yang menghubungkan berbagai perangkat di seluruh dunia, memungkinkan komunikasi tanpa batas. Informasi apa pun dapat diakses dengan mudah dan diterima secara instan oleh siapa pun, namun hal ini juga membuka peluang terjadinya penyadapan, terutama ketika informasi yang dikomunikasikan bersifat rahasia. Informasi tersebut bisa berupa pesan teks, gambar, dan lain sebagainya.

Merujuk pada Undang-Undang Nomor 27 Tahun 2022 tentang Perlindungan Data Pribadi (UU PDP). Pelindungan Data Pribadi adalah keseluruhan upaya untuk melindungi Data Pribadi dalam rangkaian pemrosesan Data Pribadi guna menjamin hak konstitusional subjek Data Pribadi. Salah satu solusi untuk mengamankan pertukaran data adalah dengan menerapkan kriptografi.

Dalam dunia Islam kita bisa meneladani sifat sifat Rasul dalam menerapkan kriptografi. Pada diri Rasulullah terdapat teladan yang sempurna untuk ummat islam. dalam surat al ahzab ayat 21 Allah menjelaskan

لَّقَدْ كَانَ لَكُمْ فِى رَسُولِ ٱللَّهِ أُسْوَةٌ حَسَنَةٌ لِّمَن كَانَ يَرْجُواْ ٱللَّهَ وَٱلْيَوْمَ ٱلْـَٔاخِرَ وَذَكَرَ ٱللَّهَ كَثِيرًا ٢١

Artinya: *Sesungguhnya telah ada pada (diri) Rasulullah itu suri teladan yang baik bagimu (yaitu) bagi orang yang mengharap (rahmat) Allah dan (kedatangan) hari kiamat dan dia banyak menyebut Allah.*(Q.S Al-ahzab :21)

Penerapan sifat-sifat wajib Rasul dalam kehidupan manusia, terutama di era digital, dapat dicapai melalui penggunaan kriptografi dalam komunikasi. Kriptografi memastikan integritas pesan, mencerminkan kejujuran (sifat shiddiq), menjamin keamanan dan keandalan informasi (sifat amanah), serta menunjukkan kebijaksanaan dalam mengikuti perkembangan zaman (sifat fathonah). Penggunaan kriptografi juga memperkuat efisiensi dan efektivitas penyampaian pesan, sejalan dengan semangat tabligh.

Kriptografi, sebagai ilmu, berperan penting dalam menjaga keamanan pesan dengan menerapkan proses penyandian. Proses kriptografi melibatkan tiga langkah utama: pembentukan kunci, enkripsi, dan dekripsi. Pada tahap enkripsi, pesan diubah atau diacak agar isinya tidak dapat dengan mudah dipahami oleh pihak yang tidak berwenang. Sementara pada tahap dekripsi, pesan yang telah diacak akan dikembalikan ke dalam bentuk pesan asli yang awalnya dikirim oleh pengirim. Dalam konteks ini, pesan yang asli disebut sebagai plainteks, sedangkan pesan yang telah mengalami proses enkripsi disebut sebagai cipherteks. Melalui langkah-langkah ini, kriptografi memainkan peran penting dalam menjaga keamanan dan kerahasiaan informasi yang dikirim melalui berbagai saluran komunikasi.

Ada dua kelompok utama dalam sistem kriptografi, yakni kriptografi simetris dan kriptografi asimetris. Kriptografi simetris menggunakan kunci yang sama

dalam proses enkripsi dan dekripsi. Beberapa contoh dari sistem kriptografi simetris termasuk Vigenere cipher dan Hill cipher. Namun, sistem kriptografi simetris memiliki kelemahan, yaitu kesepakatan terkait penggunaan kunci yang sama untuk proses komunikasi, serta ketidakamanan dalam pengiriman kunci melalui jalur publik. Sementara itu, Dalam sistem kriptografi asimetris, terdapat dua kunci yang digunakan, yaitu kunci privat dan kunci publik. Kunci privat berperan dalam proses dekripsi, sedangkan kunci publik digunakan dalam proses enkripsi (Munir, 2006).

Sistem kriptografi modern yang banyak digunakan saat ini berfokus pada dua permasalahan utama, yaitu faktorisasi prima dan logaritma diskrit. Meskipun telah terbukti aman dan efektif pada komputer klasik, perkembangan komputer kuantum telah membawa ancaman baru. Komputer kuantum diketahui memiliki kemampuan untuk menyelesaikan permasalahan faktorisasi prima dengan cepat dan relatif mudah, mengancam keamanan sistem kriptografi yang berbasis pada permasalahan ini. Untuk menghadapi tantangan ini, dunia kriptografi mulai mengembangkan sistem kriptografi post-quantum yang secara khusus dirancang untuk tetap aman bahkan dalam era komputer kuantum. Pendekatan ini berfokus pada permasalahan matematika yang dapat dengan mudah dihitung oleh penerima pesan, namun menjadi tugas yang sangat sulit untuk dipecahkan oleh pihak luar. Salah satu konsep terkemuka dalam kriptografi post-quantum adalah sistem kriptografi berbasis kode, di mana sistem McEliece telah muncul sebagai salah satu kandidat paling menjanjikan dalam upaya menjaga keamanan dalam era teknologi komputer kuantum (Ilmiyah, 2019).

Algoritma McEliece adalah sistem kriptografi kunci publik yang didasarkan pada penggunaan kode koreksi kesalahan. Salah satu jenis kode koreksi kesalahan

yang efisien yang digunakan dalam algoritma McEliece adalah kode Hamming (Oktavia, 2023). Kode Hamming adalah jenis kode koreksi kesalahan yang relatif sederhana dan efisien, yang dikembangkan oleh Richard W. Hamming pada tahun 1950. Kode Hamming memungkinkan deteksi dan perbaikan kesalahan dalam data yang dikirim melalui kanal yang mungkin mengalami gangguan atau kesalahan selama transmisi (Hafizhah & Utomo, 2023).

Kode Hamming adalah jenis kode koreksi kesalahan yang dapat mendeteksi dan memperbaiki kesalahan bit tunggal pada data. Dalam implementasi algoritma McEliece, kode Hamming digunakan untuk memperkuat keamanan pesan yang dikirimkan. Dengan menggunakan kode Hamming, pesan yang dikirimkan dapat terlindungi dari kesalahan bit yang mungkin terjadi selama transmisi. Implementasi kode Hamming pada algoritma McEliece melibatkan pembentukan kunci publik yang melibatkan matriks generator dan matriks publik. Matriks publik ini digunakan untuk mengenkripsi pesan sebelum dikirimkan. Selain itu, dalam proses dekripsi, kode Hamming digunakan untuk mendeteksi dan memperbaiki kesalahan bit pada pesan yang diterima (Hafizhah & Utomo, 2023). Dengan mengimplementasikan kode Hamming pada algoritma McEliece, pesan yang dikirimkan dapat diamankan dari serangan dan gangguan yang mungkin terjadi selama transmisi. Kode Hamming memberikan kemampuan untuk mendeteksi dan memperbaiki kesalahan bit tunggal, sehingga meningkatkan keandalan dan keamanan pesan yang dikirimkan (Hafizhah & Utomo, 2023).

Berdasarkan paparan tersebut penulis tertarik untuk mengimplementasikan kode Hamming yang terdapat pada Hafizhah & Utomo (2023) dan algoritma McEliece yang terdapat pada Sinurat & Siagian (2022) sehingga menghasilkan

algoritma kriptografi yang kuat dalam mengamankan pesan Kajian ini dituangkan dalam sebuah skripsi yang berjudul: “Implementasi Kode Hamming pada Algoritma McEliece untuk Mengamankan Pesan”.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, rumusan masalah pada skripsi ini dapat dirumuskan sebagai berikut:

1. Bagaimana proses pembentukan kunci pada algoritma McEliece dengan kode Hamming?
2. Bagaimana proses enkripsi menggunakan kunci algoritma McEliece dengan kode Hamming?
3. Bagaimana proses dekripsi dan koreksi *error* pada algoritma McEliece menggunakan kode Hamming?

1.3 Tujuan Penelitian

Tujuan dari penelitian ini dapat dijelaskan sebagai berikut:

1. Untuk mengetahui langkah-langkah dan parameter yang terlibat dalam proses pembentukan kunci algoritma McEliece dengan kode Hamming.
2. Untuk mengetahui proses enkripsi menggunakan kunci algoritma McEliece dengan kode Hamming.
3. Untuk mengetahui proses dekripsi dan koreksi *error* pada algoritma McEliece menggunakan kode Hamming.

1.4 Manfaat Penelitian

Manfaat dari penelitian ini dijabarkan sebagai berikut:

1. Memahami langkah-langkah dan parameter dalam pembentukan kunci kunci publik algoritma McEliece dengan kode Hamming.

2. Mengetahui proses enkripsi menggunakan kunci algoritma McEliece dengan kode Hamming.
3. Mengetahui proses dekripsi dan koreksi *error* pada algoritma McEliece menggunakan kode Hamming.

1.5 Batasan Penelitian

Batasan masalah yang akan dibahas dalam penelitian ini telah diuraikan sebagai berikut:

1. Penelitian ini akan membatasi diri pada algoritma kunci publik pada algoritma McEliece dengan kode Hamming, dengan penekanan pada proses pembentukan kunci, enkripsi, dekripsi, dan koreksi kesalahan menggunakan kode Hamming.
2. Penelitian akan mengeksplorasi aspek pembentukan kunci, enkripsi, dekripsi, dan koreksi kesalahan pada algoritma McEliece menggunakan kode Hamming.
3. Fokus khusus pada peran kode Hamming dalam mendeteksi dan memperbaiki kesalahan pada pesan yang dienkripsi dengan McEliece.

1.6 Daftar Istilah

1. Enkripsi atau *Encoding*

   Enkripsi atau *encoding* adalah suatu cara atau metode dalam teori koding yang mengubah suatu data asli menjadi kode-kode yang melambangkan data tersebut (Munir, 2006).

2. Dekripsi atau decoding

   Dekripsi atau *decoding* atau merupakan suatu proses kebalikan dari *encoding*. Pengertian decoding yaitu suatu cara atau metode dalam teori

koding yang mengubah kode-kode data tersebut menjadi data asli (Munir, 2006).

3. Plainteks atau *Plaintext*

   Plaintext merupakan teks asli yang masih mudah dipahami dan belum dienkripsi.

4. *Ciphertext*

   Ciphertext merupakan teks yang telah berubah strukturnya karena sudah dienkripsi dengan menggunakan suatu kunci tertentu.

