ISSN : 1411-1624
e-ISSN: 2621-8089

PENGAMANAN CITRA DIGITAL REKAM MEDIS MENGGUNAKAN PERPADUAN *HASHING ALGORITMA KECCAK* DAN *RIVEST CODE 6*

Henki Bayu Seta[1], Risma Yulistiani[2], Theresiawati[3]
Dosen Universitas Pembangunan Nasional Veteran Jakarta [1,3],
Mahasiswa Universitas Pembangunan Nasional Veteran Jakarta [2]
Jalan RS. Fatmawati Jakarta Selatan
Sur-el : henkiseta@upnvj.ac.id [1], yulistianir@gmail.com[2], theresiawati@upnvj.ac.id[3]

Abstract : *In line with advances in technology that are increasingly developing, allowing data and information to be accessed and manipulated by anyone. The leakage of personal information is very troubling when it comes to such important matters as data and medical record information. One of the medical records that is prone to data leakage is an image of a medical record, one of which is a PET scan. This research uses a combination of Keccak and Rivest Code 6 (RC6) hashing algorithm to maintain the confidentiality and security of the digital image of PET Scan medical records. Based on testing by modifying the encrypted files, this research resulted in the RC 6 algorithm being able to maintain image security, the modified files cannot be decrypted back into the original image. The Keccak algorithm is proven to be able to maintain the integrity of digital images of medical records. The testing process in this study was carried out by comparing the image size and histogram of the original image and the image resulting from the encryption and decryption process. This research test produces different hash values of the two images even though the modified image is very similar to the original image..*

Keywords: *PET Scan, Keccak Algorithm, RC6 Algorithm, Encryption, Decryption, Hashing*

Abstrak : *Selaras dengan kemajuan teknologi yang semakin berkembang, memungkinkan data dan informasi dapat diakses dan dimanipulasi oleh siapa saja. Kebocoran informasi pribadi sangat meresahkan bila terkait hal yang begitu penting seperti data dan informasi rekam medis. Salah satu data rekam medis yang rentan akan kebocoran data adalah citra rekam medis, salah satunya PET Scan. Penelitian ini menggunakan perpaduan hashing algoritma Keccak dan Rivest Code 6 (RC6) untuk menjaga kerahasiaan dan keamanan citra digital rekam medis PET Scan. Berdasarkan pengujian dengan memodifikasi file enkripsi, penelitian ini menghasilkan algoritma RC 6 mampu menjaga keamanan citra, file hasil modifikasi tidak dapat di dekripsi kembali menjadi citra asli. Algoritma Keccak terbukti mampu menjaga integrity citra digital rekam medis. Proses pengujian pada penelitian ini dilakukan dengan cara membandingkan ukuran citra dan histogram dari citra asli dan citra hasil proses enkripsi dan dekripsi. Pengujian penelitian ini menghasilkan nilai hash kedua citra berbeda meskipun citra modifikasi sangat mirip dengan citra asli.*

Kata kunci*: PET Scan, Algoritma Keccak, Algoritma RC6, Enkripsi, Dekripsi, Hashing*

1. PENDAHULUAN

Perkembangan *telemedicine* saat ini, sangat memudahkan para dokter untuk dapat saling berbagi citra medis, sehingga diagnosa penyakit pasien dapat dilakukan secara tepat dan cepat. Citra digital rekam medis yang digunakan merupakan hasil rekam medis PET (*Positron Emission Tomography*). PET Scan memiliki keunggulan yang lebih dibandingkan CT maupun MRI scan, dimana CT dan MRI hanya menunjukkan gambaran anatomi, sedangkan PET dapat menunjukkan gambaran fungsional dan anatomi tubuh pasien [1]. Hasil rekam medis

ISSN : 1411-1624
e-ISSN: 2621-8089

PET digunakan sebagai salah satu cara untuk mendiagnosis penyakit pasien, terutama penyakit ganas seperti kanker dan penyakit dalam.

Institusi kesehatan di seluruh dunia perlu bertukar dan berbagi informasi medis, namun informasi yang dipertukarkan harus diamankan agar tidak adanya modifikasi yang dapat terjadi pada informasi yang dikirimkan lewat internet[2]. Pentingnya menjaga keaslian dan keamanan citra merupakan integritas pada rekam medis, yaitu untuk mengacu kepercayaan, konsistensi dan keyakinan terhadap rekam medis[3]. Keamanan citra medis telah menjadi masalah penting ketika citra ditransmisi lewat jaringan terbuka seperti internet, di mana informasi sensitif pasien dapat diakses oleh peretas yang bermaksud jahat, kemungkinan pelanggaran keamanakan termasuk merusak citra untuk dimodifikasi atau dimasukan data palsu yang dapat menyebabkan kesalahan diagnosis dan pengobatan[4]. Hasil pemeriksaan tersebut tentu sangat penting bagi seorang pasien, sehingga perlu dijaga kerahasiaan dan keasliannya. Untuk mencegah kebocoran dan manipulasi data oleh pihak yang tidak berwenang, perlu adanya upaya pengamanan terhadap citra rekam medis.

Pengamanan citra rekam medis dapat dilakukan dengan menggabungkan aspek Confidentiality dan Integrity pada teknik kriptografi, yaitu dengan menerapkan *Algoritma Rivest Code 6* dalam proses enkripsi dan dekripsi citra, serta dipadukan dengan *Algoritma Keccak* sebagai fungsi nilai hash yang dapat menjaga *integrity* atau keaslian citra.

Berdasarkan penelitian yang dilakukan oleh Prayudi dkk tahun 2005 menghasilkan adanya fungsi f(x) = x(2x+1) dan pergeseran 5 bit ke kiri memberikan tingkat keamanan data yang tinggi. Serta avalanche effect juga memberikan kesulitan kepada kriptanalis untuk melakukan serangan[5]. Namun terdapat kekurangan jika hanya menerapkan *algoritma RC6* saja, yaitu hasil proses enkripsi dan dekripsi tidak dapat membuktikan bahwa citra rekam medis tersebut adalah file citra yang asli. Untuk itu dibutuhkan *Algoritma Keccak* dalam upaya menjaga keaslian citra yang dapat dipercaya. *Algoritma Keccak* akan membuat nilai hash yang berfungsi sebagai nilai integrity dari sebuah citra rekam medis. Penulis menggunakan *algoritma Keccak* yang merupakan pembaharuan dari algoritma MD5 untuk mengamankan integrity dengan baik. Sedangkan untuk algoritma simetris, penulis menggunakan algoritma *Rivest Code 6* sebagai pembaharuan algoritma menggantikan AES. Pada kegiatan ini, penulis menggabungkan aspek *confidential* dan *integrity* pada kriptografi secara bersamaan untuk mengamankan citra rekam medis. Penelitian ini bertujuan untuk mengetahui tingkat keberhasilan perpaduan *algoritma Keccak* dan RC6 untuk mengamankan citra digital PET Scan.

