Jurnal Sains Dan Teknologi Informasi
Vol 1, No 4, September 2022, Hal 121-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v1i4.2293
Website https://ejurnal.seminar-id.com/index.php/jussi

Implementasi Algoritma MD4 Pada Aplikasi Duplicate Audio Scanner

Tri Windi Astuti

Program Studi Tenik Informatika, Fakultas Ilmu Komputer Dan Teknologi Informasi, Universitas Budi Darma,
Jalan Sisingamangaraja No.338, Medan, Sumatera Utara, Indonesia
Email: triwindi123@gmail.com

Abstrak−Sekarang ini banyak sekali aplikasi yang membantu proses menduplikat audio, tentunya hal ini menyita lebih banyak ruang penyimpanan. Untuk itu diperlukan suatu cara yang dapat digunakan untuk mengindentifikasi file audio yang sama ataupun yang duplikat. Solusi yang dapet dilakukan untuk menangani penggandaan file audio atau duplikat tersebut adalah dengan menerapkan teknik kriptografi jenis hash dimana dilakukan proses mengindentifikasi nilai hash dari file audio sehinggan nilai yang dihasilkan berbebeda dengan file yang aslinya. Adapun algoritma kriptografi jenis hash yang digunakan adalah jenis algoritma MD4. Hasil yang didapat dari proses menerapkan algoritma MD4 adalah mengelompokan file audio berdasarkan nilai hash yang sama, sehinggan mempermudah dan mempercepat pengguna untuk menghapus file audio yang duplikat..

Kata Kunci: Kriptografi; File Audio; MD4

Abstract−Nowadays there are many applications that help the process of duplicating audio, of course this takes up more storage space. For that we need a method that can be used to identify the same or duplicate audio files. The solution that can be done to handle duplication of audio files or duplicates is to apply hash type cryptography techniques where the process of identifying the hash value of the audio file is carried out so that the resulting value is different from the original file. The hash type cryptographic algorithm used is the MD4 algorithm. The results obtained from the process of applying the MD4 algorithm are grouping audio files based on the same hash value, making it easier and faster for users to delete duplicate audio files.

Keywords: Cryptography; Audio Files; MD4

1. PENDAHULUAN

Saat ini perkembangan teknologi sangatlah pesat yang saat ini telah memberikan banyak manfaat di berbagai bidang aspek. Penggunaan teknologi oleh manusia untuk menyelesaikan berbagai suatu pekerjaan merupakan suatu keharusan. Manusia sebagai pengguna teknologi haruslah mampu untuk memanfaatkannya yang ada saat sekarang ini, maupun perkembangan teknologi selanjutnya. Tentunya perubahan ini membawa banyak manfaat untuk kita dalam hal kecepatan dan akses data. Kelebihan lain dari teknologi komputer adalah dalam hal menyalin atau menggandakan file digital di mana hasilnya akan sama persis dengan file yang asli. Hal ini berdampak pada banyaknya file yang sama ataupun file duplikat yang tersimpan dalam satu komputer, hal ini tentunya tidak diperlukan dan menyita ruang penyimpanan. Sama halnya dengan file audio, akan sangat sulit membedakan file audio tersebut. Untuk itu diperlukan suatu cara yang dapat digunakan untuk mengindentifikasi file audio yang sama ataupun file audio yang duplikat.

Audio merupakan sinyal elektrik yang digunakan untuk membawa unsur bunyi, sehingga audio dalam komunikasi bercirikan suara. Audio juga sering kali diartikan dengan sistem yang berhubungan dengan proses perekaman dan transmisi sistem pengembalian/penangkapan suara, amplifier, sambungan transmisi pembawa bunyi[1]. Sekarang ini banyak sekali aplikasi yang membantu proses menduplikat audio, tentunya hal ini menyita lebih banyak ruang penyimpanan. Untuk itu diperlukan suatu cara yang dapat digunakan untuk mengindentifikasi file audio yang sama ataupun file audio yang duplikat. Sehingga dapat diambil keputusan untuk menghapus file audio yang duplikat tersebut sehingga dapat menghemat ruang penyimpanan.