BAB II

KAJIAN TEORI

2.1 Struktur Aljabar

2.1.1 Lapangan

Definisi (Ling & Xing, 2004) Lapangan adalah himpunan elemen F yang tak kosong dengan dua operasi '+' (penjumlahan) dan '.' (perkalian) memenuhi persamaan. Untuk $a, b, c \in F$

1. $a + b$ dan $a.b$ berada di F(F tertutup untuk (+) dan (.))
2. $a + b = b + a$, $a.b = b.a$ (Komutatif)
3. $(a + b) + c = a + (b + c)$, $(a.b).c = a.(b.c)$(Assosiatif)
4. $a.(b + c) = a.b + a.c$ (Distributif)

Selanjutnya, dua elemen identitas berbeda 0 dan 1 (disebut penjumlahan dan perkalian identitas, masing-masing) harus ada di F yang memenuhi hal berikut:

5. $a + 0 = a$ untuk semua $a \in F$
6. $a.1 = a$ dan $a.0 = 0$ untuk semua $a \in F$
7. untuk setiap a di F, terdapat elemen invers penjumlahan $(-a)$ di F seperti $a + (-a) = 0$
8. Untuk setiap $a \neq 0$ di F, terdapat elemen invers perkalian a^{-1} di F sehingga $a \cdot a^{-1} = 1$

2.1.2 Lapangan Hingga

Suatu lapangan yang memuat elemen sebanyak berhingga disebut lapangan berhingga. Lapangan hingga yang memuat sebanyak q elemen dilambangkan dengan F_q.

2.1.3 Ruang Vektor atas Lapangan Hingga

Definisi (Ling & Xing, 2004) Misalkan F_q adalah lapangan hingga orde q. sebuah himpunan tak kosong V , bersama dengan beberapa penjumlahan (vektor) + dan perkalian skalar dengan elemen dari F_q , adalah ruang vektor (atau ruang linier) di atas F_q jika memenuhi semua ketentuan berikut. Untuk semua $u, v, w \in V$ dan untuk semua , $\mu \in F_q$:

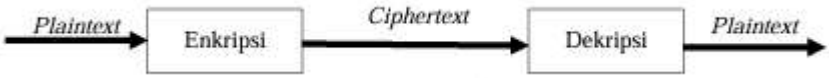
1. $u + v \in V$
2. $(u + v) + w = u + (v + w)$
3. terdapat elemen $0 \in V$ dengan sifat $0 + v = v = v + 0$ untuk semua $v \in V$
4. untuk setiap $u \in V$ terdapat sebuah elemen V yang disebut $-u$, sehingga $u + (-u) = 0 = (-u) + u$
5. $u + v = v + u$
6. $\lambda v \in V$
7. $\lambda(u + v) = \lambda u + \lambda v, (\lambda + \mu)u = \lambda u + \mu u$
8. $(\lambda\mu)u = \lambda(\mu u)$
9. jika 1 adalah identitas perkalian dari F_q , maka $1u = u$

2.2 Kriptografi

Kriptografi (cryptography) berasal dari bahasa yunani yang terdiri dari dua suku kata yaitu kripto dan graphia. Kripto berarti menyembunyikan, sedangkan graphia memiliki arti tulisan. Kriptografi adalah ilmu yang mempelajari teknikteknik matematika yang berhubungan dengan aspek keamanan informasi, seperti kerahasiaan data, integritas data, serta autentikasi data.(Riyanto, 2007). Contoh kriptografi dalam kehidupan sehari-hari diantaranya adalah transaksi

melalui ATM, pay television, komunikasi dengan telepon selular, barcode dan sebagainya. (Munir, 2004).

Fungsi-fungsi mendasar dalam kriptografi adalah enkripsi dan dekripsi. Enkripsi adalah proses mengubah suatu pesan asli (*plaintext*) menjadi suatu pesan dalam bahasa sandi (*ciphertext*). Enkripsi bisa diartikan dengan cipher atau kode, dimana pesan asli (*plaintext*) diubah menjadi kode-kode tersendiri sesuai metode yang disepakati kedua belah pihak, baik pengirim maupun penerima (Prerna et al, 2014). Sedangkan dekripsi adalah proses mengubah pesan dalam suatu bahasa sandi (*ciphertext*) menjadi pesan asli (*plaintext*) kembali. Urutan proses kriptografi dapat dilihat pada Gambar berikut :



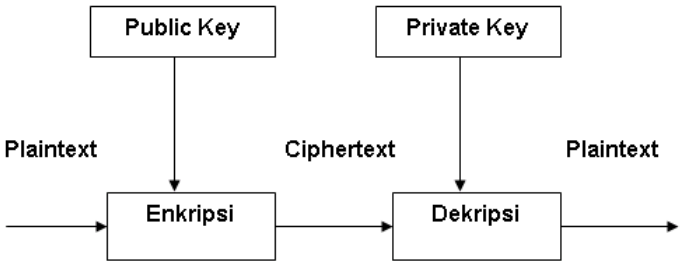
Gambar 2.1 Proses enkripsi dekripsi pesan

Sistem kriptografi adalah algoritma, seluruh kemungkinan plaintext, ciphertext dan kunci. P adalah notasi yang digunakan untuk plaintext, C adalah ciphertext, E adalah fungsi enkripsi dan D adalah fungsi dekripsi (Schneier, 1996)

2.3 Algoritma McEliece

Kriptografi McEliece adalah sistem kunci publik kripto tertua yang menggunakan *error correcting codes*. Sama seperti semua yang memiliki kunci publik, Sistem kripto, sistem ini terdiri dari 3 algoritma:

1. Pembangkitan kunci generator,
2. Enkripsi (menggunakan kunci publik) dan
3. Dekripsi (menggunakan kunci privat).



Gambar 2.2 Algoritma McEliece

2.3.1 Pembentukan Kunci Generator

Pada algoritma McEliece, proses pembangkitan kunci memiliki peran sentral dalam membentuk kunci publik dan privat yang esensial untuk sistem kriptografi ini. Dengan rumusan matematis

$$G' = S \times G \times P \quad \text{.....(1)}$$

Dimana G adalah matriks generator awal dengan ukuran $k \times n$, S adalah matriks non singular acak dengan ukuran $k \times k$, P adalah matriks permutasi berukuran $n \times n$ yang memiliki nilai 1 disetiap kolom dan barisnya, dan dalam hal ini menciptakan matriks G' dengan ukuran $k \times n$. Langkah-langkah ini menyiratkan transformasi kompleks yang menghasilkan kunci publik yang kuat dan aman. Matriks generator G, yang menciptakan ruang kunci, digabungkan dengan matriks non singular acak S, yang memberikan unsur tersembunyi pada kunci. Matriks permutasi P kemudian diterapkan untuk meningkatkan keamanan dan mengacaukan pola matriks yang mendasari (Oktavia, 2023).

2.3.2 Pengenkripsian Pesan

Dalam algoritma McEliece, proses enkripsi adalah langkah penting untuk mengamankan pesan. Dengan menggunakan persamaan

$$C_i = m_i \times G' \text{ .....(2)}$$

Dimana C_i adalah *ciphertext* yang dihasilkan dengan panjang n bit, m_i adalah pesan teks yang akan dienkripsi dengan panjang k bit, dan G' adalah matriks kunci publik hasil dari proses pembangkitan kunci yang berukuran $k \times n$, algoritma ini memetakan pesan teks menjadi bentuk terenkripsi. Matriks G' bertanggung jawab untuk mengubah pesan menjadi ruang kunci kriptografi, menghasilkan *ciphertext* yang memiliki panjang n bit, sesuai dengan panjang kunci yang ditentukan. Proses ini melibatkan perkalian matriks yang kompleks, yang membuatnya tangguh terhadap serangan kriptanalisis yang umum. Namun, untuk mempertimbangkan potensi kesalahan yang mungkin terjadi selama transmisi atau penyimpanan, matriks *error* e dengan panjang n bit ditambahkan ke dalam *ciphertext*, menghasilkan *ciphertext* yang dimodifikasi

$$C_i' = C_i + e \qquad \text{.....(3)}$$

Penambahan *error* ini memungkinkan algoritma McEliece untuk mengatasi gangguan dan memberikan ketahanan terhadap kesalahan, menjadikan sistem ini kuat dalam mengamankan pesan selama proses komunikasi (Oktavia, 2023).

2.3.3 Pendekripsian Pesan

Pada tahap pendekripsian algoritma McEliece, kunci privat berperan penting dalam mengembalikan *ciphertext* menjadi pesan asli. Dalam konteks ini, langkah kunci dekripsi melibatkan perkalian matriks invers dari matriks permutasi P dan matriks non singular acak S, yaitu P invers dan S invers, dengan $ciphertext$ yang diperoleh sebelumnya. Dengan rumusan matematis

$$y_i = C_i' \times P^{-1} \qquad \text{.....(4)}$$

Dimana C' adalah *ciphertext* yang telah ditambahkan dengan matriks *error* yang berukuran n bit. Matriks invers P adalah invers dari matriks permutasi yang mana elemen kunci privat yang diperlukan dalam proses dekripsi. Kemudian binary y_i merupakan hasil dari *standard form generator* matriks yang mana k bit dari kiri merupakan pesan informasi atau data bit, dan $(n-k)$ bit dari kanan merupakan *parity* bit. Langkah selanjutnya adalah dengan mengambil k bit binary y_i dari kiri untuk dikalikan dengan S invers.

$$m_i = y_i \times S^{-1} \quad \text{.....(5)}$$

Melalui proses ini, pesan yang dienkripsi dapat direkonstruksi secara akurat dan efisien. Penggunaan P invers dan S invers memainkan peran vital dalam mengatasi efek dari matriks permutasi dan matriks non singular acak yang diterapkan selama proses enkripsi, sehingga memungkinkan sistem untuk mengembalikan pesan ke bentuk aslinya (Oktavia, 2023).

2.4 Kode Linear

2.4.1 Pengertian Kode Linear

Definisi (Ling & Xing, 2004) Sebuah kode linear C dengan panjang n di atas F_q merupakan subruang vektor dari ruang vektor F_q^n. Dengan F_q^n adalah himpunan semua vektor dengan panjang n yang entri-entrinya merupakan elemen dari F_q

$$F_q^n = \{(v_1, v_2, \dots, v_n);\ v_i \in F_q\}$$

Kode (n, k) adalah kode linear C dengan panjang n dan dimensi k di atas lapangan F_q.

2.4.2 Matriks Generator

Definisi (Ling & Xing, 2004) Matriks generator G untuk kode $C(n,k)$ adalah matriks berukuran $k \times n$ yang memiliki baris-baris yang membentuk basis ruang vektor C.

Kata kode dari suatu kode linear C di atas lapangan F dengan matriks generator G adalah semua kombinasi linear (di atas F) dari baris-baris G. Kode C disebut sebagai kode yang dibangun oleh matriks G. Melakukan operasi baris elemen pada G akan menghasilkan matriks yang juga membangun C. Jika ditemukan matriks generator berbentuk $G = [I_k\ X]$, di mana I_k adalah matriks identitas berukuran $k \times k$ dan X adalah suatu matriks berukuran $k \times (n - k)$, maka simbol informasi terletak pada posisi pertama k dari sebuah kata kode. Matriks G dengan bentuk tersebut disebut matriks generator dengan bentuk standar. Meskipun tidak dapat dijamin bahwa akan selalu ada G untuk C, menukar posisi koordinat dari suatu kode C akan menghasilkan ruang bagian C' yang memiliki bobot dan jarak Hamming yang sama dengan C. Oleh karena itu, C dan C' merupakan kode yang sama. Hal ini mendorong adanya definisi tambahan, di mana matriks permutasi adalah suatu matriks identitas yang baris atau kolomnya telah ditukar.