2. METODOLOGI PENELITIAN

2.1 *Algoritma Rivest Code 6 (RC6)*

Pengembangan dari algoritma RC-5 yang dilengkapi dengan beberapa parameter. Dirancang oleh Ronald D. Rivest, M.J.B. Robshaw, R. Sidney dan Y.I. Yin, sebagai

ISSN : 1411-1624
e-ISSN: 2621-8089

kandidat Advanced Encryption Standard (AES) yang diajukan kepada NIST oleh RSA Laboratories. RC6 sering dituliskan RC6-w/r/b, dimana parameter w merupakan ukuran kata dalam suatu bit, r adalah bilangan bulat (bukan negatif) yang menunjukan banyaknya iterasi selama proses enkripsi dan b menunjukan ukuran kunci enkripsi dalam byte [6]. Setelah algoritma ini masuk kandidat AES, maka ditetapkan bahwa nilai w=32, r=20, dan b bervariasi antara 16, 24 dan 32 byte [7]. Ukuran kunci algoritma RC6 beragam diantara 128, 192 dan 256 bit. Aturan memecah blok 128bit menjadi 4 blok 32-bit dalam RC6 dapat dilihat sebagai berikut.

- a+b operasi penjumlahan bilangan integer
- a-b operasi pengurangan bilangan integer
- a⊕b operasi eksklusif-OR (XOR)
- a×b operasi perkalian bilangan integer
- a⋘b, a dirotasikan ke kiri sebanyak variabel kedua (b)
- a⋙ b, a dirotasikan ke kanan sebanyak variabel kedua (b)

Enkripsi Algoritma Rivest Code 6 (RC6)

Algoritma RC6 memecah block 128 bit menjadi 4 buah block 32 bit, maka RC6 bekerja dengan 4 buah block 32-bit A, B, C, D. Byte yang pertama dari *plaintext* atau *ciphertext* ditempatkan pada byte A dan byte terakhir ditempatkan pada byte D. Maka hasil prosesnya akan didapat (A, B, C, D) = (B, C, D, A) yang diartikan bahwa nilai di sisi kanan berasal dari register di sisi kiri[5].

Algoritma RC6 memiliki 44 buah sub kunci yang dibangkitkan dari kunci dan dinamakan S[0] sampai S[43], masing-masing panjangnya 32 bit. Proses enkripsi pada algoritma RC6 dimulai dengan *whitening* yang bertujuan untuk menyamarkan iterasi yang pertama dan terakhir pada proses iterasi. Pada proses whitening awal, nilai B akan dijumlahkan dengan S[0], dan nilai D dijumlahkan dengan S[i]. Pada masing-masing iterasi pada RC6 menggunakan 2 buah sub kunci. Sub kunci pada iterasi yang pertama menggunakan S[2] dan S[3], sedangkan iterasi-iterasi berikutnya menggunakan sub-sub kunci lanjutannya. Setelah iterasi ke-20 selesai, dilakukan proses whitening akhir dimana nilai A dijumlahkan dengan S[42], dan nilai C dijumlahkan dengan S[43] [5].

Iterasi pada algoritma RC6 mengkuti aturan, nilai B dimasukan ke dalam fungsi f, yang di definisikan sebagai f(x)=x(2x+1), kemudian di putar ke kiri sejauh lg-w atau 5-bit. Hasilnya dimisalkan sebagai u. Nilai ini kemudian melalui proses XOR dengan C dan hasilnya menjadi nilai C. Nilai baru C, dimisalkan sebagai t, yang kemudian menjadi acuan C untuk memutar nilainya ke kiri, dan nilai u dijadikan acuan nilai A untuk melakukan proses perputaran ke kiri. Lalu sub kunci S[2i] pada iterasi dijumlahkan dengan A, dan sub kunci S[2i+1] dijumlahkan dengan C. Keempat bagian dari block kemudian dipertukarkan dengan mengikuti aturan seperti di atas, bahwa nilai A ditempatkan pada D, nilai B pada A dan nilai C ditempatkan di B, kemudian nilai asli D ditempatkan pada C. Aturan terus diulang sampai iterasi ke 20 kali [5].

```
B = B + S[ 0 ]
D = D + S[ 1 ]
 for i = 1 to 20 do  {
```

ISSN : 1411-1624
e-ISSN: 2621-8089

```
    t = ( B x ( 2B + 1 ))⋘ 5
    u = ( D x ( 2D + 1 ))⋘ 5
    A = ( ( A ⊕ t ) ⋘ u ) + S[ 2i ]
    C = ( ( C ⊕ u ) ⋘ t ) + S[ 2i + 1 ]
    (A,B,C,D) = (B,C,D,A)
    }
A = A + S[ 42 ]
C = C + S[ 43 ]
```

Deskripsi Algoritma Rivest Code 6 (RC6)

Proses dekripsi pada RC6 menggunakan rumus dan cara yang sama dengan proses enkripsi, hanya membalik prosesnya, dan jika dalam proses enkripsi menggunakan penjumlahan, sedangkan pada dekripsi menggunakan pengurangan. Sub kunci pada proses *whitening* setelah iterasi terakhir diterapkan sebelum iterasi pertama, begitu pula sebaliknya sub kunci yang diterapkan pada proses whitening sebelum iterasi pertama digunakan proses whitening setelah iterasi terakhir[5].

```
C = C - S[ 43 ]
A = A - S[ 42 ]
for i = 20 downto 1 do {
    (A,B,C,D) = (D,A,B,C)
    u = ( D x ( 2D + 1 ))⋘ 5
    t = ( B x ( 2B + 1 ))⋘ 5
    C = ( ( C - S[ 2i + 1 ] )⋙ t ) ⊕ u
    A = ( ( A - S[ 2i ] ) ⋙ u ) ⊕ t
  }
D = D - S[ 1 ]
B = B - S[ 0 ]
```

Pembangkitan kunci RC6

Pengguna memasukkan sebuah kunci yang besarnya b byte, dimana 0≤b≤255. byte kunci ini kemudian ditempatkan dalam array c w-bit words L[0]…L[c-1]. Byte pertama kunci akan ditempatkan sebagai pada L[0], byte kedua pada L[1], dan seterusnya. (Catatan, bila b=0 maka c=1 dan L[0]=0). Masing-masing nilai kata w-bit akan dibangkitkan pada penambahan kunci round 2r+4 dan akan ditempatkan pada array S[0,…,2r+3].

Konstanta Pw = B7E15163 dan Qw = 9E3779B9 (dalam satuan hexadecimal) adalah konstanta yang digunakan dalam penjadwalan kunci pada RC6. Nilai P32 diperoleh dari perluasan bilangan biner e-2, dimana e adalah sebuah fungsi logaritma. Sedangkan nilai Q32 diperoleh dari perluasan bilangan biner ∅-1, dimana ∅ dapat dikatakan sebagai “golden ratio” (rasio emas) (Halik and Prayudi, 2005). Algoritma untuk pembangkitan kunci RC6 adalah sebagai berikut:

```
S[ 0 ]=Pw
for i = 1 to 43 do S[i]= S[i-1]+ Qw
    A = B = i = j = 0
for k = 1 to 132 do {
    A = S[ i ]= ( S[ i ]+ A + B )⋘ 3
    B = L[ j ]= ( L[ j ]+ A + B )⋘ ( A + B )
    i = ( i + 1 ) mod 44    j = ( j + 1 )
    mod c       }
```

2.2 Fungsi *Hash*

Fungsi *hash* adalah fungsi yang menerima string input yang panjangnya sembarang dan mengubahnnya menjadi string yang memiliki panjang tetap, dan umumnya menjadi lebih kecil daripada input. Output dari fungsi hash adalah fungsi satu arah yang dapat menghasilkan signature pada data. Perubahan bit sedikit saja dapat mengubah output dari fungsi *hash*. Fungsi

ISSN : 1411-1624
e-ISSN: 2621-8089

hash biasanya digunakan untuk memastikan integrity dari data dan tanda tangan digital [8].