Dalam kriptografi terdapat fungi hash yang merupakan fungsi satu arah yang dapat membangkitkan identitas sebuah file, di mana jika file audio itu isinya sama maka akan menghasilkan nilai hash yang sama pula. Hal ini dapat digunakan untuk mengindentifikasi file audio yang sama ataupun duplikat. Dalam fungsi hash terdapat Algoritma MD4 ataupun Message Digest 4 yang merupakan salah satu algoritma hashing yang sering digunakan untuk mengenkripsi data yang lebih aman dari algoritma sebelumnya. Dengan menerapkan algoritma ini pada aplikasi file scanner maka hasil dari pencarian file audio tersebut di kelompokkan berdasarkan nilai hash yang sama dan pengguna dapat memudahkan dan mempercepat pengguna untuk menghapus file audio yang duplikat tanpa harus membuka dan mendengarkan isi dari file audio tersebut.

Penelitian ini menguraikan tentang bagaimana mempermudah mencari file audio yang di duplikat tanpa harus membuka dan mendengarkan isi dari file audio tersebut, Dengan menerapkan algoritma MD4 (Message Digest 4). Langkag awal yang harus dilakukan adalah padding byte dalam satuan bit kongruen 448 modulo

512. Maka pesan tersebut kekurangan 64 bit untuk mencapai kelipatan 512. Setelah itu masukkan panjang pesan, pesan awal dilakukan dengan padding dan direpsentasikan 64 bit. Panjang pesan yang digunakan harus lebih kecil dari 264. Bila panjang pesan melebihi 264, maka pesan terendah yang dipresentasikan oleh 264 yang akan digunakan. Lalu proses selanjutnya inisialisasi penyangga MD digunakan untuk menghasilkan Message Digest, masing-masing variable ini mempresentasikan sebuah nilai 32-bit register. Kemudian proses pesan dalam blok 32-bit, untuk tahap ini proses pesan dilakukan sebanyak 32 byte.

Berdasarkan penelitian sebelumnya yang dilakukan oleh Lemcia Hutajulu di dalam penelitiannya algoritma MD4 digunakan untuk mendeteksi orisinalitas citra digital menyimpulkan bahwa hasil dari penerapan menggunakan algoritma MD4 dapat mempermudah mengetahui dalam mendeteksi orisinalitas citra digital apakah citra tersebut sudah

Jurnal Sains Dan Teknologi Informasi
Vol 1, No 4, September 2022, Hal 121-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v1i4.2293
Website https://ejurnal.seminar-id.com/index.php/jussi

dimanipulasi atau tidak[2].

Selanjutnya berdasarkan penelitian yang dilakukan oleh Gokma Lumbantoruan di dalam penelitiannya setelah dilakukan pendeteksian citra digital maka dapat membantu mengetahui apakah citra tersebut sudah di manipulasi atau tidak berdasarkan nilai hash dari masing-masing citra yang digunakan[3].

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Untuk mendukung kelancaran penelitian ini, maka dilakukan tahapan penelitian sebagai berikut:

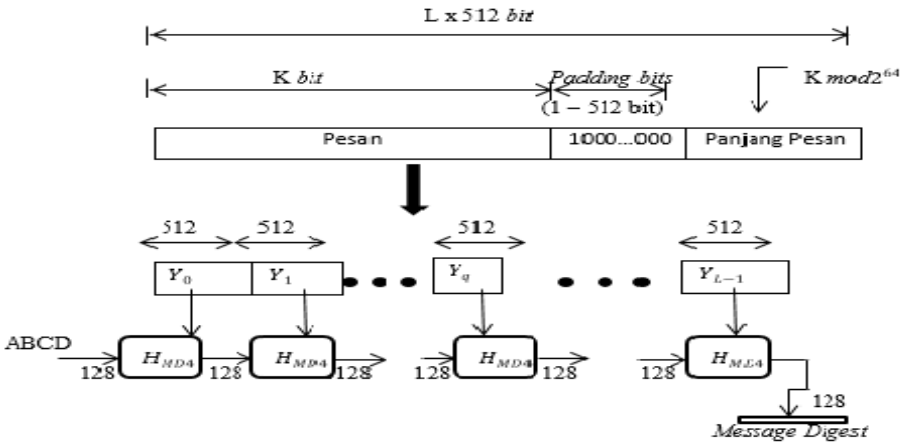
a. Studi pustaka
   Merupakan bentuk penelitian yang dilakukan berdasarkan kepustakaan atau buku-buku, jurnal, dan sumber-sumber sejenis (tertulis maupun dokumen) lain yang mempunyai hubungan dengan masalah yang sedang dibahas
b. Analisa dan Perancangan
   Analisa digunakan setelah data-data yang diperoleh dari studi pustaka kemudian dilakukan analisa sesuai dengan permasalahan penulis dalam penelitian ini
c. Perancangan dan Implementasi
   Tahap ini merupakan perancangan serta membangun aplikasi dan mengimplementasikan algoritma Algoritma SHA-256 yang digunakan dan aplikasi Duplicate Image Scanner yang dibangun pada penelitian ini..
d. Pengujian
   Pada tahap ini dilakukan Pengujian dari teori atau data yang telah dirancang dan diimplementasikan guna menguji kebenarannya.
e. Dokumentasi
   Pada tahap ini seluruh kegiatan dalam pembuatan sistem didokumentasikan kedalam bentuk tulisan berupa laporan penelitian.