2.4.3 Matriks Parity Check

Matriks parity check adalah matriks yang digunakan dalam teori kode linear untuk mendeteksi dan memperbaiki kesalahan dalam pesan yang diterima. Matriks ini memainkan peran penting dalam proses dekode pesan dalam kode linear.

Definisi (Ling & Xing, 2004) Matriks *parity check* H untuk suatu kode linear C adalah matriks generator untuk kode dual $C^{\perp}$. Matriks parity check disebut sebagai *standar form* jika memiliki *form* $[Y\ In-k]$ dimana $Y = X^T$.

2.5 Kode Hamming

Definisi (Hamming, 1986; Lidl dan Pilz, 1998) Kode biner C_m dengan panjang $n = 2^m - 1$. $m \geq 2$, yang menggunakan matriks *parity check* H berukuran $m \times (2^m - 1)$ dengan semua kolomnya non-nol disebut kode Hamming.

Setiap dua kolom di H saling bebas secara linear. Namun, tidak setiap tiga kolom di H saling bebas linear, sehingga dimensi minimum jarak (mld) dari H adalah 3. Ini berarti kode Hamming ini adalah kode linear $(2^m - 1, 2^m - 1 - m, 3)$, yang dapat mendeteksi *error* ≤ 2 dan dapat mengoreksi 1 kesalahan. Untuk kode Hamming biner, dapat digunakan metode khusus dalam menentukan kesalahan dan koreksi. Misalkan kolom-kolom H diurutkan sedemikian rupa sehingga kolom pertama adalah representasi biner dari i. Jika y diterima, dan $S(y_i) = H.y^T.S(y_i)$ adalah kolom ke-i dari H, maka kesalahan terjadi pada kolom ke-i dari y.

2.5.1 Jarak dan Bobot Hamming

Definisi (Ling & Xing, 2004) Misalkan x dan y adalah kata yang panjangnya n di atas alfabet A. jarak (Hamming) dari x ke y, dilambangkan dengan $d(x,y)$, didefinisikan sebagai banyaknya tempat di mana x dan y berbeda. jika $x = x_1, x_2, \cdots, x_n$ dan $y = y_1, y_2, \cdots, y_n$, maka

$$d(x,y) = d(x_1,y_1) + d(x_2,y_2) + \cdots + d(x_n,y_n)$$

dimana $x_i$dany_i dianggap sebagai kata dengan panjang 1, dan

$$d(x_i, y_i) = \begin{cases} 1\ jika\ x_i \neq y_i \\ 0\ jika\ x_i = y_i \end{cases}$$

Definisi (Ling & Xing, 2004) Misalkan x menjadi sebuah kata di F . Bobot (Hamming) dari x, dilambangkan dengan (x), didefinisikan sebagai banyaknya koordinat bukan nol dalam x

$$wt(x) = d(x, 0)$$

Untuk setiap elemen x dari F_q , kita dapat menentukan bobot Hamming sebagai berikut

$$wt(x) = d(x, 0) = \begin{cases} 1\ jika\ x \neq 0 \\ 0\ jika\ x = 0 \end{cases}$$

2.6 Kajian Islam

Rasulullah Muhammad SAW adalah teladan yang utama dan sempurna bagi umat Muslim. Setiap tindakan, perkataan, dan sikap beliau merupakan contoh yang baik untuk diikuti oleh mereka yang berharap kepada rahmat Allah dan hari kiamat. Mereka yang mengingat Allah dengan banyak dzikir dan tafakur akan mendapatkan petunjuk dari suri tauladan yang mulia ini. Ayat ini menunjukkan pentingnya mengambil Rasulullah sebagai contoh dalam kehidupan sehari-hari dan sebagai sumber inspirasi untuk menjalankan ajaran Islam dengan baik

لَّقَدْ كَانَ لَكُمْ فِى رَسُولِ ٱللَّهِ أُسْوَةٌ حَسَنَةٌ لِّمَن كَانَ يَرْجُواْ ٱللَّهَ وَٱلْيَوْمَ ٱلْـَٔاخِرَ وَذَكَرَ ٱللَّهَ كَثِيرًا ٢١

Artinya: *Sesungguhnya telah ada pada (diri) Rasulullah itu suri teladan yang baik bagimu (yaitu) bagi orang yang mengharap (rahmat) Allah dan (kedatangan) hari kiamat dan dia banyak menyebut Allah*.(Q.S Al-ahzab :21)

Para Nabi dan Rasul, sebagai manusia terbaik yang dipilih oleh Allah, memikul tanggung jawab berat dalam menjalankan misi ilahi. Dengan perilaku yang sangat

terpuji, mereka menunjukkan sifat wajib yang mencirikan kesucian karakter mereka, terhindar dari perbuatan dosa, dan senantiasa hidup dalam ketakwaan. Keseluruhan kehidupan mereka dipenuhi dengan tindakan dan sikap yang terpuji, mencerminkan ketetapan dalam iman dan ketaatan yang mendalam pada ajaran Allah. ada 4 sifat wajib yang dimiliki oleh Nabi dan Rasul diantaranya:

1. Shiddiq

   Sifat wajib pertama yang harus dimiliki oleh para rasul adalah sidiq, yang berarti jujur dan tidak mungkin mereka memiliki sifat kebalikannya, yaitu sifat kadzib atau berdusta. Makna dari sifat ini adalah bahwa semua ajaran yang disampaikan oleh para utusan Allah adalah benar, dan tidak mungkin terdapat dusta atau kebohongan di balik ajaran-ajaran yang mereka sampaikan. Oleh karena itu, bagi para rasul, memiliki sifat sidiq adalah suatu kewajiban yang tak terelakkan

2. Amanah

   Para rasul wajib memiliki sifat amanah, yang artinya mereka dapat dipercaya dan tidak mungkin melakukan khianat. Sifat ini melindungi mereka secara fisik dan spiritual dari tindakan yang bertentangan dengan prinsip-prinsip syariat. Mereka menjauhi perbuatan zina, menghindari minuman keras, tidak berbohong, tidak merasa iri atau dengki, tidak bersikap sombong, dan tidak tergoda oleh pujian atau celaan. Keberadaan sifat amanah menjadi landasan yang menjamin integritas dan ketulusan para rasul dalam menyampaikan risalah Allah kepada umat manusia.

3. Fathonah

Para rasul, sebagai utusan Allah, senantiasa membawa sifat fatanah, yaitu kecerdasan, kebijaksanaan, dan kepandaian yang sangat penting. Mereka tidak hanya memahami berbagai permasalahan kompleks yang dihadapi umat manusia, tetapi juga memiliki keterampilan untuk memberikan solusi yang tepat. Peran mereka bukan hanya sebagai pembimbing spiritual, tetapi juga sebagai penyuluh yang mampu memberikan panduan dan solusi bijaksana untuk mengatasi tantangan kehidupan. Sifat fatanah ini menjadi landasan kokoh dalam menjalankan tanggung jawab mereka sebagai utusan Allah SWT kepada umat manusia.

4. Tabligh

Tugas utama para rasul adalah tablig, yaitu menyampaikan wahyu dari Allah kepada umat manusia. Dalam menjalankan misi kerasulannya, mereka memiliki tanggung jawab untuk menyampaikan wahyu tersebut, termasuk pengetahuan, syariat, pedoman, dan berbagai risalah kenabian. Para rasul meyakinkan umat manusia untuk mempercayai dan memahami wahyu tersebut sebagai bagian integral dari keimanan mereka. Meskipun kadang wahyu yang disampaikan tidak mudah atau bahkan tidak menyenangkan, para rasul senantiasa menjalankan tugas tablig mereka tanpa mengurangi satu huruf pun. Kesetiaan dan kepatuhan mereka terhadap misi ini mencerminkan integritas tinggi dalam menjalankan peran sebagai utusan Allah.

BAB III

METODE PENELITIAN

3.1 Jenis Penelitian

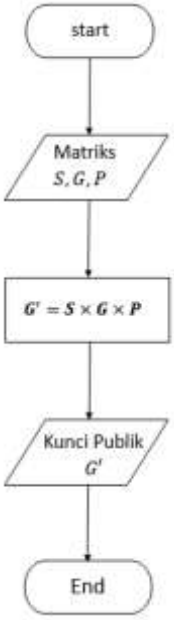
Penelitian yang dilakukan dalam tugas akhir ini menggunakan metode literatur. Penelitian ini fokus pada pembahasan dan penjelasan mengenai materi yang diambil dari buku-buku dan jurnal-jurnal terkait.

3.2 Tahapan Penelitian

Tahapan penerapan kode Hamming dalam algoritma McEliece untuk mengamankan pesan dapat dibagi menjadi beberapa bagian penting. Dimulai dari pembangkitan kunci yang digunakan pada proses enkripsi, kemudian dilanjutkan dengan proses dekripsi untuk menghasilkan *ciphertext.* Tahap terakhir adalah proses dekripsi dan pengkoreksian *error.*

3.2.1 Pembangkitan Kunci

Langkah-langkah dalam pembangkitan kunci dijelaskan pada flowchart berikut.

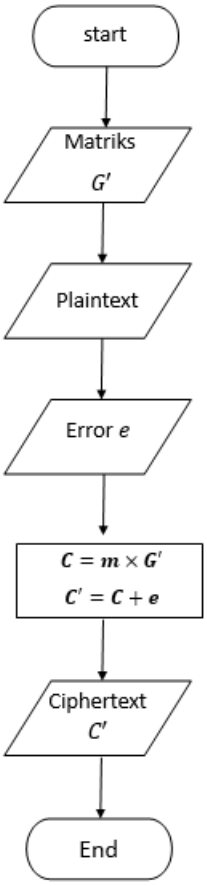


Gambar 3.1 Flowchart proses pembentukan kunci

1. Pengirim memilih sebuah kode linear (n, k) yang memiliki kemampuan memperbaiki *error*, dalam penelitian ini kita menggunakan kode Hamming.
2. Bangkitkan matriks generator berukuran $k \times n$.
3. Pengirim secara acak memilih matriks invertibel S biner dengan ukuran $k \times k$.
4. pengirim secara acak memilih matriks permutasi P dengan ukuran $n \times n$.
5. Pengirim menghitung matriks G' dengan ukuran $k \times n$. $G' = S \times G \times P$

3.2.2 Pengenkripsian Pesan

Langkah-langkah dalam pengenkrpsian pesan dijelaskan pada flowchart berikut

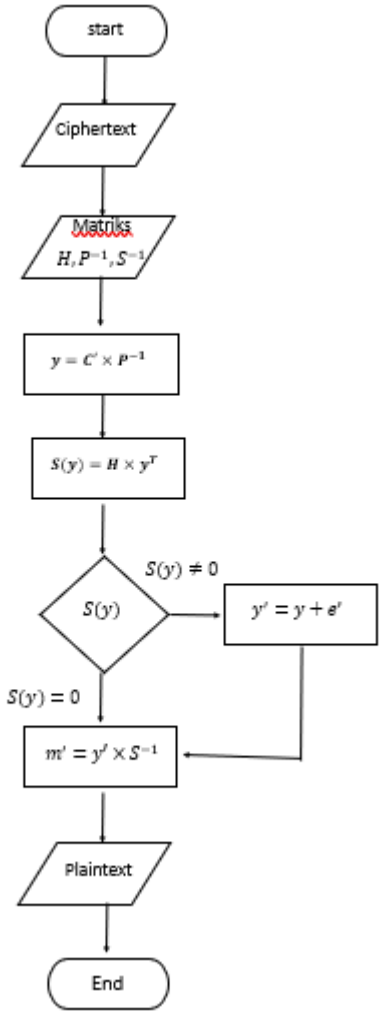


Gambar 3.2 Flowchart proses enkripsi algoritma McEliece

1. Pengirim mengubah *plaintext* ke bentuk binary berdasarkan tabel ASCII.
2. Pengirim membagi binary ke dalam blok blok dengan panjang 4-bit
3. Pengirim menghitung vektor $C_i = m_i \times G'$.
4. Pengirim secara acak menghasilkan vektor e berukuran n-bit yang memiliki t elemen non-nol (vektor dengan panjang n dan bobot t).
5. Pengirim menghitung ciphertext $C_i' = C_i + e$.