2.3 Algoritma Keccak

Keccak adalah salah satu algoritma fungsi *hash* berdasarkan kontruksi *spons* menggunakan fungsi permutasi *f-Keccak* dengan panjang permutasi *b* tetap, di mana:

$$b = 25 \times 2^l$$

Dengan:

$$0 \leq l \leq 6$$

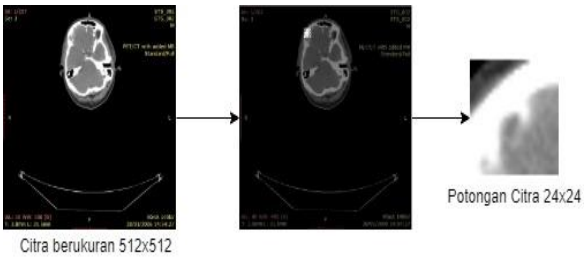
Algoritma *Keccak* memiliki prinsip sama seperti algoritma *cipher block*, dimana proses dilakukan terhadap blok-blok, setiap hasil proses bergantung dari masukan dan hasil proses bergantung pada masukan sebelumnya, namun pada algoritma *Keccak* tidak memiliki *key schedule*, menggunakan konstanta *round* yang bersifat tetap dari pada *round key* [9].

f-Keccak adalah permutasi lebih dari $s \in Z_2^b$, dimana $0 \leq s \leq b-1$. b disebut sebagai panjang permutasi, pada kandidat *SHA3* digunakan, *f-Keccak[1600]*. Secara resmi, fungsi permutasi *Keccak* yang disepakati adalah *Keccak-f* [1600], di mana l = 6 atau $b = 25 \times 2^6$. Algoritma *Keccak* bekerja berdasarkan fungsi spon yang terdiri dari *padding*, *absorbing* dan *squeezing* dan menerima tiga parameter masukan, yaitu *bitrate (r), capacity (c),* dan *difersity (d).* Kontruksi ini dapat dikatakan sebagai alur proses kerja algoritma *Keccak* dari *input* sampai menghasilkan *output*, yang nanti di dalam kontruksi ini terjadi beberapa proses fungsi permutasi untuk menghasilkan *hashing*.

Dalam sekuel permutasi *f-Keccak* [*b*] didesripsikan pada keadaan *a* yang merupakan array 3 dimensi pada elemen GF(2), dinamakan $a = [5][5][w]$, dengan $w = 2^l$. Pemetaan antara bit s dan bit a adalah $s[w(5y+x)+z] = a[x][y][z]$. Pendeskripsian $a[x][y][z]$ dengan $x, y \in \mathbb{Z}_5$ dan $x, y \in \mathbb{Z}_w$, dinotasikan posisi bit (x, y, z). Oleh karena itu, pengindeksan dimulai dari 0, ekspresi dalam koordinat *x* dan *y* harus diambil modulo 5 dan dalam koordinat *z* modulo *w*. Terkadang indeks [*z*] dapat dihilangkan, baik indeks [*y*][*z*] atau ketiga indeks menyiratkan bahwa pernyataan tersebut valid untuk semua indeks yang dihilangkan. *f-Keccak* adalah permutasi berulang, terdiri dari urutan putaran n_r, diindeks dengan i_r dari 0 sampai 1 $n_r - 1$ [10].

3. HASIL DAN PEMBAHASAN

Penelitian ini menggunakan citra digital rekam medis sebanyak 26 citra digital hasil PET Scan diperoleh melalui website https://www.cancerimagingarchive.net/ yang dapat diakses secara umum untuk kepentingan penelitian[11]. untuk memudahkan proses analisis, citra berukuran 512x512 pixel terlebih dahulu dipotong menjadi citra berukuran 24x24 pixel serta diurai sehingga akan didapatkan nilai pixel kemudian dikonversi ke bentuk hexadecimal, seperti terlihat pada gambar 1.



Gambar 1. Ilustrasi pemotongan citra

ISSN : 1411-1624
e-ISSN: 2621-8089

Setelah dilakukan pemotongan citra, setiap nilai pada 1 pixel ditampilkan dalam bentuk bilangan decimal, seperti terlihat pada gambar 2

0	0	0	0	31	54	66	56	57	49	45	63	82	97	116	173	255	254	255	254	255	255	255	255
0	0	0	45	74	61	54	42	39	47	66	85	123	175	255	255	255	255	255	255	255	255	255	255
0	2	51	47	47	41	44	47	54	81	114	162	255	255	255	255	255	254	255	255	253	255	255	255
0	63	73	50	47	55	66	97	105	139	220	255	255	255	255	255	255	255	255	255	255	255	255	255
61	64	74	70	62	85	100	132	180	255	255	255	255	255	255	255	255	255	254	255	255	255	255	255
63	62	63	76	89	107	132	197	255	255	255	255	255	255	255	255	255	254	255	254	229	210	203	193
64	63	75	96	116	138	234	255	255	255	254	255	254	255	255	255	255	244	194	173	164	166	162	155
59	68	91	113	159	255	255	255	255	255	255	255	255	254	255	255	219	182	164	154	150	160	153	146
63	87	116	166	255	255	255	254	255	255	255	255	255	254	255	230	195	179	166	156	159	159	146	150
97	117	188	255	255	254	255	255	255	255	255	255	254	255	233	197	174	163	157	154	152	153	147	
125	198	254	255	255	255	255	255	255	255	255	255	255	255	255	247	202	168	158	161	142	137	152	153
202	254	255	255	255	255	255	255	255	245	225	194	181	221	252	236	196	172	151	146	136	143	143	139
254	255	255	255	255	255	254	254	240	189	162	143	136	189	228	218	184	156	150	142	139	146	141	139
255	255	255	255	255	255	255	255	205	174	137	127	144	177	194	188	170	146	141	150	143	136	128	141
254	255	255	255	255	255	255	254	247	193	151	127	132	152	170	171	164	151	143	128	143	138	132	131	126
255	255	255	255	255	255	255	255	242	185	146	130	137	157	163	153	141	141	152	132	135	141	130	129	123
255	255	255	255	255	255	255	255	245	186	160	148	148	163	162	146	135	131	141	140	126	140	138	126	123
255	255	254	255	255	255	255	255	255	183	147	150	164	169	158	146	135	140	131	137	127	139	139	131	132
255	255	255	255	253	255	254	247	178	141	138	153	149	141	143	136	140	125	125	129	139	131	135	142
254	255	255	254	254	255	255	233	175	155	135	135	134	137	134	127	137	135	128	128	127	128	142	135
255	255	255	231	230	255	255	235	184	160	145	130	135	141	145	130	134	150	135	133	125	123	124	130
255	255	246	204	206	236	239	226	192	151	141	133	131	132	146	146	137	144	131	136	138	130	125	139
255	255	240	211	217	227	219	205	187	159	142	138	137	137	130	138	148	146	141	137	131	134	133	136
255	255	255	226	208	207	197	192	184	174	154	146	137	136	124	127	147	135	139	142	130	116	118	125