2.2 Kriptografi

Kriptografi (*Cryptography*) berasal dari bahasa Yunani yaitu dari kata *cryptos*dan *graphia*yang berarti penulisan rahasia. Kriptografi adalah ilmu ataupun seni yang mempelajari bagaimana membuat sesuatu pesan yang dikirim oleh pengirim dapat disampaikan kepada penerima dengan aman. Kriptografi merupakan bagian dari suatu cabang ilmu matematika yang disebut kriptografi(*crytology*). Kriptografi bertujuan menjaga kerahasiaan informasi yang terkandung dalam data sehingga informasi tersebut tidak dapat diketahuioleh pihak yang tidak sah. Perancang algoritma kriptografi disebut kriptografer.

2.3 Metode MD4

MD4 menerima masukan berupa pesan atau *message* dengan ukuran sembarang dan menghasilkan *message digest* yang panjangnya 128 *bit*. Dengan panjang *message digest* 128 *bit*, secara *brute force* dibutuhkan percobaan sebanyak 2.128 kali untuk menemukan dua buah pesan atau lebih yang mempunyai *message digest* yang sama[5]. Gambaran pembuatan *message digest* dengan algoritma MD4 diperlihatkan pada gambar 1 :



Gambar 1. Pembuatan *message digest* dengan algoritma MD4.

Langkah-langkah pembuatan *message digest* secara garis besar adalah menambahkan *padding bits*, menambahkan nilai panjang pesan semula, menginisialisasi penyangga (*buffer*) MD, dan mengolah pesan dalam blok berukuran 512 *bit*[5].

a. Menambahkan *padding bits*.
b. Menambahkan nilai panjang pesan.

Jurnal Sains Dan Teknologi Informasi
Vol 1, No 4, September 2022, Hal 121-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v1i4.2293
Website https://ejurnal.seminar-id.com/index.php/jussi

c. Menginisialisasi penyangga MD
d. Mengolah pesan dalam blok berukuran 512 *bit*.

2.4 File Audio

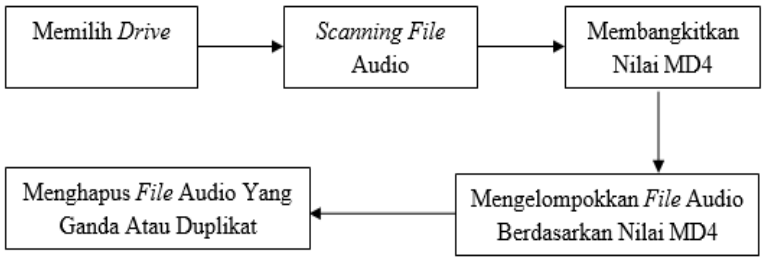
File audio (suara) adalah fenomena fisik yang dihasilkan oleh getaran suatu benda yang berupa sinyal analog dengan amplitudo yang berubah secara kontinu terhadap satuan waktu yang disebut frekuensi. Rekaman audio adalah bagaimana memastikan orisinalitas dari suara pelaku yang menjadi kunci dalam penyelidikan dan pengungkapan kasus. Definisi audio lainnya adalah merupakan salah satu elemen yang penting, karena ikut berperan dalam membangun sebuah sistem komunikasi dalam bentuk suara, ialah suat sinyal elektrik yang akan membawa unsur-unsur bunyi di dalamnya[1].