3.2.3 Pendekripsian Pesan dan Pengkoreksian *Error*

Langkah-langkah dalam pendekripsian dan pengoreksian *error* dijelaskan pada flowchart berikut



Gambar 3.3 Flowchart proses pengenkripsian dan pengkoreksian *error*

1. Penerima menghitung invers dari matriks P.
2. Penerima menghitung $y_i = C_i'.P^{-1}$.
3. Penerima menghitung matriks *parity check* H.
4. Kemudian penerima memeriksa *error* dengan mencari *syndrome* dari y_i ($S(y_i) = H.y^T$). jika $S(y) = 0$ maka tidak terjadi *error* pada y_i, jika $S(y_i) \neq 0$ ini menujukkan terjadi *error* pada y_i.

5. Selanjutnya penerima mencari letak bit yang *error* dengan melihat matriks H. Jika $S(y_i)$ sama dengan kolom ke-i pada matriks H, maka *error* terjadi pada bit ke-i dari biner y_i.
6. Tahap berikutnya adalah mengoreksi pesan y_i dengan menambahkan *error* e dengan bobot 1 pada bit yang salah. $y_i' = y_i + e'$
7. Setelah y_i dikoreksi dan menghasilkan y_i', kemudian mengambil 4-bit dari kiri pada binary y_i'.
8. Penerima menghitung invers S.
9. Penerima mengoperasikan $y_i'.S^{-1}$. Didapatkan pesan $m_i' = y_i'.S^{-1}$
10. Penerima menggabungkan blok-blok sesuai urutan sehingga memiliki panjang 8-bit.
11. Pesan diubah ke bentuk teks berdasarkan tabel ASCII.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Pembentukan Kunci

Algoritma pembentukan kunci pembangkit pada algoritma McEliece memegang peran krusial dalam memastikan keamanan dan keefektifan sistem kriptografi ini. Dengan langkah awal menentukan nilai n dan k, matriks generator G terbentuk sebagai fondasi dari kunci publik. Keunikan algoritma McEliece terletak pada kemampuannyamemanfaatkan matriks permutasi acak P dan matriks non singular S yang dibangkitkan secara acak untuk membentuk kunci tambahan G'. Proses ini menciptakan kekacauan struktural yang memperkuat keamanan kunci publik, menjadikan algoritma McEliece sebagai pendekatan yang tangguh dalam melindungi pesan dari ancaman kriptanalisis.

4.1.1 Algoritma Pembentukan Kunci pada Algoritma McEliece

1. Input:
   - Menentukan nilai n dan k (dalam contoh, $n = 7$ dan $k = 4$).
   - Bangkitkan matriks generator G dengan ukuran ($n \times k$).
   - Bangkitkan matriks (S) secara acak dengan dimensi ($k \times k$).
   - Bangkitkan matriks (P) secara acak dengan dimensi ($n \times n$).
2. Proses:
   - Hitung matriks (G') menggunakan rumus ($G' = S \times G \times P$).
3. Output:
   - Matriks (G') yang akan digunakan sebagai matriks generator pada tahap enkripsi.

4.1.2 Simulasi Pembentukan Kunci pada Algoritma McEliece

Dalam tahap awal sistem kriptografi McEliece, pengirim memulai dengan membangkitkan kunci, sebuah proses penting yang akan membentuk dasar dari keamanan seluruh sistem. Pertama-tama, pengguna menggunakan algoritma decoding, seperti metode kode Hamming, untuk menentukan parameter kunci (n) dan (k), yang pada contoh ini diperoleh nilai (n=7) dan (k=4). Selanjutnya, matriks generator (G) ditentukan sebagai berikut :

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Setelahnya, dilakukan pembangkitan matriks (S) secara acak dengan dimensi ($k \times k$) dan matriks (P) dengan dimensi ($n \times n$).

$$S = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

$$P = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Selanjutnya, matriks generator (G') dihitung dengan mengalikan matriks (S), (G), dan (P).

$$G' = S \times G \times P$$

$$= \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$\times \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Dengan demikian, matriks generator (G') dengan ukuran (7×4) berhasil dibentuk dan siap digunakan sebagai kunci publik pada tahap enkripsi dalam algoritma McEliece. Proses ini menunjukkan kerangka kerja yang kompleks dan kuat dalam membangun kunci pada algoritma McEliece, sebuah langkah penting dalam menjaga keamanan informasi melalui sistem kriptografikunci publik.

4.2 Proses Enkripsi

Proses enkripsi pada algoritma McEliece membuktikan kehandalan dan ketangguhan sistem dalam melindungi informasi sensitif. Melalui konversi pesan teks menjadi representasi biner dengan panjang 8 bit, langkah awal ini memastikan bahwa setiap karakter pesan memiliki representasi yang unik. Selanjutnya, matriks generator G' yang telah dibentuk menjadi kunci publik digunakan untuk menghasilkan blok-blok *ciphertext*, menjadikan pesan tidak mudah terbaca oleh pihak yang tidak berhak. Penambahan *error* acak pada setiap blok ciphertext memberikan lapisan tambahan keamanan, melengkapi proses enkripsi ini sebagai

langkah kritis dalam menjaga kerahasiaan pesan dalam sistem kriptografi McEliece.

4.2.1 Algoritma Enkripsi Menggunakan McEliece

1. Input:
   - Menentukan plainteks yang akan dikirimkan
   - Menentukan vektor e berukuran (n)-bit yang memiliki t elemennon-nol (vektor dengan panjang n dan bobot t).
2. Proses:
   - Mengubah plainteks ke bentuk binary m berdasarkan tabelASCII.
   - Kode biner dibagi menjadi blok blok dengan panjang (k) bit
   - Hitung vektor $C = m \times G'$.
   - Hitung $C' = C + e$.
3. Output:
   - Matriks (C') merupakan cipertext yang siap untuk ditransmisikan.

4.2.2 Simulasi Proses Enkripsi Menggunakan Algoritma McEliece

Selanjutnya pada baagian ini pengirim mengubah pesan teks ke bentuk biner berdasarkan tabel ASCII dengan panjang 8 bit. Sebagai contoh pengirim akan mengirimkan pesan “AKU”, berdasarkan tabel ASCII didapatkan pesan AKU = 010000010100101101010101

$$A = [0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$$

$$K = [0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1]$$

$$U = [0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1]$$

Kemudian membagi kode biner ke beberapa blok m_i dengan panjang 4 bit.

$$m_1 = [0 \quad 1 \quad 0 \quad 0]$$

$$m_2 = [0 \quad 0 \quad 0 \quad 1]$$

$$m_3 = [0 \quad 1 \quad 0 \quad 0]$$

$$m_4 = [1 \quad 0 \quad 1 \quad 1]$$

$$m_5 = [0 \quad 1 \quad 0 \quad 1]$$

$$m_6 = [0 \quad 1 \quad 0 \quad 1]$$

Selanjutnya pesan m_i yang telah diubah ke bentuk biner dikalikan dengan matriks generator G' untuk menghasilkan matriks C_i.

$$C_1 = m_1 \times G' = [0 \quad 1 \quad 0 \quad 0] \times \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$$= [1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0]$$

$$C_2 = m_2 \times G' = [0 \quad 0 \quad 0 \quad 1] \times \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$$= [0 \quad 1 \quad 0 \quad 1 \quad 1 \quad 1 \quad 0]$$

$$C_3 = m_3 \times G' = [0 \quad 1 \quad 0 \quad 0] \times \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$$= [1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0]$$

$$C_4 = m_4 \times G' = [1 \quad 0 \quad 1 \quad 1] \times \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$$= [0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 1]$$

$$C_5 = m_5 \times G' = [0 \quad 1 \quad 0 \quad 1] \times \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$$= [1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0]$$

$$C_6 = m_6 \times G' = [0 \quad 1 \quad 0 \quad 1] \times \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$$= [1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0]$$

Kemudian C_i ditambahkan dengan *error* acak e dengan panjang 7 bit, dalam hal ini $e = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$. Dan didapatkan nilai C_i' sebagai berikut :

$$C_1' = C_1 + e = [1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0] + [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$$

$$= [1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1]$$

$$C_2' = C_2 + e = [0 \quad 1 \quad 0 \quad 1 \quad 1 \quad 1 \quad 0] + [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$$

$$= [0 \quad 1 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1]$$

$$C_3' = C_3 + e = [1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0] + [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$$

$$= [1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1]$$

$$C_4' = C_4 + e = [0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 1] + [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$$

$$= [0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0]$$

$$C_5' = C_5 + e = [1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0] + [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$$

$$= [1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1]$$

$$C_6' = C_6 + e = [1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0] + [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$$

$$= [1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1]$$

Ciphertext yang didapatkan dari pesan AKU = 110010101011111100101001101010010111001011.

4.3 Proses Dekripsi dan Pengkoresian *Error*

Terdapat langkah-langkah esensial untuk mengembalikan pesan ke bentuk semula. Dimulai setelah *ciphertext* diterima langkah awal dalam proses deskripsi ini adalah mengidentifikasi invers matriks (P) dan (S) yang sebelumnya digunakan pada tahap pembentukan kunci. Selanjutnya, invers matriks (P) ini akan dikalikan dengan setiap blok *ciphertext.* Kemudian masuk ke tahap pengkoreksaian *error* dengan mengkalikan dengam matriks parity check. Setelah tahap pengkoreksian *error*, *ciphertext* yang telah dikoreksi *error* dikalikan dengan invers dari matriks (S) dan hasilnya akan diubah ke dalam bentuk teks kembali berdasarkan tabel ASCII.