Gambar 2. Nilai pixel potongan citra 24x24 dalam bentuk Decimal

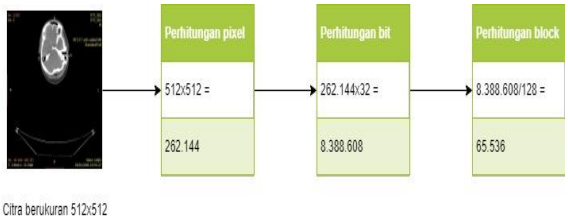
Selanjutnya dilakukan konversi hexadecimal terhadap seluruh nilai pixel pada potongan citra 24x24 pixel. Potongan citra akan dibagi sepanjang 128 bit atau 16 byte. Panjang 1 byte sama dengan 1 pixel. Hasil konversi *hexadecimal* dapt dilihat pada gambar 3.

```
00000000    FF D8 FF E0 00 10 4A 46 - 49 46 00 01 01 00 00 01    ÿØÿà..JFIF......
00000010    00 01 00 00 FF DB 00 43 - 00 01 01 01 01 01 01 01    ....ÿÛ.C........
00000020    01 01 01 01 01 01 01 01 - 01 01 01 01 01 01 01 01    ................
00000030    01 01 01 01 01 01 01 01 - 01 01 01 01 01 01 01 01    ................
00000040    01 01 01 01 01 01 01 01 - 01 01 01 01 01 01 01 01    ................
00000050    01 01 01 01 01 01 01 01 - 01 FF DB 00 43 01 01 01    .........ÿÛ.C...
00000060    01 01 01 01 01 01 01 01 - 01 01 01 01 01 01 01 01    ................
00000070    01 01 01 01 01 01 01 01 - 01 01 01 01 01 01 01 01    ................
00000080    01 01 01 01 01 01 01 01 - 01 01 01 01 01 01 01 01    ................
00000090    01 01 01 01 01 01 01 01 - 01 01 01 01 01 01 FF C0    ..............ÿÀ
000000A0    00 11 08 02 00 02 00 03 - 01 22 00 02 11 01 03 11    ................
000000B0    01 FF C4 00 1F 00 00 01 - 05 01 01 01 01 01 01 00    .ÿÄ.............
000000C0    00 00 00 00 00 00 00 01 - 02 03 04 05 06 07 08 09    ................
000000D0    0A 0B FF C4 00 B5 10 00 - 02 01 03 03 02 04 03 05    ..ÿÄ.µ..........
000000E0    05 04 04 00 00 01 7D 01 - 02 03 00 04 11 05 12 21    ..........}.....!
000000F0    31 41 06 13 51 61 07 22 - 71 14 32 81 91 A1 08 23    1A..Qa..q.2.....
00000100    42 B1 C1 15 52 D1 F0 24 - 33 62 72 82 09 0A 16 17    B.Á.RÑð.3br.....
00000110    18 19 1A 25 26 27 28 29 - 2A 34 35 36 37 38 39 3A    .....456789.
00000120    43 44 45 46 47 48 49 4A - 53 54 55 56 57 58 59 5A    CDEFGHIJSTUVWXYZ
00000130    63 64 65 66 67 68 69 6A - 73 74 75 76 77 78 79 7A    cdefghijstuvwxyz
00000140    83 84 85 86 87 88 89 8A - 92 93 94 95 96 97 98 99    ƒ......Š.......
00000150    9A A2 A3 A4 A5 A6 A7 A8 - A9 AA B2 B3 B4 B5 B6 B7    š.....¢£¤¥¦§¨©ª².µ..
00000160    B8 B9 BA C2 C3 C4 C5 C6 - C7 C8 C9 CA D2 D3 D4 D5    .¹ºÂÃÄÅÆÇÈÉÊÒÓÔÕ
00000170    D6 D7 D8 D9 DA E1 E2 E3 - E4 E5 E6 E7 E8 E9 EA F1    Ö.ØÙÚáâãäåæçèéêñ
00000180    F2 F3 F4 F5 F6 F7 F8 F9 - FA FF C4 00 1F 01 00 03    òóôõö.øùúÿÄ.....
00000190    01 01 01 01 01 01 01 01 - 01 00 00 00 00 00 00 01    ................
000001A0    02 03 04 05 06 07 08 09 - 0A 0B FF C4 00 B5 11 00    ..........ÿÄ.µ..
000001B0    02 01 02 04 04 03 04 07 - 05 04 04 00 01 02 77 00    ..............w.
000001C0    01 02 03 11 04 05 21 31 - 06 12 41 51 07 61 71 13    .......1..AQ.aq.
000001D0    22 32 81 08 14 42 91 A1 - B1 C1 09 23 33 52 F0 15    .2...B...Á..3Rð.
000001E0    62 72 D1 0A 16 24 34 E1 - 25 F1 17 18 19 1A 26 27    brÑ...4á.ñ......
000001F0    28 29 2A 35 36 37 38 39 - 3A 43 44 45 46 47 48 49    ...56789.CDEFGHI
00000200    4A 53 54 55 56 57 58 59 - 5A 63 64 65 66 67 68 69    JSTUVWXYZcdefghi
00000210    6A 73 74 75 76 77 78 79 - 7A 82 83 84 85 86 87 88    jstuvwxyz.ƒ.....
00000220    89 8A 92 93 94 95 96 97 - 98 99 9A A2 A3 A4 A5 A6    .Š.........š.....
00000230    A7 A8 A9 AA B2 B3 B4 B5 - B6 B7 B8 B9 BA C2 C3 C4    ...ª².µ...¹ºÂÃÄ
00000240    C5 C6 C7 C8 C9 CA D2 D3 - D4 D5 D6 D7 D8 D9 DA E2    ÅÆÇÈÉÊÒÓÔÕÖ.ØÙÚâ
00000250    E3 E4 E5 E6 E7 E8 E9 EA - F2 F3 F4 F5 F6 F7 F8 F9    ãäåæçèéêòóôõö÷øù
00000260    FA FF DA 00 0C 03 01 00 - 02 11 03 11 00 3F 00 FF    úÿÚ.............ÿ
00000270    00 3F FA 2B EA FF 00 D8 - 67 C2 9E 03 F1 FF 00 ED    ..ú.êÿ.ØgÂž.ñÿ.í
00000280    75 FB 3E 7C 3B F8 9D E0 - 4D 17 E2 3F 81 3E 23 7C    uû>|;ø.àM.â?.>#|
00000290    50 F0 97 80 7C 4D E1 8D - 77 56 F1 AE 87 6D 2E 99    Pð..|Má.wVñ®.m..
000002A0    E2 FD 5E D7 42 B8 D4 6D - 35 4F 87 FE 2C F0 67 88    âý^×B¸Ôm5O.þ,ðg.
```

Gambar 3. Hasil Konversi Hexadecimal

3.1 Analisis Algoritma RC6

Sebelum proses enkripsi, citra akan disesuaikan dengan mengubah panjang pixel pada citra menjadi 32 bit per pixel, atau mengalikan seluruh pixel pada gambar dengan 32 bit. Hal ini dimaksudkan untuk menyesuaikan aturan pada algoritma RC6, yaitu ukuran kata satuan (w) per bit adalah 32 bit. Kemudian pembagian block pada citra dilakukan dengan memecah seluruh jumlah bit pada citra dengan 128 bit, hasilnya setiap block tersebut akan dipecah lagi menjadi 4 buah block register A, B, C, dan D yang memiliki nilai masing-masing 32 bit. Gambar 4 menunjukkan ilustrasi perhitungan bit dan perhitungan block pada sampel citra yang akan digunakan pada proses analisa.