3. HASIL DAN PEMBAHASAN

3.1 Analisa Masalah

Masalah yang dihadapi ketika akan melakukan pencarian *file* audio yang duplikat atau ganda pada sebuah media penyimpanan membutuhkan ketelitian daningatan yang kuat dari penggunanya karena harus mendengarkan seluruh durasi dari*file* audio tersebut. Masalah ini dapat diatasi dengan cara memberikan identitas darisetiap *file* audio, sehingga ketika didapatkan *file* audio yang memiliki identitas yangsama maka *file* audio tersebut merupakan *file* audio yang duplikat atau ganda. Dalam penelitian ini cara yang digunakan untuk mendapatkan identitas *file* audio tersebut menggunakan fungsi *hash* MD4.

Langkah awal yang dilakukan ketika ingin melakukan pencarian *file* audio yang ganda atau duplikat adalah melakukan *scanning* pada sebuah ruang penyimpanan yang di dalamnya terdapat banyak *file* audio. Setelah melakukan proses *scanning* maka langkah selanjutnya adalah membangkitkan identitas dari *file*audio hasil *scanning* tersebut. Setelah identitas dari *file* audio tersebut berhasil di dapatkan maka langkah selanjutnya adalah melakukan pengelompokkan *file* audio berdasarkan identitas *file* audio yang di bangkitkan menggunakan fungsi *hash* MD4Untuk lebih jelasnya dapat dilihat pada gambar blok diagram di bawah ini:

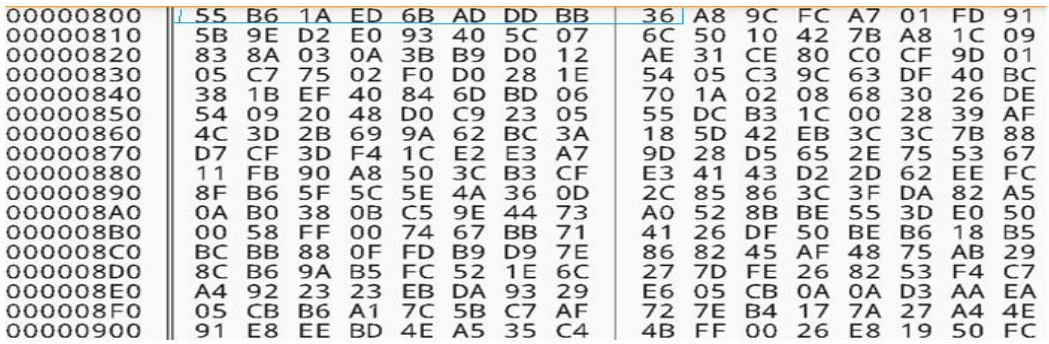


Gambar 2. Prosedur Pencarian *File* Audio Duplikat atau Ganda

3.2 Penerapan Algoritma MD4

Pada penerapan MD4 pada penelitan ini menggunakan objek *Audio* dengan spesifikasi, sebagai berikut:
Nama Audio : Sakura.mp3
Ekstensi/Type : *.MP3
Kapasitas : 3,98 Megabyte = 417.333.248 bytes
Durasi : 00:03:47
Bit rate : 1536 kbps

Proses ekstrasi nilai *Hexadecimal file* Sakura.mp3, penulis menggunakan aplikasi *Binary Viewer*, penulis mengambil nilai *hexadecimal* mulai dari *offset* 00000800 s/d 00000900 atau lebih jelasnya dapat dilihat pada gambar berikut ini:



00000800	55	B6	1A	ED	6B	AD	DD	BB	36	A8	9C	FC	A7	01	FD	91
00000810	5B	9E	D2	E0	93	40	5C	07	6C	50	10	42	7B	A8	1C	09
00000820	83	8A	03	0A	3B	B9	D0	12	AE	31	CE	80	C0	CF	9D	01
00000830	05	C7	75	02	F0	D0	28	1E	54	05	C3	9C	63	DF	40	BC
00000840	38	1B	EF	40	84	6D	BD	06	70	1A	02	08	68	30	26	DE
00000850	54	09	20	48	D0	C9	23	05	55	DC	B3	1C	00	28	39	AF
00000860	4C	3D	2B	69	9A	62	BC	3A	18	5D	42	EB	3C	3C	7B	88
00000870	D7	CF	3D	F4	1C	E2	E3	A7	9D	28	D5	65	2E	75	53	67
00000880	11	FB	90	A8	50	3C	B3	CF	E3	41	43	D2	2D	62	EE	FC
00000890	8F	B6	5F	5C	5E	4A	36	0D	2C	85	86	3C	3F	DA	82	A5
000008A0	0A	B0	38	0B	C5	9E	44	73	A0	52	8B	BE	55	3D	E0	50
000008B0	00	58	FF	00	74	67	BB	71	41	26	DF	50	BE	B6	18	B5
000008C0	BC	BB	88	0F	FD	B9	D9	7E	86	82	45	AF	48	75	AB	29
000008D0	8C	B6	9A	B5	FC	52	1E	6C	27	7D	FE	26	82	53	F4	C7
000008E0	A4	92	23	23	EB	DA	93	29	E6	05	CB	0A	0A	D3	AA	EA
000008F0	05	CB	B6	A1	7C	5B	C7	AF	72	7E	B4	17	7A	27	A4	4E
00000900	91	E8	EE	BD	4E	A5	35	C4	4B	FF	00	26	E8	19	50	FC

Gambar 2. Nilai *Hexadecimal File* Sakura.mp3

Jurnal Sains Dan Teknologi Informasi
Vol 1, No 4, September 2022, Hal 121-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v1i4.2293
Website https://ejurnal.seminar-id.com/index.php/jussi

Untuk melakukan contoh perhitungan secara manual penulis hanya menggunakan 9 *Byte* data sebagai *input* dalam perhitungan menggunakan algoritma MD4 yaitu 55 B6 1A ED 6B AD DD BB 36.

a. Penambahan *Padding Bit*

Untuk melakukan penambahan *Padding Bit* nilai masukkan di ubah ke dalam bentuk biner sebagai berikut :

Tabel 1. Konversi Heksadesimal ke Biner

Data Dalam Heksadesimal	Data Dalam Biner
55	01010101
B6	10110110
1A	00011010
ED	11101101
6B	01101011
AD	10101101
DD	11011101
BB	10111011
36	00110110

Penambahan *Padding Bit* dilakukan dengan persamaan sebagai berikut:

M= 448 Mod 512
72 = 448 Mod 512
L = (448-72) Mod 512
L = 376

Data sebanyak 9 *Byte* di atas diketahui 72 bit, untuk mencukupi 512 bit ditambah bit pengganjal *(padding bits)* sebanyak 376 bit yang di awali dengan bit 1 dan diikuti oleh bit 0 sebanyak 376 bit. Untuk lebih jelasnya dapat di lihat pada tabel 2. berikut ini :

Tabel 2. Penambahan *Padding Bit*

01010101	10110110	00011010	11101101	01101011	10101101	11011101	10111011
00110110	**10000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**
00000000	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**
00000000	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**
00000000	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**
00000000	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**
00000000	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**

b. Penambahan *Length* (Panjang Pesan)

Langkah kedua, adalah proses penambahan 64 bit nilai yang menyatakan panjang pesan semula. Jika panjang pesan lebih besar dari 2^{64} maka yang diambil adalah panjangnya dengan modulo 2^{64}. Dengan kata lain, jika panjang pesan semula adalah K bit, maka 64 bit yang ditambahkan menyatakan K modulo 2^{32}, proses ini dapat dinamakan *low-order word*. Setelah ditambah dengan 64 bit, panjang pesan menjadi 512 bit. Panjang data yang digunakan adalah 72 bit sehingga jika di ubah ke dalam bentuk biner maka hasilnya adalah 01001000. Untuk lebih jelasnya dapat di lihat pada tabel 3. berikut ini:

Tabel 3. Penambahan Panjang Pesan

01010101	10110110	00011010	11101101	01101011	10101101	11011101	10111011
00110110	10000000	00000000	00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000000	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**00000000**	**01001000**

c. *Initialize MD Buffer*

Langkah ketiga, melakukan proses inisialisasi penyangga (*buffer*) *message digest*. Empat kata *buffer* yang didefinisikan dengan A, B, C, D digunakan dalam melakukan komputasi *message digest*. Setiap dari A, B, C, D merupakan sebuah data *register* yang terdiri dari 32 bit. Total panjang penyangga adalah 128 *bit*. *Register- register* ini diinisialisasikan sebagai berikut:

A = 01 23 45 67
B = 89 AB CD EF
C = FE DC BA 98
D = 76 54 32 10

Jurnal Sains Dan Teknologi Informasi
Vol 1, No 4, September 2022, Hal 121-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v1i4.2293
Website https://ejurnal.seminar-id.com/index.php/jussi

d. Proses Kompresi MD4

Langkah ke 4 adalah melakukan proses kompresi menggunakan Fungsi f_F, f_G, f_H, untuk setiap 16 blok *word* yaitu pesan 512 bit yang di bagi menjadi 16 bagian dan setiap bagian terdiri dari 32 bit. Proses kompresi terdiri dari 3 ronde untuk setiap 16 blok *word* pesan. Untuk ronde pertama memproses pesan menggunakan fungsi f_F, ronde kedua memproses pesan dengan menggunakan fungsi f_G, dan pada ronde ketiga memproses pesan menggunakan fungsi f_H. Sebelum masuk ronde pertama maka pesan di bagi menjadi 16 blok word yaitu M 0-M 15 yang dapat di lihat pada tabel 4. berikut ini:

Tabel 4. 16 Blok *Word* Pesan

M 0	=	01010101	10110110	00011010	11101101	=	55B61AED
M 1	=	01101011	10101101	11011101	10111011	=	6BADDDBB
M 2	=	00110110	10000000	00000000	00000000	=	36000000
M 3	=	00000000	00000000	00000000	00000000	=	00000000
M 4	=	00000000	00000000	00000000	00000000	=	00000000
M 5	=	00000000	00000000	00000000	00000000	=	00000000
M 6	=	00000000	00000000	00000000	00000000	=	00000000
M 7	=	00000000	00000000	00000000	00000000	=	00000000
M 8	=	00000000	00000000	00000000	00000000	=	00000000
M 9	=	00000000	00000000	00000000	00000000	=	00000000
M 10	=	00000000	00000000	00000000	00000000	=	00000000
M 11	=	00000000	00000000	00000000	00000000	=	00000000
M 12	=	00000000	00000000	00000000	00000000	=	00000000
M 13	=	00000000	00000000	00000000	00000000	=	00000000
M 14	=	00000000	00000000	00000000	00000000	=	00000000
M 15	=	00000000	00000000	00000000	01001000	=	00000048

Setelah membagi pesan menjadi 16 bagian maka selanjutnya adalah masuk ke dalam ronde pertama dengan menggunakan fungsi f_F, ronde kedua dengan menggunakan fungsi f_g, dan ronde ketiga menggunakan fungsi f_h, fungsi kompresi algoritma MD4 dapat dilihat pada tabel 5. berikut ini:

Tabel 5. Fungsi MD4

Nama	Notasi	G(B, C, D)
f_F	F (B, C, D)	(B ∧ C) ∨ (~ B ∧ D)
f_g	G (B, C, D)	(B ∧ C) ∨ (C ∧ ~ D)
f_h	H (B, C, D)	B ⊕ C ⊕ D

Ronde 1
[A B C D 0 3]

I	A	B	C	D
0	01234567	89ABCDEF	FEDCBA98	76543210

A = (A + F(B,C,D) + M [i]) <<< S
A = (A + F(B,C,D) + M [0]) <<< 3
A = (01234567 + F(B,C,D) + 55B61AED) <<< 3
F (B,C,D) = (B ∧ C) ∨ (~ B ∧ D)
= (89ABCDEF ∧ FEDCBA98) ∨ (~ 89ABCDEF ∧ 76543210)
= (88888888) ∨ (76543210)
= FEDCBA98
A = (01234567 + FEDCBA98 + 55B61AED) <<< 3
A = (55B61AEC) <<< 3
A = ADB0D760
Maka nilai MD *Buffer* menjadi :