4.3.1 Algoritma Dekripsi Menggunakan Algoritma McEliece dengan Kode Hamming

1. Input
    - *Ciphertext* C_i' dengan panjang (n-bit).
    - Menghitung invers dari matriks $(n \times n)$.
    - Menghitung invers dari matriks S $(k \times k)$.
    - Menentukan matriks parity check H.
2. Proses
    - Mengalikan ciphertext (C_i') dengan matriks (P^{-1}). $(y_i = C_i' \times P^{-1})$
    - Mencari syndrome $S(y_i)$ dengan mengalikan matriks *parity check* dengan tranpose matriks (y_i). $(S(y_i) = H \times y^T)$
    - Mencari letak bit yang *error*.
    - Mengoreksi bit yang *error*.

- Mengalikan matriks (y_i') yang telah dikoreksi dengan matriks (S^{-1}). ($m_i' = y_i' \times S^{-1}$)
- Binary m_i' yang didapatkan dikembalikan kedalam bentuk teks berdasarkan tabel ASCII.

3. Output
   - Menghasilkan teks sesuai dengan pesan awal.

4.3.2 Simulasi Proses Dekripsi Algoritma McEliece dengan Kode Hamming

Proses dekripsi dan pengkoreksian *error* dapat dilakukan setelahpenerima menerima *cipertext*. Selanjutnya penerima menghitung nilai dari P^{-1}

$$P^{-1} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Setelah nilai P^{-1} didapatkan, *ciphertext* (C_i') dikalikan dengan P^{-1} untuk menghasilkan matriks y_i

$$y_1 = C_1' \times P^{-1} = [1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1] \times \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= [1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1]$$

$$y_2 = C_2' \times P^{-1} = [0 \quad 1 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1] \times \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= [1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1]$$

$$y_3 = C_3' \times P^{-1} = [1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1] \times \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= [1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1]$$

$$y_4 = C_4' \times P^{-1} = [0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0] \times \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= [0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0]$$

$$y_5 = C_5' \times P^{-1} = [1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1] \times \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= [0 \quad 1 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0]$$

$$y_6 = C_6' \times P^{-1} = [1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1] \times \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= [0 \quad 1 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0]$$

Setelah itu, memasuki tahap pengkoreksian *error* dengan mencari nilai dari *syndrome* $S(y_i)$. Untuk mencari nilai dari *syndrome* $S(y_i)$, matriks *parity check* H akan dikalikan dengan binary y_i^T

$$S(y_i) = H \times y_i^T$$

Matriks parity *check* sendiri didapatkan dari matriks G, dimana matriks generator berbentuk $G = [I_4 \; X]$, di mana I_4 adalah matriks identitas berukuran 4×4 dan X adalah suatu matriks berukuran 4×3. Dan matriks parity check $H = [Y \; I_3]$ dimana $Y = X^T$. Maka didapatkan matriks H berukuran 3×7.

$$G = \left[\begin{array}{cccc:ccc} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}\right]$$

$$\underbrace{\qquad\qquad}_{I_4} \quad \underbrace{\qquad}_{X}$$

$$H = \left[\begin{array}{cccc:ccc} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

$$\underbrace{\qquad\qquad}_{Y} \quad \underbrace{\qquad}_{I_3}$$

Setelah matriks *parity check* H didapatkan, selanjutnya mencari *syndrome* $S(y_i)$. Dengan mengalikan matriks *parity check* H dengan transpose dari y_i.

$$S(y_1) = H \times y_1^T = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

$$S(y_2) = H \times y_2^T = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

$$S(y_3) = H \times y_3^T = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

$$S(y_4) = H \times y_4^T = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

$$S(y_5) = H \times y_5^T = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

$$S(y_6) = H \times y_6^T = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

Jika nilai dari syndrom $S(y_i) \neq 0$ maka hal ini menunjukkan bahwa terjadi *error* pada y_i. Untuk mencari letak posisi bit yang salah dengan melihat matriks H. Jika $S(y_i)$ sama dengan kolom pada matriks H, maka *error* terjadi pada bit ke-i dari binary y.

$S(y_1) = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$ = nilai yang sama pada matriks H kolom ke-3, maka letak bit *error* pada y_1 terletak pada bit ke-3.

$S(y_2) = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$ = nilai yang sama pada matriks H kolom ke-3, maka letak bit *error* pada y_2 terletak pada bit ke-3.

$S(y_3) = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$ = nilai yang sama pada matriks H kolom ke-3, maka letak bit *error* pada y_3 terletak pada bit ke-3.

$S(y_4) = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$ = nilai yang sama pada matriks H kolom ke-3, maka letak bit *error* pada y_4 terletak pada bit ke-3.

$S(y_5) = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$ = nilai yang sama pada matriks H kolom ke-3, maka letak bit *error* pada y_5 terletak pada bit ke-3.

$S(y_6) = \begin{bmatrix}0\\1\\1\end{bmatrix}$ = nilai yang sama pada matriks *H* kolom ke-3, maka letak bit *error* pada y_6 terletak pada bit ke-3.

Setelah posisi *error* ditemukan, Langkah selanjutnya adalah pengkoreksian *error* dengan menambahkan *error* e' dengan bobot 1 pada bit yang salah.

$$y_1' = y_1 + e' = [1\ \ 0\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1] + [0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0]$$
$$= [1\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 1]$$
$$y_2' = y_2 + e' = [1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1] + [0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0]$$
$$= [1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 1\ \ 1]$$
$$y_3' = y_3 + e' = [1\ \ 0\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1] + [0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0]$$
$$= [1\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 1]$$
$$y_4' = y_4 + e' = [0\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1\ \ 0] + [0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0]$$
$$= [0\ \ 1\ \ 1\ \ 0\ \ 1\ \ 1\ \ 0]$$
$$y_5' = y_5 + e' = [0\ \ 1\ \ 1\ \ 1\ \ 0\ \ 1\ \ 0] + [0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0]$$
$$= [0\ \ 1\ \ 0\ \ 1\ \ 0\ \ 1\ \ 0]$$
$$y_6' = y_6 + e' = [0\ \ 1\ \ 1\ \ 1\ \ 0\ \ 1\ \ 0] + [0\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0]$$
$$= [0\ \ 1\ \ 0\ \ 1\ \ 0\ \ 1\ \ 0]$$

Setelah y_i dikoreksi didapatkan y_i' dan karena y_i' dihasilkan dari *standard form generator matriks* yang mana 4 bit dari kiri merupakan pesan informasi atau data bit, dan 3 bit dari kanan merupakan *parity* bit. Langkah selanjutnya adalah dengan mengambil 4 bit binary y_i' dari kiri untuk dikalikan dengan *S* invers.

$$m_i' = y_i' \times S^{-1}$$

Didapatkan nilai dari invers matriks S

$$S^{-1} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

Kemudian y_i' dikalikan dengan invers matriks S

$$m_1' = y_1' \times S^{-1} = [1 \quad 0 \quad 0 \quad 1] \times \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} = [0 \quad 1 \quad 0 \quad 0]$$

$$m_2' = y_2' \times S^{-1} = [1 \quad 1 \quad 0 \quad 0] \times \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} = [0 \quad 0 \quad 0 \quad 1]$$

$$m_3' = y_3' \times S^{-1} = [1 \quad 0 \quad 0 \quad 1] \times \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} = [0 \quad 1 \quad 0 \quad 0]$$

$$m_4' = y_4' \times S^{-1} = [0 \quad 1 \quad 1 \quad 0] \times \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} = [1 \quad 0 \quad 1 \quad 1]$$

$$m_5' = y_5' \times S^{-1} = [0 \quad 1 \quad 0 \quad 1] \times \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} = [0 \quad 1 \quad 0 \quad 1]$$

$$m_6' = y_6' \times S^{-1} = [0 \quad 1 \quad 0 \quad 1] \times \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} = [0 \quad 1 \quad 0 \quad 1]$$

Didapatkan hasil dari dekripsi adalah 010000010100101101010101. Untuk mengembalikan kedalam bentuk teks, perlu membagi pesan biner tersebut ke dalam blok blok dengan Panjang 8-bit. Berdasarkan tabel ASCII didapatkan 01000001-01001011-01010101 = AKU.

4.4 Analisa Hasil

Pembentukan Kunci pada Algoritma McEliece menunjukkan bahwa langkah-langkah yang kompleks dan kuat telah berhasil membentuk matriks generator (G') dengan ukuran (7×4). Tahap awal melibatkan algoritma decoding, dalam hal ini menggunakan metode kode Hamming, untuk menentukan parameter kunci (n) dan (k). Proses ini penting dalam membentuk dasar keamanan sistem kriptografi McEliece. Matriks generator (G) terbentuk sebagai hasil dari algoritma decoding dan menjadi dasar untuk pembangkitan matriks (S) dan (P) yang dibangkitkan secara acak. Penggunaan matriks permutasi acak (P) dan non singular matriks (S) menciptakan kekacauan struktural yang memperkuat keamanan kunci publik, menjadi keunikan algoritma McEliece.

Proses enkripsi pada algoritma McEliece menunjukkan kehandalan dan ketangguhan sistem dalam melindungi informasi sensitif. Tahap awal, yang melibatkan konversi pesan teks menjadi representasi biner dengan panjang 8 bit, kemudian membagi blok-blok dengan panjang 4 bit. Pengalian dengan matriks generator (G') sebagai kunci publik menjadikannya sulit terbaca oleh pihak yang tidak berhak. Proses penggabungan *error* acak pada setiap blok *ciphertext* memberikan lapisan tambahan keamanan, menjadikan proses enkripsi sebagai langkah kritis dalam menjaga kerahasiaan pesan dalam sistem algoritma McEliece.

Proses dekripsi dan pengkoreksian *error* pada algoritma McEliece dengan Kode Hamming melibatkanbeberapa langkah kritis untuk mengembalikan pesan terenkripsi ke bentuk semula.Setelah menerima *ciphertext*, langkah awal adalah mengidentifikasi matriks invers P dan S yang sebelumnya digunakan dalam pembentukan kunci. Matriks invers ini digunakan untuk mendekripsi setiap blok

pesan terenkripsi, dan hasilnya dikombinasikan untuk menghasilkan pesan biner dalam format ASCII. Pada tahap dekripsi, setiap blok pesan terenkripsi dikalikan dengan matriks invers P, kemudian pengkoreksian error dengan mencari *syndrome* $(S(y_i))$ dihitung dengan mengalikan matriks *parity check* H dengan transposisi dari matriks hasil dekripsi. Jika *syndrome* tidak sama dengan nol, itu menunjukkan adanya *error* pada pesan, dan lokasi bit yang salah dapat ditemukan dengan memeriksa matriks H. Setelah lokasi *error* ditemukan, dilakukan pengkoreksian *error* dengan menambahkan *error* pada bit yang bersangkutan. Selanjutnya, hasil dikalikan dengan matriks invers S untuk mendapatkan pesan yang telah dikoreksi. Hasil binary dikembalikan ke bentuk teks berdasarkan tabel ASCII. Dengan mengikuti langkah-langkah ini, algoritma McEliece dengan Kode Hamming dapat diandalkan untuk mengamankan dan mendekripsi pesan dengan efektif.

Penelitian ini berhasil menunjukkan bahwa Algoritma McEliece mampu menghadirkan langkah-langkah pengamanan pesan yang kompleks dan kuat. Algoritma McEliece dalam penelitian ini melibatkan langkah-langkah pengkoreksian error dengan Kode Hamming yang menunjukkan keberhasilan dalam mengembalikan pesan terenkripsi ke bentuk semula. Hal ini menunjukkan bahwa kemampuan Algoritma McEliece dengan kode Hamming sebagai sarana andal untuk mengamankan dan mendekripsi pesan dengan efektif.