Gambar 4. Perhitungan jumlah bit dan block

Sampel citra yang digunakan adalah citra berukuran 512x512, artinya ada 512 baris dan 512 kolom pada citra, nilai yang terdapat pada 1 pixel mempresentasikan kandungan warna yang terdapat pada tiap satuan pixel yang dinyatakan dengan 1 warna sama dengan 1 byte atau 8 bit. Pada pengolahan citra, 32 bit sama artinya dengan 4 komponen warna dasar yaitu Cyan, Magenta, Yellow, dan Key (Black), akan tetapi untuk citra dengan komponen 24 bit (RGB) dan Grayscale, secara umum kekurangan bit pada setiap block akan diisi dengan bit pada block selanjutnya.

Pada RC6, ketetapan 32 bit per satuan pixel mempresentasikan ukuran setiap kata/pixel pada

ISSN : 1411-1624
e-ISSN: 2621-8089

citra, sehingga setiap pixel pada citra harus memiliki jumlah 32 bit. Karena algoritma RC6 merupakan algoritma block chipper, maka jumlah pixel pada citra yang telah dikonversi ke dalam bentuk bit harus dibagi dengan jumlah yang tetap yaitu 128 bit. Pada pembagian ini, setiap 128bit akan dipecah lagi menjadi 4 buah block register yang memiliki nilai 32bit yaitu block A, B, C dan D, jumlah setiap pixel harus tetap sama 32 bit per satu pixel, karena jika jumlahnya kurang, maka harus ditambahkan padding pada citra tersebut, penambahan padding tersebut akan mempengaruhi pada proses enkripsi citra, sehingga ketika ciphertext didekripsi, citra tidak dapat kembali menjadi plaintext.

Algoritma RC6 bekerja dengan 4 buah register block berukuran 32 bit, yaitu A, B, C dan D. setelah pembagian block selesai, maka proses selanjutnya akan masuk pada tahap enkripsi, pembangkitan kunci dan dekripsi. Karena ukuran dari semua sampel citra adalah sama, 512x512, maka jumlah bit dan pembagian blocknya sama seperti sampel pada ilustrasi.

1. Pembangkitan Kunci Algoritma RC6

Proses pembangkitan kunci pada algoritma RC6 dilakukan dengan memasukkan kunci yang panjangnya dinyatakan dengan *b*, dimana $0 \leq b \leq 255$ *byte*. Panjang kunci ini lalu akan ditempatkan pada array *c w-bit words* atau L[0], L[1], …, L[c-1]. Kemudian masing-masing karakter pada kunci ditempatkan pada array, secara urut: L[0], … L[c-1]. Masing-masing nilai karakter atau kata w-bit akan dibangkitkan pada proses penambahan kunci *round* 2r+4 dan ditempatkan pada array S[0],…,S[2r+3]. Hasil perhitungan kunci dapat dilihat pada tabel 1.

Tabel 1. Hasil Perhitungan kunci

Round	*Value*	*Round*	*Value*
0	A= S[0] = BF0A8B18, B = L[0] = F85671C8	22	A= S[22] = 94F2 CA88, B = L[0] = 66E7 B788
1	A= S[1] = 6BCE3FE0, B = L[0] = 2127A648	23	A= S[23] = 55BE 1890, B = L[0] = E530 99C8
2	A= S[2] = A3157E8, B = L[0] = 5ACA0A88	24	A= S[24] = 401B 6898, B = L[0] = 2A62 2C08
3	A= S[3] = BC1907F0, B= L[0] = B71A ACC8	25	A= S[25] = AE4E 48A0, B = L[0] = C585 BE48
4	A= S[4] = 1F9767F8, B= L[0] = B592 BF08	26	A= S[26] = EABD A8A8, B = L[0] = 821D 5088
5	A= S[5]= 2106 C800, B= L[0] = B4CE 5148	27	A= S[27] = A4B1 08B0, B = L[0] = 3674 E2C8
6	A= S[6] = 181A 2808, B= L[0] = 6745 E388	28	A= S[28] = 08C4 68B8, B = L[0] = F9CC 7508
7	A= S[7] = 562D 8810, B= L[0] = EB9D 75C8	29	A= S[29] = 35D7 C8C0, B = L[0] = 7D24 0748
8	A= S[8] = 5B40 E818, B= L[0] = 36F5 0808	30	A= S[30] = AAEB 28C8, B = L[0] = 407B 9988
9	A= S[9] = D054 4820, B=L[0] = 3A4C 9A48	31	A= S[31] = 5FFE 88D0, B = L[0] = 03DC CC40
10	A= S[10] = 8567 A828, B= L[0] = FDA4 2C88	32	A= S[32] = 155E EC98, B = L[0] = C9DF DFC8
11	A= S[11] = 3A7B 0830, B= L[0] = C0FB BEC8	33	A= S[33] = E236 74E0, B = L[0] = 60B4 BE48
12	A= S[12] = EF8E 6838, B= L[0] = 8453 5108	34	A= S[34] = 7155 78E8, B = L[0] = 9053 D288
13	A= S[13] = A4A1 C840, B= L[0] = 47AA E348	35	A= S[35] = D902 08F0, B = L[0] = 4AB0 F4C8
14	A= S[14] = 59B5 2848, B= L[0] = 0B02 7588	36	A= S[36] = DB0B 68F8, B = L[0] = 2DE5 0708
15	A= S[15] = 0EC8 8850, B= L[0] = CE5A 07C8	37	A= S[37] = F6B2 C900, B = L[0] = 24C0 9948
16	A= S[16] = C3DB E858, B= L[0] = 91B1 9A08	38	A= S[38] = 7C86 2908, B = L[0] = 0A38 2B88
17	A= S[17] = 78EF 4860, B= L[0] = 5509 2C48	39	A= S[39] = C899 8910, B = L[0] = 968F BDC8
18	A= S[18] = 2E02 A868, B= L[0] = 1860 BE88	40	A= S[40] =7DAC E918, B = L[0] = A1E7 5008
19	A= S[19] = E316 0870, B= L[0] = DBB8 50C8	41	A= S[41] = 72C0 4920, B = L[0] = A53E E248
20	A= S[20] =89E5 3640, B= L[0] = 2CEE 5148	42	A= S[42] = 27D3 A928, B = L[0] = 6896 7488
21	A= S[21] = 4A0E A8C0, B= L[0] = B7E9 E948	43	A= S[43] = C804 91C0, B = L[0] = 84DA 4B48