I	A	B	C	D
0	ADB0D760	89ABCDEF	FEDCBA98	76543210

[D A B C 1 7]
A=D, B=A, C=B, D=C

I	A	B	C	D
1	76543210	ADB0D760	89ABCDEF	FEDCBA98

A = (A + F(B,C,D) + M [i]) <<< S
A = (A + F(B,C,D) + M [1]) <<< 7
A = (76543210 + F(B,C,D) + 6BADDDBB) <<< 7

Jurnal Sains Dan Teknologi Informasi
Vol 1, No 4, September 2022, Hal 121-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v1i4.2293
Website https://ejurnal.seminar-id.com/index.php/jussi

F (B,C,D) = (B ∧ C) ∨ (~ B ∧ D)
= (ADB0D760 ∧ 89ABCDEF) ∨ (~ ADB0D760 ∧ FEDCBA98)
= (89A0C560) ∨ (524C2898)
= DBECEDF8

A = (76543210 + DBECEDF8 + 6BADDDBB) <<< 7
A = (BDEEFDC3) <<< 7
A = F77EE180

Maka nilai MD *Buffer* menjadi :

I	A	B	C	D
1	F77EE180	ADB0D760	89ABCDEF	FEDCBA98

Tabel 6. Hasil Penyambungan bit di A B C dan D

Int	A	B	C	D
T_0	01234567	89ABCDEF	FEDCBA98	76543210
T_1	F77EE180	ADB0D760	89ABCDEF	FEDCBA98
T_2	74403800	F77EE180	ADB0D760	89ABCDEF
T_3	AA780000	74403800	F77EE180	ADB0D760
T_4	17BDC700	AA780000	74403800	F77EE180
T_5	FB8CC000	17BDC700	AA780000	74403800
T_6	E7C00000	FB8CC000	17BDC700	AA780000
T_7	38000000	E7C00000	FB8CC000	17BDC700
T_8	DA543800	38000000	E7C00000	FB8CC000
T_9	86600000	DA543800	38000000	E7C00000
T_{10}	00000000	86600000	DA543800	38000000
T_{11}	C0000000	00000000	86600000	DA543800
T_{12}	05A1C000	C0000000	00000000	86600000
T_{13}	30000000	05A1C000	C0000000	00000000
T_{14}	00000000	30000000	05A1C000	C0000000
T_{15}	02400000	00000000	30000000	05A1C000
T_{16}	3FB0D768	00000000	30000000	05A1C000
T_{17}	48000000	3FB0D768	00000000	30000000
T_{18}	61AED000	48000000	3FB0D768	00000000
T_{19}	00000000	61AED000	48000000	3FB0D768
T_{20}	686C2918	00000000	61AED000	48000000
T_{21}	00000000	686C2918	00000000	61AED000
T_{22}	35F23000	00000000	686C2918	00000000
T_{23}	00000000	35F23000	00000000	686C2918
T_{24}	A2F2C810	00000000	35F23000	00000000
T_{25}	00000000	A2F2C810	00000000	35F23000
T_{26}	C9F18000	00000000	A2F2C810	00000000
T_{27}	00000000	C9F18000	00000000	A2F2C810
T_{28}	67224600	00000000	C9F18000	00000000
T_{29}	00000000	67224600	00000000	C9F18000
T_{30}	C9F18000	00000000	67224600	00000000
T_{31}	00000000	C9F18000	00000000	67224600
T_{32}	244F0768	C9F18000	00000000	67224600
T_{33}	C19AD000	244F0768	C9F18000	00000000
T_{34}	22BB4000	C19AD000	244F0768	C9F18000
T_{35}	0BB40000	22BB4000	C19AD000	244F0768
T_{36}	1724BB40	0BB40000	22BB4000	C19AD000
T_{37}	8D968000	1724BB40	0BB40000	22BB4000
T_{38}	0BDA0000	8D968000	1724BB40	0BB40000
T_{39}	1DA00000	0BDA0000	8D968000	1724BB40
T_{40}	F5F8C7D8	1DA00000	0BDA0000	8D968000
T_{41}	328FB000	F5F8C7D8	1DA00000	0BDA0000
T_{42}	8BBEC000	328FB000	F5F8C7D8	1DA00000
T_{43}	DBEC0000	8BBEC000	328FB000	F5F8C7D8
T_{44}	C6B1BEC0	DBEC0000	8BBEC000	328FB000
T_{45}	E65D8000	C6B1BEC0	DBEC0000	8BBEC000
T_{46}	F7F60000	E65D8000	C6B1BEC0	DBEC0000
T_{47}	1F840000	F7F60000	E65D8000	C6B1BEC0