4.5 Kajian Islami

Sifat wajib yang dimiliki oleh seorang Rasul adalah ciri khas mereka dalam melaksanakan tugas mereka sebagai pemimpin. Namun, sifat-sifat ini tidak hanya relevan bagi pemimpin semata, melainkan setiap individu harus mengadopsi sifat-sifat ini dalam menjalani kehidupan mereka, dan dapat diterapkan dalam berbagai

bidang. Beberapa sifat ini meliputi:

1. Shiddiq

Sifat shiddiq yang berarti kejujuran dan kebenaran, merupakan karakteristik penting yang ada dalam diri Rasulullah Muhammad SAW. Beliau selalu bersikap jujur dan tulus dalam perkataan dan tindakan. Dalam alquran surat Maryam ayat 41

وَٱذْكُرْ فِى ٱلْكِتَـٰبِ إِبْرَٰهِيمَ ۚ إِنَّهُۥ كَانَ صِدِّيقًا نَّبِيًّا ٤١

Artinya: *Ceritakanlah (Hai Muhammad) kisah Ibrahim di dalam Al Kitab (Al Quran) ini. Sesungguhnya ia adalah seorang yang sangat membenarkan lagi seorang Nabi.*

Dalam interaksi sosial, sifat shiddiq juga memiliki peranan yang sangat penting, seperti ketika berbicara tentang penyaluran informasi. Mirip dengan cara yang diterapkan oleh Rasulullah, informasi yang disampaikan harus memiliki dasar kebenaran, akurat, dan relevan. Di era digital saat ini, implementasi sifat shiddiq dalam konteks pesan yang dikirimkan berarti bahwa isi pesan yang dikirim harus identik dengan apa yang diterima, tanpa ada upaya manipulasi ataupenyebaran informasi palsu.

2. Amanah

Amanah yang memiliki arti dapat dipercaya, adalah sifat yang sangat penting dalam konteks tugas para rasul. Para rasul selalu berusaha untuk menjauhi perbuatan dosa agar mereka dapat mempertahankan kepercayaan umat pada mereka. dalam alquran surat an-nisa ayat 58 disebutkan

۞ إِنَّ ٱللَّهَ يَأْمُرُكُمْ أَن تُؤَدُّوا۟ ٱلْأَمَـٰنَـٰتِ إِلَىٰٓ أَهْلِهَا وَإِذَا حَكَمْتُم بَيْنَ ٱلنَّاسِ أَن تَحْكُمُوا۟ بِٱلْعَدْلِ ۚ إِنَّ ٱللَّهَ نِعِمَّا يَعِظُكُم بِهِۦٓ ۗ إِنَّ ٱللَّهَ كَانَ سَمِيعًۢا بَصِيرًا ٥٨

Artinya: *Sesungguhnya Allah menyuruh kamu menyampaikan amanat kepada yang berhak menerimanya, dan (menyuruh kamu) apabila menetapkan hukum di antara manusia supaya kamu menetapkan dengan adil. Sesungguhnya Allah memberi pengajaran yang sebaik-baiknya kepadamu. Sesungguhnya Allah adalah Maha Mendengar lagi Maha Melihat.*

Hal ini juga relevan dalam konteks penyampaian pesan di era digital oleh perangkat elektronik, di mana pesan yang diterima harus dapat dipercaya sebagai orisinal dan tidak mengalami perubahan atau manipulasi selama prosespengiriman. Keamanahan informasi menjadi esensial dalam era digital, dan menjaga integritas pesan adalah aspek yang sangat penting dalam menjaga kepercayaan pengguna terhadap teknologi komunikasi.

3. Fathonah

Fatanah yang berarti memiliki kecerdasan, kebijaksanaan, dan kepandaian, adalah sifat yang sangat penting bagi para rasul. Sebagai utusan Allah SWT kepada umat manusia, para rasul memiliki kemampuan untuk memahami berbagai permasalahan yang dihadapi umat manusia dan memberikan solusi yang tepat untuk mengatasi permasalahan tersebut. Dalam proses menyampaikan informasi, penting untuk menggunakan metode yang sesuai dan cerdas, yang dapat beradaptasi dengan perkembangan zaman. Manusia terus mengembangkan pengetahuan yang telah diberikan kepada mereka, dan ini memungkinkan mereka untuk memanfaatkan ilmutersebut dalam meningkatkan efektivitas komunikasi mereka.

4. Tabligh

Tablig yang berarti menyampaikan wahyu, adalah salah satu tugas utama para rasul. Dalam melaksanakan misi kerasulannya, seorang rasul

bertanggung jawab untuk menyampaikan wahyu yang diterimanya kepada umat manusia, yangkemudian harus diimani oleh mereka. Wahyu yang telah disampaikan oleh para rasul tersebut dapat berupa pengetahuan, syariat, maupun pedoman, ataupun risalah kenabian yang lain. Sekalipun wahyu yang disampaikannya tidak mudah maupun bukan sesuatu yang menyenangkan, para rasul akan senantiasa menyampaikannya tanpa mengurangi satu huruf pun sesuai dengan surah Al Maidah ayat 67

۞ يَـٰٓأَيُّهَا ٱلرَّسُولُ بَلِّغْ مَآ أُنزِلَ إِلَيْكَ مِن رَّبِّكَ ۖ وَإِن لَّمْ تَفْعَلْ فَمَا بَلَّغْتَ رِسَالَتَهُۥ ۚ وَٱللَّهُ يَعْصِمُكَ مِنَ ٱلنَّاسِ ۗ إِنَّ ٱللَّهَ لَا يَهْدِى ٱلْقَوْمَ ٱلْكَـٰفِرِينَ ٦٧

Artinya: *Hai Rasul, sampaikanlah apa yang diturunkan kepadamu dari Tuhanmu. Dan jika tidak kamu kerjakan (apa yang diperintahkan itu, berarti) kamu tidak menyampaikan amanat-Nya. Allah memelihara kamu dari (gangguan) manusia. Sesungguhnya Allah tidak memberi petunjuk kepada orang-orang yang kafir.*

Nilai tabligh memiliki makna yang mencakup tindakan komunikasi dan penyampaian informasi yang bermanfaat kepada siapa pun tanpa pandang bulu.Ini mencakup upaya untuk berbagi pengetahuan, nilai-nilai, dan pesan positif kepada semua orang dengan tujuan memperluas pemahaman dan pengetahuan, serta mempromosikan kebaikan dalam masyarakat dan lingkungan sekitarnya.

Upaya menerapkan sifat-sifat wajib Rasul dalam kehidupan manusia dapat diwujudkan melalui berbagai aspek kehidupan, termasuk kehidupan modern di era digital. Salah satu cara untuk mencapai hal ini adalah dengan memanfaatkan kriptografi dalam proses pengiriman pesan. Penggunaan kriptografi memastikan bahwa pesan yang dikirim identik dengan pesan yang diterima, dan tidak ada

manipulasi atau penyebaran informasi palsu (menggambarkan sifat Shiddiq). Dengan demikian, orisinalitas pesan dapat dipercaya (mencerminkan sifat Amanah), sementara penggunaan teknologi ini juga mencerminkan kebijaksanaan dan kecerdasan dalam beradaptasi dengan perkembangan zaman (menggambarkan sifat Fathonah). Selain itu, kriptografi memastikan bahwa pesan tersampaikan dengan aman kepada penerima yang dituju, mencerminkan semangat Tabligh, atau penyampaian pesan yang efisien dan efektif.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil dan pembahasan diatas didapatkan kesimpulan yaitu:

1. Pembentukan matriks generator melibatkan matriks non singular S berukuran $k \times k$ dikalikan dengan matriks G yang berukuran $k \times n$ (matriks generator berbentuk $G = [I_k \ X]$, di mana I_k adalah matriks identitas berukuran $k \times k$ dan X adalah suatu matriks berukuran $k \times (n - k)$). Kemudian dikalikan lagi dengan matriks permutasi P berukuran $n \times n$ dan menghasilkan matriks generator G'. ($G' = S \times G \times P$)
2. Proses enkripsi dimulai dari mengubah pesan teks menjadi biner dengan panjang 8-bit berdasarkan tabel ASCII. Kemudian dibagi ke dalam blok blok m_i dengan panjang 4-bit. Selanjutnya m_i dikalikan dengan matriks generator G' ($C_i = m_i \times G'$) dan hasilnya ditambahkan error dengan bobot 1 untuk menghasilkan *cipherteks* ($C_i' = C_i + e$).
3. Proses dekripsi diawali dengan mengalikan *chiperteks* dengan invers matriks permutasi ($y_i = C_i' \times P^{-1}$). Kemudian kode Hamming mulai mengkoreksi dan memperbaiki data bit dengan mencari *syndrome* dari y_i. Matriks parity check H yang didapatkan dari $H = [Y \ I_3]$ dimana $Y = X^T$. *Syndrome* didapatkan dari matriks *parity check* yang dikalikan dengan tranpose dari y_i ($S(y_i) = H \times y_i^T$). Setelah pesan selesai dikoreksi dan diperbaiki kemudian dikalikan dengan invers dari matriks non singular S. ($m_i' = y_i' \times S^{-1}$). Kemudian blok blok digabungkan secara berurutan sehingga menciptakan biner dengan panjang 8-bit. Dengan mengacu pada

tabel ASCII pesan asli dalam bentuk biner dapat diubah ke bentuk teks.

Dengan mengimplementasikan kode Hamming, algoritma McEliece dapat meningkatkan keandalan dan keamanan pertukaran informasi, sekaligus menghadapi potensi ancaman dari komputer kuantum.

5.2 Saran

Saran untuk penelitian selanjutnya, diharapkan dapat melakukan pengamanan pertukaran data dalam bentuk gambar, audio, ataupun video dengan menggunakan algoritma McEliece yang menggunakan kode Hamming sebagai koreksi *error*.

DAFTAR PUSTAKA

Sinurat, S., & Siagian, E. R. (2022). Learning Text Data Security in Documents Using McEliece's Algorithm. INFOKUM, 10(5), 323-330.

Hafizhah, H., & Utomo, P. H. (2023, March). Error Detection dan Error Correction pada Komunikasi Digital Menggunakan Hamming Code. In Prosiding Seminar Pendidikan Matematika dan Matematika (Vol. 7).

Ling, S., & Xing, C. (2004). Coding theory: a first course. Cambridge University Press.

Vanstone, S. A., & Van Oorschot, P. C. (2013). An introduction to error correcting codes with applications (Vol. 71). Springer Science & Business Media.

Gallian, Joseph A. (2006). Contemporary Abstract Algebra, Seventh Edition. United States of America: Brooks/Cole Cengage Learning.

Departemen Agama, R. I. (2009). Al-quran dan dan Terjemahan.

Rurik, W., & Mazumdar, A. (2016, September). Hamming codes as error- reducing codes. In *2016 IEEE Information Theory Workshop (ITW)* (pp. 404-408). IEEE.