2. Enkripsi Algoritma RC6

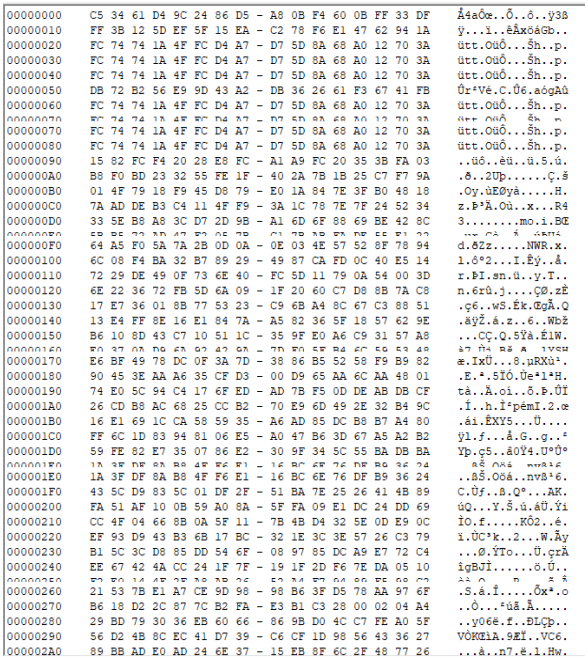
Proses enkripsi pada RC6, memecah blok 128 bit menjadi 4 buah block register 32 bit, dalam proses enkripsi citra, jumlah pixel yang telah diubah menjadi bit, dibagi 128 bit agar kemudian bisa dipecah lagi menjadi 4 buah block register 32 bit. Pada kegiatan ini, jumlah keseluruhan bit pada citra berukuran 512x512 pixel adalah 8.388.608, setelah dibagi 128bit hasilnya adalah 65.536 block, artinya pada 1 block dari 65.536 block tersebut, terdapat 4

ISSN : 1411-1624
e-ISSN: 2621-8089

buah block register yang jumlahnya masing-masing adalah 32 bit. Block register ini dinamakan A,B,C dan D. Pada byte pertama pada plaintext akan ditempatkan pada blok A, sedangkan byte terakhir akan ditempatkan pada block D. Hasil dari proses enkripsi pada RC6 akan didapatkan (A, B, C, D) = (B, C, D, A), yang artinya nilai dari sisi kanan berasal dari nilai pada register di sisi kiri.

Proses enkripsi dimulai dari proses pembangkitan 44 buah sub kunci yang dinamakan S[0] sampai S[43], yang masing-masing panjangnya 32 bit. Pada proses enkripsi algoritma RC6, terdapat 20 round atau 20 iterasi yang masing-masing terdiri atas 2 buah sub kunci (2r+4). Proses pertama dan terakhir pada enkripsi RC6 adalah proses whitening, yaitu proses menyamarkan nilai pada iterasi pertama yaitu menjumlahkan nilai register block B dengan S[0] dan nilai register block D dengan S[1], dan menyamarkan nilai pada iterasi terakhir yaitu menjumlahkan nilai register block A dengan S[42] dan nilai register block C dengan S[43], proses ini bertujuan untuk mempersulit pengenalan pola dan nilai asli pada plaintext, sehingga ciphertext sulit untuk dikenali.

Selain iterasi pada proses whitening, aturan yang digunakan pada proses enkripsi setip iterasinya adalah dengan memasukan nilai register blok B ke dalam fungsi f(x) = x(2x+1) kemudia diputar ke kiri sejumlah log2 w atau 5 bit, hal ini dilakukan pula pada nilai register blok D, hasil dari proses tersebut didefinisikan sebagai t untuk fungsi pada block B dan u pada fungsi block D. Nilai t kemudian di XOR kan dengan nilai pada block register A, kemudian hasil XOR diputar ke kiri sejauh t, lalu sub kunci S[2i] dijumlahkan dengan A. sedangkan untuk nilai C, di XOR kan dengan nilai u, dan diputar ke kiri sejauh u, lalu dijumlahkan dengan sub kunci S[2i+1]. Keempat block register lalu dipertemukan dengan aturan bahwa nilai A menempati D, nilai B ditempatkan di A, nilai C ditempatkan di B dan nilai D ditempatkan di C, iterasi selain proses whitening dilakukan sebanyak 20 kali atau r = 20. Proses enkripsi dikerjakan berdasarkan block dan block register dapat dilihat pada gambar 5.



Gambar 5. Potongan hasil enkripsi

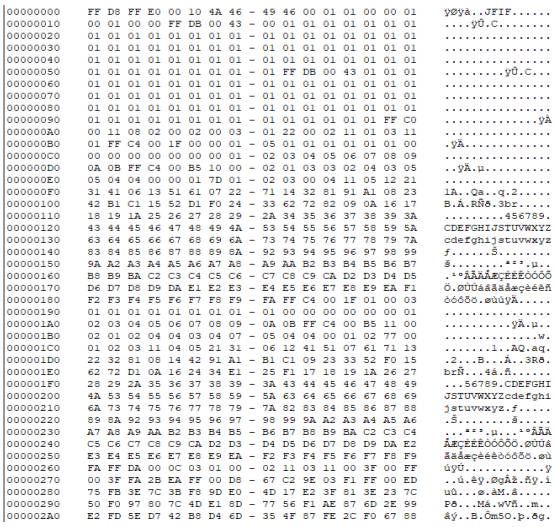
Setelah enkripsi, citra tidak lagi dapat dibuka seperti citra asli sebelumnya, karena proses enkripsi juga ikut mengubah kode file pada citra .jpeg menjadi tidak terdefinisi, lihat gambar 6.

Gambar 6. File citra yang telah di enkripsi

ISSN : 1411-1624
e-ISSN: 2621-8089

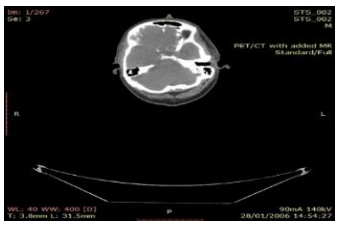
3. Dekripsi Algoritma RC6

Pada proses dekripsi, ciphertext dikembalikan menjadi *plaintext* dengan cara yang sama dengan proses enkripsi, hanya saja proses yang dilalui merupakan kebalikan dari proses enkripsi. Seperti pada proses whitening terakhir pada enkripsi, jadi dilakukan pada proses pertama dengan sub kunci pada iterasi terakhir dan jika pada proses enkripsi dijumlahkan, maka pada proses dekripsi menjadi dikurangi, begitu pun prosesnya pada proses whitening pertama, pada dekripsi jadi dilakukan pada proses iterasi terakhir. Setelah dikonversi, citra akan melalui tahap dekripsi, lihat gambar 7.



Gambar 7. Hasil proses dekripsi dalam bentuk hexadecimal

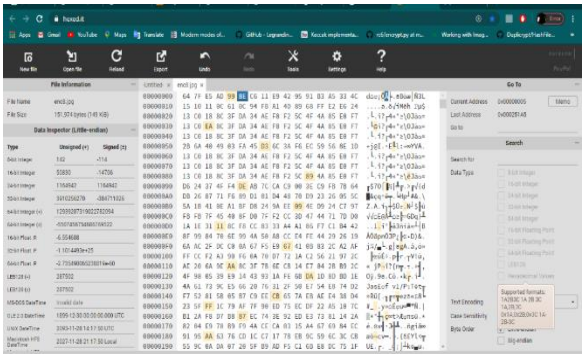
Pada proses dekripsi, citra mulanya tidak dapat dibuka karena masih berbentuk file enkripsi, kemudian file tersebut di konversi ke dalam hexadecimal, melalui proses dekripsi dan kembali menjadi citra yang asli, lihat gambar 8.