Jurnal Sains Dan Teknologi Informasi
Vol 1, No 4, September 2022, Hal 121-127
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v1i4.2293
Website https://ejurnal.seminar-id.com/index.php/jussi

Setelah putaran ke-47 A B C dan D ditambahkan ke- A B C dan D

	A	B	C	D
T_{47}	1F840000	F7F60000	E65D8000	C6B1BEC0
Initial Hash Value	01234567	89ABCDEF	FEDCBA98	76543210
Hasil	20A74567	81A1CDEF	E53A3A98	3D05F0D0

Maka hasil dari proses penerapan MD4 untuk mendapatkan kode *hash* dari audio sebagai berikut. Jadi hasil *hash* yang di dapat dari *file* audio *.MP3 adalah.
Hasil *Hash* : 20A74567 81A1CDEF E53A3A98 3D05F0D0

4. KESIMPULAN

Berdasarkan analisa yang telah diuraikan pada bab-bab sebelum nya dan juga berdasarkan hasil pengamatan penulis dari rumusan masalah, maka dapat diambil kesimpulan dimana setelah melakukan proses mengindentifikasi file audio dengan cara mengelompokkan nilai hash yang sama. Maka proses mengindentifikasi file audio dapat berjalan sesuai dengan hasil hash audio asli dengan yang telah diduplikat atau digandakan. Setelah menerapkan metode MD4 untuk mengindentifikasi file audio maka seluruh file audio yang memiliki nilai hash yang sama akan tampil. Hasil pengecekan berdasarkan hasil pungsi hash file audio asli yang dibandngkan dengan hasil pada pungsi hash file audio yang sama.

REFERENCES

[1] S. U. Lubis, P. Studi, and T. Informatika, “IMPLEMENTASI METODE MD5 UNTUK MENDETEKSI ORISINALITAS,” vol. 3, pp. 402–408, 2019.
[2] P. Metode, M. D. Dan, and S. H. A. Untuk, “Perbandingan metode md4 dan sha 384 untuk mendeteksi orisinalitas citra digital,” vol. 3, pp. 243– 253, 2019.
[3] G. Lumbantoruan, H. Sunandar, and I. Saputra, “PERBANDINGAN METODE MD2 DAN MD4 UNTUK MENDETEKSI KEASLIAN,” vol. 3, pp. 385–391, 2019.
[4] nuraini Indah kusuma dewi, okta veza, “Analisis dan implementasi sistem informasi penjualan berbasis web pada ukm tiara cakery batam,” Responsive, vol. 2, no. 2614–7602, p. 30, 2018.
[5] P. Publik, D. Bidang, S. Di, and K. Makassar, “No Title,” 2009.
[6] M. K. Emy Setyaningsih, S.Si., Kriptografi dan Implementasinya Menggunakan Matlab. Yogyakarta: CV. ANDI OFFSET, 2015.
[7] R. Sadikin, Kriptografi Untuk Kemanan Jaringan dan Implementasinya Dalam Bahasa Java, Yogyakarta: Andi, 2012.
[8] R. Munir, Kriptografi, Bandung: R. Munir, 2006.
[9] Y. Kurniawan, Kriptografi Keamanan Internet dan Jaringan Komunikasi, Bandung: Informatika, 2017.
[10] R. Munir, Kriptografi, Bandung: Informatika Bandung, 2006.
[11] D. Rachamawati, J. T. Tarigan and A. B. C. Ginting, "A Comparative Study of Message Digest 5 (MD5) and SHA-256 Algorithm," in 2nd International Conference on Computing and Applied Informatics 2017, Medan, 2018.
[12] F. I. P. Standards Publications, "Secure Hash Standard," National Institute of Standards and Technology, Amerika Serikat, 2002.
[13] S. Patil, N. Jagtap, S. Rajput and R. Sangore, "A Duplicate File Finder System," International Journal of Science Spirituality Business and Technology, pp. 10-14, 2017.