Biswas, B., & Sendrier, N. (2008). McEliece cryptosystem implementation: Theory and practice. In *Post-Quantum Cryptography: Second International Workshop, PQCrypto 2008 Cincinnati, OH, USA, October 17-19, 2008 Proceedings 2* (pp. 47-62). Springer Berlin Heidelberg.

Ilmiyah, N. F. (2019, May). Kajian Tentang Kriptosistem McEliece Dalam Menghadapi Tantangan Komputer Kuantum Di Era Revolusi Industri 4.0.In *Prosiding Seminar Nasional MIPA Kolaborasi* (Vol. 1, No. 1, pp. 216-226)

Hamming, R. W. (1986). Coding and information theory. Prentice-Hall, Inc.

Lidl, R., Pilz, G., Lidl, R., & Pilz, G. (1998). Cryptology. Applied Abstract Algebra, 239-282.

Kumar, U. K., & Umashankar, B. S. (2007, February). Improved hamming code for error detection and correction. In 2007 2nd International Symposium on Wireless Pervasive Computing. IEEE.

Singh, A. K. (2016, December). Error detection and correction by hamming code. In 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC) (pp. 35-37). IEEE.

Oktavia, R. E., Utomo, P. H., & Martini, T. S. (2023). PENERAPAN KODE REED SOLOMON PADA KRIPTOSISTEM MCELIECE. *FIBONACCI: Jurnal Pendidikan Matematika dan Matematika, 9*(1), 79-88.

LAMPIRAN

Lampiran 1. Table ASCII

Decimal	Octal	Hex	Binary	Value	Description
000	000	000	0000 0000	NUL	**"null" character**
001	001	001	0000 0001	SOH	**start of header**
002	002	002	0000 0010	STX	**start of text**
003	003	003	0000 0011	ETX	**end of text**
004	004	004	0000 0100	EOT	**end of transmission**
005	005	005	0000 0101	ENQ	**enquiry**
006	006	006	0000 0110	ACK	**acknowledgment**
007	007	007	0000 0111	BEL	**bell**
008	010	008	0000 1000	BS	**backspace**
009	011	009	0000 1001	HT	**horizontal tab**
010	012	00A	0000 1010	LF	**line feed**

Decimal	Octal	Hex	Binary	Value	Description
011	013	00B	0000 1011	VT	**vertical tab**
012	014	00C	0000 1100	FF	**form feed**
013	015	00D	0000 1101	CR	**carriage return**
014	016	00E	0000 1110	SO	**shift out**
015	017	00F	0000 1111	SI	**shift in**
016	020	010	0001 0000	DLE	**data link escape**
017	021	011	0001 0001	DC1	**device control 1 (XON)**
018	022	012	0001 0010	DC2	**device control 2**
019	023	013	0001 0011	DC3	**device control 3 (XOFF)**
020	024	014	0001 0100	DC4	**device control 4**
021	025	015	0001 0101	NAK	**negative acknowledgement**
022	026	016	0001 0110	SYN	**synchronous idle**

Decimal	Octal	Hex	Binary	Value	Description
023	027	017	0001 0111	ETB	**end of transmission block**
024	030	018	0001 1000	CAN	**cancel**
025	031	019	0001 1001	EM	**end of medium**
026	032	01A	0001 1010	SUB	**substitute**
027	033	01B	0001 1011	ESC	**escape**
028	034	01C	0001 1100	FS	**file separator**
029	035	01D	0001 1101	GS	**group separator**
030	036	01E	0001 1110	RS	**request to send/record separator**
031	037	01F	0001 1111	US	**unit separator**
032	040	020	0010 0000	SP	**space**
033	041	021	0010 0001	!	**exclamation mark**
034	042	022	0010 0010	"	**double quote**

Decimal	Octal	Hex	Binary	Value	Description
035	043	023	0010 0011	#	**number sign**
036	044	024	0010 0100	$	**dollar sign**
037	045	025	0010 0101	%	**percent**
038	046	026	0010 0110	&	**ampersand**
039	047	027	0010 0111	'	**single quote**
040	050	028	0010 1000	(	**left/opening parenthesis**
041	051	029	0010 1001	)	**right/closing parenthesis**
042	052	02A	0010 1010	*	**asterisk**
043	053	02B	0010 1011	+	**plus**
044	054	02C	0010 1100	,	**comma**
045	055	02D	0010 1101	-	**minus or dash**
046	056	02E	0010 1110	.	**dot**

Decimal	Octal	Hex	Binary	Value	Description
047	057	02F	0010 1111	/	**forward slash**
048	060	030	0011 0000	0	
049	061	031	0011 0001		1
050	062	032	0011 0010	2	
051	063	033	0011 0011		3
052	064	034	0011 0100	4	
053	065	035	0011 0101		5
054	066	036	0011 0110	6	
055	067	037	0011 0111		7
056	070	038	0011 1000	8	
057	071	039	0011 1001		9
058	072	03A	0011 1010	:	**colon**

Decimal	Octal	Hex	Binary	Value	Description
059	073	03B	0011 1011	;	**semi-colon**
060	074	03C	0011 1100	<	**less than**
061	075	03D	0011 1101	=	**equal sign**
062	076	03E	0011 1110	>	**greater than**
063	077	03F	0011 1111	?	**question mark**
064	100	040	0100 0000	@	**"at" symbol**
065	101	041	0100 0001		A
066	102	042	0100 0010	B	
067	103	043	0100 0011		C
068	104	044	0100 0100	D	
069	105	045	0100 0101		E
070	106	046	0100 0110	F	

Decimal	Octal	Hex	Binary	Value	Description
071	107	047	0100 0111		G
072	110	048	0100 1000	H	
073	111	049	0100 1001		I
074	112	04A	0100 1010	J	
075	113	04B	0100 1011		K
076	114	04C	0100 1100	L	
077	115	04D	0100 1101		M
078	116	04E	0100 1110	N	
079	117	04F	0100 1111		O
080	120	050	0101 0000	P	
081	121	051	0101 0001		Q

Decimal	Octal	Hex	Binary	Value	Description
082	122	052	0101 0010	R	
083	123	053	0101 0011		S
084	124	054	0101 0100	T	
085	125	055	0101 0101		U
086	126	056	0101 0110	V	
087	127	057	0101 0111		W
088	130	058	0101 1000	X	
089	131	059	0101 1001		Y
090	132	05A	0101 1010	Z	
091	133	05B	0101 1011	[	**left/opening bracket**
092	134	05C	0101 1100	\	**back slash**

Decimal	Octal	Hex	Binary	Value	Description
093	135	05D	0101 1101	]	**right/closing bracket**
094	136	05E	0101 1110	^	**caret/circumflex**
095	137	05F	0101 1111	_	**underscore**
096	140	060	0110 0000	`	
097	141	061	0110 0001		a
098	142	062	0110 0010	b	
099	143	063	0110 0011		c
100	144	064	0110 0100	d	
101	145	065	0110 0101		e
102	146	066	0110 0110	f	
103	147	067	0110 0111		g

Decimal	Octal	Hex	Binary	Value	Description
104	150	068	0110 1000	h	
105	151	069	0110 1001		i
106	152	06A	0110 1010	j	
107	153	06B	0110 1011		k
108	154	06C	0110 1100	l	
109	155	06D	0110 1101		m
110	156	06E	0110 1110	n	
111	157	06F	0110 1111		o
112	160	070	0111 0000	p	
113	161	071	0111 0001		q
114	162	072	0111 0010	r	

Decimal	Octal	Hex	Binary	Value	Description
115	163	073	0111 0011		s
116	164	074	0111 0100	t	
117	165	075	0111 0101		u
118	166	076	0111 0110	v	
119	167	077	0111 0111		w
120	170	078	0111 1000	x	
121	171	079	0111 1001		y
122	172	07A	0111 1010	z	
123	173	07B	0111 1011	{	**left/opening brace**
124	174	07C	0111 1100	|	**vertical bar**
125	175	07D	0111 1101	}	**right/closing brace**

Decimal	Octal	Hex	Binary	Value	Description
126	176	07E	0111 1110	~	**tilde**
127	177	07F	0111 1111	DEL	**delete**