Gambar 8. Citra hasil Dekripsi

4. Pengujian Algoritma RC6

Pada pengujian ini, file enkripsi dimodifikasi menggunakan *tools* pada *website https://hexed.it/*. Pada website ini, nilai hexadecimal pada file enkripsi dapat dibaca dan dimodifikasi, sehingga memungkinkan seorang peretas memodifikasi file enkripsi, sehingga ketika pesan berisi file enkripsi dikirim, penerima akan menerima file enkripsi yang telah dimodifikasi. Berikut tampilan file yang dimodifikasi pada website https://hexed.it/. Gambar 9 menunjukkan tanda kuning pada file enkripsi sampel citra kedelapan merupakan nilai hexadecimal yang telah dimodifikasi. Kemudian file hasil modifikasi tersebut disimpan untuk selanjutnya melalui tahap dekripsi. berikut perbedaan file enkripsi yang asli dengan file modifikasi.



Gambar 9. Modifikasi File Enkripsi

Setelah proses dekripsi selesai, citra hasil dekripsi dibandingkan untuk melihat apakah proses modifikasi pada file enkripsi berpengaruh pada penampakan citra tersebut.

Gambar 10 Citra Hasil Dekripsi File Enkripsi Asli

ISSN : 1411-1624
e-ISSN: 2621-8089

Gambar 11. Citra Hasil Dekripsi File Enkripsi Modifikasi

Berdasarkan pengujian pada sampel citra 1, modifikasi nilai hexadecimal pada file enkripsi, tidak dapat dikembalikan menjadi file citra seperti sebelum proses enkripsi, sehingga informasi pada citra asli masih tetap terjaga meskipun file enkripsi telah dimodifikasi, terlihat pada gambar 10 dan 11.

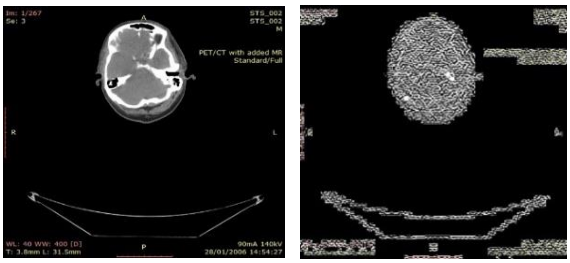
3.2 *Algoritma Keccak*

Keccak merupakan algoritma cipher block satu arah yang bekerja berdasarkan kontruksi spon, pada algoritma ini, meskipun bersifat cipher block, pada algoritma ini tidak ada key-schedule dan memiliki konstanta round yang bersifat tetap. Pada algoritma ini, sama seperti algoritma RC6 proses dilakukan terhadap *block-block* data yang saling berhubungan. Keccak dapat dinyatakan dengan fungsi permutasi Keccak-f[b] atau Keccak-f[r+c] dengan Panjang permutasi b. ukuran setiap lane di mana $b=25 \times 2^{l}$ dengan $0 \leq l \leq 6$.

Hash menggunakan algoritma keccak bertujuaan untuk menjaga keaslian citra, atau integrity. Menjaga integrity pada citra sangat penting agar citra tersebut tidak dapat dimodifikasi sehinga merusak informasi pada citra tersebut, terutama jika citra tersebut merupakan citra penting seperti citra digital rekam medis. Nilai hash merupakan nilai yang sensitif, sedikit saja citra dimodifikasi, maka nilai hash-nya akan berubah.

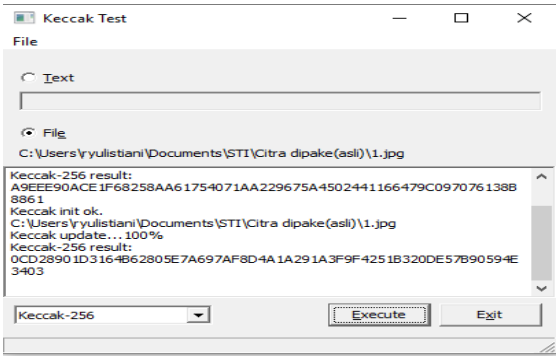
Pada pengujian ini, dilakukan perbandingan nilai keluaran hash pada citra asli dan citra modifikasi, untuk melihat apakah perubahan pada citra akan mempengaruhi nilai hash yang dihasilkan dari penerapan algoritma keccak. Jika tidak ditemukannya perubahan atau perubahannya sangat sedikit maka algoritma keccak kurang aman dalam mengamankan integrity pada citra. Proses modifikasi pada pengujian ini menggunakan tools yang sama dengan tools pada pengujian sebelumnya, yaitu *website https://hexed.it/*.

Untuk melihat perbedaan antara citra asli dan citra modifikasi, dapat dilihat pada gambar 12.



Gambar 12. Citra asli versus citra modifikasi

Langkah selanjutnya, hitung nilai *hash* kedua citra dan bandingkan, lihat gambar 13.



Gambar 13. Perbandingan nilai Hash

Nilai Hash Keccak-256: Citra Asli (1.jpg) = 0CD28901D3164B62805E7A697AF8D4A1A291A3F9F4251B320DE57B90594E3403

Nilai Hash Keccak-256:Citra Modifikasi (1(1).jpg)= A9EEE90ACE1F68258AA61754071AA229675A4502441166479C097076138B8861

ISSN : 1411-1624
e-ISSN: 2621-8089

Berdasarkan perhitungan nilai hash pada kedua citra. Perbandingan nilai hash antara citra asli dan citra modifikasi, tidak sama, itu menandakan bahwa integrity pada citra tetap terjaga keasliannya. Sehingga jika citra hasil enkripsi menggunakan RC6 dimodifikasi oleh orang lain, algoritma keccak dapat menunjukkan integrity citra tetap terjaga keasliannya.

3.3. Uji Perbandingan Ukuran pada Citra

Perubahan ukuran pada proses enkripsi merupakan hasil yang wajar, akan tetapi ketika dilakukan dekripsi terhadap citra, perubahan ukuran akan menjadi masalah ketika ukuran hasil dekripsi tidak sama dengan ukuran pada citra asli yang belum melalui proses enkripsi. Perubahan ukuran pada citra dekripsi dapat menandakan bahwa proses dekripsi tidak berjalan dengan sempurna dan membuat kualitas citra berubah, sehingga ditakutkannya mempengaruhi informasi pada citra juga ikut berubah.

Berdasarkan pada tabel 2, dari 26 percobaan terhadap citra yang berbeda, dengan menggunakan satuan ukuran bytes, tidak ditemukannya perbedaan ukuran citra asli dan citra yang telah melalui proses enkripsi maupun dekripsi. Artinya, citra yang dienkripsi dapat dikembalikan lagi ke citra semula tanpa merusak kualitas ukuran citra, hal ini menandakan bahwa aplikasi sudah sangat baik dalam melakukan proses enkripsi dan dekripsi dengan algoritma RC6. Hal tersebut menunjukkan, bahwa algoritma RC6 mampu mengamankan citra digital rekam medis tanpa mempengaruhi ukuran citra.