Lampiran 2. Progam implementasi kode Hamming dan McEliece menggunakan python

```
#proses pembentukan kunci
import numpy as np
def matrix_multiply_modulo_2(matrix_a, matrix_b):
    result = np.dot(matrix_a, matrix_b) % 2
    return result
# Matriks Singular S (4x4)
S = np.array([
    [1, 1, 0, 1],
    [1, 0, 0, 1],
    [0, 1, 1, 1],
    [1, 1, 0, 0]
])
# Matriks Generator G (4x7)
I_k = np.eye(4, dtype=int)
X = np.array([
    [1, 1, 0],
    [1, 0, 1],
    [0, 1, 1],
    [1, 1, 1] ])
G = np.concatenate((I_k, X), axis=1)

# Matriks X diisi secara acak untuk contoh G = np.concatenate((I_k, X),
axis=1)
```

```
print("\nMatriks G:")
print(G)
# Matriks Permutasi P (7x7)
P = np.array([[0, 1, 0, 0, 0, 0, 0],
[0, 0, 0, 1, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 1],
[1, 0, 0, 0, 0, 0, 0],
[0, 0, 1, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 1, 0],
[0, 0, 0, 0, 1, 0, 0]])
# Hitung matriks G'
matriks_G_prime = matrix_multiply_modulo_2(S, matrix_multiply_modulo_2(G, P))
print("\nMatriks G':")
print(matriks_G_prime)

#proses enkripsi
def text_to_binary(text):
    # Mengonversi teks menjadi representasi biner 8 bit berdasarkan tabel ASCII
    binary_text = ''.join(format(ord(char), '08b') for char in text)
    return binary_text

def split_binary_into_blocks(binary_text, block_length):
    # Membagi biner menjadi blok-blok dengan panjang tertentu
```

```
        blocks = [binary_text[i:i+block_length] for i in range(0, len(binary_text), block_length)]
        return blocks
    def name_blocks(blocks):
        named_blocks = {f"m_{i+1}": block for i, block in enumerate(blocks)}
        return named_blocks
    # Masukan teks
    input_text = "AKU"
    # Ubah teks menjadi representasi biner 8 bit
    binary_representation = text_to_binary(input_text)
    # Bagi representasi biner menjadi blok-blok dengan panjang 4 bit
    block_length = 4
    binary_blocks = split_binary_into_blocks(binary_representation, block_length)
    named_blocks = name_blocks(binary_blocks)
    print("Teks:", input_text)
    print("Representasi Biner 8 Bit:", binary_representation)
    print(f"Blok-blok dengan Panjang {block_length} Bit:")
    for name, block in named_blocks.items():
        print(f"{name}: {block}")

    # Fungsi untuk mengalikan blok biner dengan matriks G'
    def multiply_with_G_prime(binary_blocks, matriks_G_prime):
        result_blocks = {}
        for i, (name, block) in enumerate(binary_blocks.items(), start=1):
```

```
            # Ubah blok biner menjadi representasi matriks 4x1
            block_matrix = np.array([int(bit) for bit in block]).reshape(1, -1)
        # Transposisi matriks
            result = matrix_multiply_modulo_2(block_matrix, matriks_G_prime)
            result_blocks[f"c_{i}"] = result
        return result_blocks
    # Kalikan blok-binernya dengan matriks G' dan definisikan hasilnya sebagai c_i
    result_with_G_prime = multiply_with_G_prime(named_blocks, matriks_G_prime)
    print("\nHasil perkalian blok-binernya dengan matriks G' (c_i):")
    for name, result in result_with_G_prime.items():
        print(f"{name}: {result}")

    # Generate random error e with length 7 and weight 1
    def generate_random_error(length):
        error = np.zeros(length, dtype=int)
        error_index = np.random.randint(length)
        error[error_index] = 1
        return error

    # Generate random error e
    random_error = generate_random_error(7)
    print("Nilai e (error acak dengan panjang 7 bit dan berbobot 1):")
    print(random_error)
```

```
# Define C'_i as the addition of C_i with error e
def add_error_to_C(result_with_G_prime, error):
    result_with_error = {}
    for name, result in result_with_G_prime.items():
        result_with_error[name] = (result + error) % 2
    return result_with_error
# Add error e to C_i and define it as C'_i
result_with_error = add_error_to_C(result_with_G_prime, random_error)
# Display C'_i
print("\nHasil penjumlahan C_i dengan error e (C'_i):")
for name, result in result_with_error.items():
    print(f"{name}: {result}")

#proses dekripsi dan pengkoreksian error
# Calculate the inverse of matrix P
P_inv = np.linalg.inv(P) % 2

# Function to perform decryption operation
def decrypt_message(result_with_error, P_inverse):
    decrypted_result = {}
    for name, result in result_with_error.items():
        # Multiply C'_i with P^-1
        decrypted = matrix_multiply_modulo_2(result, P_inverse)
        decrypted_result[name] = decrypted
    return decrypted_result
```

```
    # Perform decryption operation
    decrypted_result = decrypt_message(result_with_error, P_inv)
    # Display the decrypted result
    print("\nHasil Dekripsi (y_i = C'_i * P^-1):")
    for name, result in decrypted_result.items():
        print(f"{name}: {result}")
    # Transpose matrix X to get Y
    Y = np.transpose(X)
    # Create identity matrix I_3 (3x3)
    I_h = np.eye(3, dtype=int)
    # Concatenate Y and I_3 to get matrix H (Parity Check)
    H = np.concatenate((Y, I_h), axis=1)
    # Display matrix H
    print("\nMatriks Parity Check (H):")
    print(H)

    # Function to calculate syndrome S(y_i) = H * (y_i)^T
    def calculate_syndrome(decrypted_result, parity_check_matrix):
        syndromes = {}
        for name, result in decrypted_result.items():
            # Transpose y_i to perform the operation
            y_i_transpose = np.transpose(result)
            # Calculate S(y_i) = H * (y_i)^T
            syndrome = matrix_multiply_modulo_2(parity_check_matrix,
y_i_transpose)
```

```
            syndromes[name] = syndrome
        return syndromes
    # Calculate syndrome S(y_i) for each y_i obtained from previous operations
    syndrome_y_i = calculate_syndrome(decrypted_result, H)
    # Display the syndromes
    print("\nSyndrome S(y_i) = H * (y_i)^T:")
    for name, syndrome in syndrome_y_i.items():
        print(f"{name}: {syndrome}")
    # Function to check and correct errors in y_i
    def check_and_correct_errors(syndromes, parity_check_matrix, decrypted_result):
        error_positions = {}
        for name, syndrome in syndromes.items():
            # Check if syndrome matches any column in H
            error_pos = np.where((parity_check_matrix == syndrome).all(axis=0))[0]
            if len(error_pos) > 0:
                error_positions[name] = error_pos[0]  # Get the first error position found
        return error_positions
    # Check and correct errors in y_i
    error_positions = check_and_correct_errors(syndrome_y_i, H, decrypted_result)
    # Display the error positions if errors are found
    if error_positions:
```

```
        print("\nPosisi bit yang salah pada setiap y_i:")
        for name, pos in error_positions.items():
            print(f"{name}: Bit ke-{pos+1}")
    else:
        print("\nTidak ada kesalahan pada y_i.")

    # Function to correct errors in y_i
    def correct_errors(decrypted_result, error_positions):
        for name, pos in error_positions.items():
            # Get the corresponding y_i and error position
            y_i = decrypted_result[name]
            # Correct the error bit
            y_i[0, pos] = (y_i[0, pos] + 1) % 2  # Toggle the bit
        return decrypted_result
    # Correct errors in y_i
    corrected_result = correct_errors(decrypted_result, error_positions)
    # Display the corrected result
    print("\nHasil setelah pengkoreksian error:")
    for name, result in corrected_result.items():
        print(f"{name}: {result}")
    # Function to take the leftmost 4 bits and multiply with the inverse of matrix S
    def multiply_with_S_inverse(corrected_result, S_inverse):
        m_prime = {}
        for name, result in corrected_result.items():
```

```
            # Take the leftmost 4 bits
            left_4_bits = result[:, :4]
            # Multiply with the inverse of matrix S
            m_i_prime = matrix_multiply_modulo_2(left_4_bits, S_inverse)
            m_prime[name] = m_i_prime
        return m_prime
    # Calculate inverse of matrix S
    S_inv = np.linalg.inv(S) % 2
    # Multiply the leftmost 4 bits with the inverse of matrix S for each corrected result
    m_prime_result = multiply_with_S_inverse(corrected_result, S_inv)
    # Display the result after multiplication with S^-1
    print("\nHasil setelah perkalian dengan invers dari matriks S (m'_i):")
    for name, result in m_prime_result.items():
        print(f"{name}: {result}")
    # Combine all m'_i into a single row
    def combine_m_prime(m_prime_result):
        combined = np.concatenate([value for value in m_prime_result.values()], axis=1)
        return combined
    # Function to convert binary block to text ASCII
    def binary_block_to_text(binary_block):
        decimal_value = int(binary_block, 2)  # Convert binary to decimal value
        text = chr(decimal_value)  # Convert decimal value to ASCII character
        return text
```

```
    # Function to convert binary blocks to ASCII text
    def blocks_to_text(blocks):
        text = ''
        for block in blocks:
            text += binary_block_to_text(''.join(map(str, block)))  # Convert the block to a string and then to ASCII
        return text
    # Combine all m'_i into a single row
    combined_m_prime = combine_m_prime(m_prime_result)
    # Display the combined m'_i
    print("\nHasil gabungan semua m'_i menjadi satu baris:")
    print(combined_m_prime)

    def split_combined_m_prime(combined_m_prime, block_length):
        blocks = [combined_m_prime[:, i:i+block_length] for i in range(0, combined_m_prime.shape[1], block_length)]
        return blocks

    # Panjang blok yang diinginkan
    block_length = 8
    split_blocks = split_combined_m_prime(combined_m_prime, block_length)
    print(f"\nBlok-blok dengan Panjang {block_length} Bit:")
    for i, block in enumerate(split_blocks, start=1):
        print(f"Blok ke-{i}: {block}")
```

RIWAYAT HIDUP

M. Fajrul Falakh lahir di Jember pada tanggal 17 September 1999, tinggal di Malang Jl. Mt. Haryono Gg. IX No. 433 Kota Malang. Merupakan anak kedua dari tiga bersaudara dari pasangan Nur Alim Zaini dan Umi Astutik.

Riwayat pendidikannya dimulai dari MI Mambau'ul Khoirot di Umbulsari-Jember yang berhasil diselesaikan pada tahun 2011. Kemudian, melanjutkan pendidikan di MTs Ma'arif NU Kencong Jember dan berhasil menamatkannya pada tahun 2014. Kemudian melanjutkan ke jenjang pendidikan menengah atas di SMK Ma'arif NU Kencong Jember dengan mengambil jurusan Teknik Komputer dan Jaringan, lulus pada tahun 2017. Pada tahun yang sama, ia melanjutkan pendidikannya di UIN Maulana Malik Ibrahim Malang, memilih Program Studi Matematika.

Selama masa kuliahnya, tidak hanya berfokus pada akademis semata, namun juga aktif dalam berbagai kegiatan organisasi di luar kampus. Salah satunya adalah keanggotaan dan peran pengurus dalam PSHT UIN Malang sebagai organisasi ekstra kampus yang ikut serta dalam kegiatan pengembangan diri. Tidak hanya itu, ia juga meraih sejumlah prestasi dalam bidang seni ganda. Ia berhasil memenangkan beberapa kompetisi, juara 2 seni ganda tingkat provinsi pada Unitomo Cup di Universitas Budi Utomo Kota Surabaya tahun 2018, juara 3 seni ganda tingkat nasional pada UGM Terate Championship di Universitas Gajah Mada tahun 2018 dan berbagai prestasi lainnya dalam berbagai kompetisi seni ganda baik tingkat provinsi maupun nasional.

Demikianlah gambaran singkat mengenai perjalanan hidup dan prestasi yang telah diraih oleh M. Fajrul Falakh. Dengan komitmen yang kuat dalam berbagai kegiatan di luar kampus yang telah membentuknya menjadi individu yang berkompeten dan berprestasi.

KEMENTERIAN AGAMA RI
UNIVERSITAS ISLAM NEGERI
MAULANA MALIK IBRAHIM MALANG
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Gajayana No. 50 Dinoyo Malang Telp/Fax.(0341)558933

BUKTI KONSULTASI SKRIPSI

Nama : M. Fajrul Falakh
NIM : 17610114
Fakultas/ Program Studi : Sains dan Teknologi/ Matematika
Judul Skripsi : Implementasi Kode Hamming pada Algoritma McEliece untuk Mengamankan Pesan
Pembimbing I : Muhammad Khudzaifah, M.Si
Pembimbing II : Mohammad Nafie Jauhari, M.Si

No	Tanggal	Hal	Tanda Tangan
1	25 Agustus 2023	Konsultasi Bab I, II, dan III	1.
2	28 Agustus 2023	Konsultasi Kajian Islam	2.
3	18 September 2023	Revisi Bab I, II, dan III	3.
4	20 September 2023	Revisi Kajian Islam	4.
5	16 Oktober 2023	ACC Bab I, II, dan III	5.
6	17 Oktober 2023	ACC Kajian Islam	6.
7	29 November 2023	Konsultasi Bab IV dan V	7.
8	05 Desember 2023	ACC Bab IV dan V	8.
9	05 Desember 2023	ACC Kajian Islam Bab IV	9.
10	21 Desember 2023	Revisi Seminar Hasil	10.
11	22 Desember 2023	ACC Revisi Seminar Hasil	11.
12	27 Desember 2023	ACC Sidang Skripsi	12.
13	30 Desember 2023	ACC Keseluruhan Kajian Islami	13.
14	30 Desember 2023	ACC Keseluruhan	14.

Malang, 30 Desember 2023
Mengetahui,
Ketua Program Studi Matematika

Dr. Elly Susanti, M.Sc
NIP. 19741129 200012 2 005