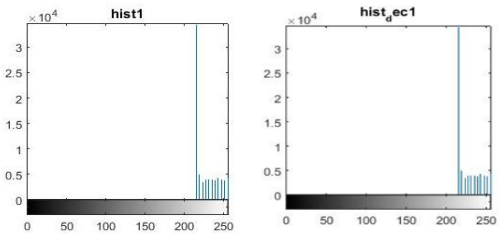
Tabel 2. Hasil uji ukuran citra

No.	*Nama*	*Kunci*	*Ukuran Citra Asli (bytes)*	*Ukuran Citra Enkripsi (bytes)*	*Ukuran Citra Dekripsi (bytes)*	*Tingkat Perubahan*
1	Citra 1	1234	79.183	79.183	79.183	0%
2	Citra 2	1234	76.600	76.600	76.600	0%
3	Citra 3	1234	80.623	80.623	80.623	0%
4	Citra 4	1234	114.508	114.508	114.508	0%
5	Citra 5	1234	142.834	142.834	142.834	0%
6	Citra 6	1234	151.863	151.863	151.863	0%
7	Citra 7	1234	157.984	157.984	157.984	0%
8	Citra 8	1234	151.974	151.974	151.974	0%
9	Citra 9	1234	157.637	157.637	157.637	0%
10	Citra 10	1234	155.204	155.204	155.204	0%
11	Citra 11	1234	175.939	175.939	175.939	0%
12	Citra 12	1234	175.598	175.598	175.598	0%
13	Citra 13	1234	175.913	175.913	175.913	0%
14	Citra 14	1234	176.937	176.937	176.937	0%
15	Citra 15	1234	180.046	180.046	180.046	0%
16	Citra 16	1234	178.044	178.044	178.044	0%
17	Citra 17	1234	173.587	173.587	173.587	0%
18	Citra 18	1234	163.883	163.883	163.883	0%
19	Citra 19	1234	147.448	147.448	147.448	0%
20	Citra 20	1234	156.013	156.013	156.013	0%
21	Citra 21	1234	162.337	162.337	162.337	0%
22	Citra 22	1234	163.668	163.668	163.668	0%
23	Citra 23	1234	155.888	155.888	155.888	0%
24	Citra 24	1234	141.103	141.103	141.103	0%
25	Citra 25	1234	133.118	133.118	133.118	0%
26	Citra 26	1234	118.587	118.587	118.587	0%

3.4 Analisis perbandingan histogram

Untuk meninjau tingkat keamanan tersebut, dilakukan proses pengujian histogram pada citra asli dan citra hasil dekripsi. Hal ini dilakukan untuk melihat bagaimana kualitas citra secara mendetail, agar perubahan sedikit saja pada citra hasil dekripsi dapat diamati, lihat gambar 14.

ISSN : 1411-1624
e-ISSN: 2621-8089



Gambar 14 perbandingan kualitas citra dari *histogram* citra asli dan citra hasil dekripsi.

Histogram pada citra asli dan citra yang telah didekripsi menunjukkan ukuran yang sama dan tidak ada perubahan. Skala yang diambil pada pengukuran ini adalah nilai intensitas pixel dari 0-255. Hasil perbandingan yang menunjukkan tidak adanya perubahan antara histogram asli dan hasil dekripsi membuktikan bahwa tidak adanya perubahan pada kualitas citra berdasarkan intensitas nilai pixel pada histogram.

4. KESIMPULAN

Berdasarkan hasil implementasi hashing *algoritma Keccak dan Rivest Code 6* pada pengamanan citra digital rekam medis maka dapat diambil suatu kesimpulan :

1. *Algoritma RC6* mampu mengamankan citra digital rekam medis, serta mengembalikan citra hasil enkripsi dengan baik tanpa merusak keaslian citra. Berdasarkan pengujian yang dilakukan dengan memodifikasi file enkripsi, file tersebut tidak dapat didekripsi kembali, sehingga kerahasiaan citra terjaga dengan baik, serta mudah dikenali jika terjadi modifikasi.
2. *Algoritma Keccak* dapat dipercaya dalam menjaga integrity dari citra. Hal ini dapat dilihat dari pengujian menggunakan hash decryptor tools¸dimana nilai hash dari 26 citra tidak dapat di-*generate* menggunakan 2 tools yang digunakan, hal tersebut menandakan bahwa keaslian citra dapat dipercaya.
3. Pengujian terhadap perbandingan ukuran citra asli, citra enkripsi dan citra dekripsi, meunjukan bahwa tidak adanya perubahan terhadap 26 citra yang diuji.
4. Pengujian *histogram* terhadap citra asli dan citra hasil dekripsi dari 26 citra sampel menunjukan tidak adanya perubahan, artinya kualitas citra pada proses enkripsi dan dekripsi menggunkan RC6 tetap terjaga dengan baik sehingga tidak mengubah informasi yang terdapat dalam citra digital rekam medis.

Untuk pengembangan yang lebih baik lagi bagi penelitian selanjutnya sebaiknya menerapkan algoritma dan metode *authentication* sebagai pelengkap dari ketiga dasar pengamanan kriptografi. Serta membalut RC6 dengan algoritma pertukaran kunci asimetris sehingga kunci bersifat public, untuk lebih memudahkan user.

DAFTAR PUSTAKA

[1] Wismayana, I. Putu Ary, Elysanti Dwi Martadiani, and Lisna Astuti. "18 F-FLOURODEOXYGLUCOSE (18 FDG) POSITRON-EMISSION TOMOGRAPHY (PET) SEBAGAI MODALITAS IMAGING PENATALAKSANAAN KANKER TIROID."

[2] Al-Haj, Ali, Noor Hussein, and Gheith Abandah. "Combining cryptography and digital watermarking for secured transmission of medical images." 2016. *2nd International Conference on Information Management (ICIM). IEEE*, 2016.

[3] Susanto, Edy., dkk, “Manajemen Informasi Kesehatan IV: Etika Profesi dan Hukum Kesehatan." (2017).

[4] Tan, Chun Kiat, et al. "Security protection

ISSN : 1411-1624
e-ISSN: 2621-8089

of DICOM medical images using dual-layer reversible watermarking with tamper detection capability." *Journal of Digital Imaging* 24.3 (2011): 528-540.

[5] Prayudi, Yudi, and Idham Halik. "Studi dan Analisis Algoritma RIVEST CODE 6 (RC6) Dalam Enkripsi/Dekripsi Data." *Seminar Nasional Aplikasi Teknologi Informasi (SNATI*). 2005.

[6] Abdurohman, Maman. "Analisis Performansi Algoritma Kriptografi RC6." *Journal Departemen Teknik Elektro ITB* (2002).

[7] Toyib, Rozali, and Ardi Wijaya. "ANALISIS PERBANDINGAN ALGORITMA SIMETRIS RIVEST CODE 5 DENGAN ALGORITMA SIMETRIS RIVEST CODE 6) (Studi Kasus: SMK Negeri Seluma)." *Jurnal Informatika Upgris* 4.2 (2018).

[8] Paulus, Erick, and Mochamad Azmi Fauzan. *"A Framework to Ensure Data Integrity and Safety."* (2018).

[9] Nazal, Muhammad Asghar, Reza Pulungan, and Mardhani Riasetiawan. "Data Integrity and Security using Keccak and Digital Signature Algorithm (DSA)." *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)* 13.3: 273-282.

[10] Bertoni, Guido, et al. "Keccak specifications." *Submission to nist (round 2)* (2009): 320-337.

[11] Clark, Kenneth, et al. "The Cancer Imaging Archive (TCIA): maintaining and operating a public information repository." *Journal of digital imaging 26.6* (2013): 1045-1057.