JPEG/EXIF MESSAGE DIGEST COMPILATION WITH SHA512 HASH FUNCTION

Rachmad Fitriyanto*[1], Anton Yudhana[2], Sunardi[3]
[1]Magister of Informatics Engineering, Universitas Ahmad Dahlan, Yogyakarta
[2,3] Electrical Engineering Department, FTI, Universitas Ahmad Dahlan, Yogyakarta
e-mail: *[1]fitriyanto7477@gmail.com, [2]yudhana@ee.uad.ac.id, [3]sunardi@mti.uad.ac.id

Abstract

Security information method for jpeg / exif documents generally aims to prevent security attack by protecting documents with password and watermark. Both methods cannot used to determine the condition of data integrity at the detection stage of the information security cycle. Message Digest is the essence of a file that used to represent data integrity. This study aims to compile a message digest to detect changes that occur in jpeg / exif documents in information security. The research phase consists of five stages. The first stage, identification of the jpeg / exif document structure conducted using Boyer-Moore string matching algorithm to find jpeg/exif segments location. The Second stage is segment content acquisition, conducted based on segment location and length obtained. The Third step, computing message digest for each segment using SHA512 hash function. Fourth stage, jpeg / exif document modification experiments to identified affected segments. Fifth stage is selecting and combining the hash value of the segment into message digest. Obtained result show message digest for jpeg / exif documents composed of two parts, the hash value of the SOI segment and the APP1 segment. The SOI segment value used to detect modifications for jpeg to png conversion and image editing. The APP1 hash value used to detect metadata editing. The SOF0 hash values use to detect modification for image recoloring, cropping and resizing.

Keywords—Boyer-Moore, Hash Value, Jpeg/exif , Message Digest, SHA512

1. INTRODUCTION

The jpeg/exif document is a document format that results from the use of digital cameras such as smartphone camera. The jpeg/ exif documents as image files are widely used in digital communications such as on social media. The Exchange of information requires security to ensure the information received is the same as the information sent. Information security for jpeg/exif documents generally designed to prevent document modifications. The use of passwords in the jpeg/exif document has long been used but can still be overcome with a variety of password remover tools that are widely available. Other forms of security are shown in the study by Wijayanto [1]. This study shows exif metadata data from jpeg/exif documents can used to prevent copyright theft. The usage of watermark as information security method in study by Sukarno [2] provide protection for preventing document for modification. The use of passwords exif metadata and watermark cannot used in detection stage of information security cycle to detect changes that occur in received jpeg/exif documents.

Message Digest is the essence of a file that can used to represent data integrity. The Message digest is widely used to detect the integrity of data in installers provided by open-source application developers. The Message digest compiled using the hash function. The Hash function is a cryptographic method for the one-way encryption process. The output hash function is a hash value that has characteristics that cannot translated or decrypt into the original form. The Hash value is very sensitive to changes in the input of any size. Secure Hash Algorithm (SHA) is a hash function developed by the National Institute of Science and Technology (NIST). SHA consists of

several categories that distinguished based on the size of the output hash value [3]. SHA512 is a hash function variant of the SHA-2 group [4]. SHA512 has an output size of 512 bits that make this variant better than previous hash function.

The use of SHA512 for information security was found in a study by Refialy [5]. This study uses SHA512 to compile the hash value of a pdf document. The results obtained indicate the resulting hash value is able to detect small changes in the modification of pdf documents. The jpeg / exif file has a larger size than a pdf document. The size of the jpeg / exif document is the result of the development of optical technology in digital cameras. The larger the size of the document, the process of composing the message digest requires more time. This study aims to compile a concise message digest to detect changes in jpeg / exif documents in information security.

String matching algorithms are used to match or compare one or several characters and strings [6]. The Boyer-Moore string matching algorithm is included in the exact string matching algorithm that performs searches by comparing the characters of the strings tested with the pattern sought [7]. The Boyer-Moore have two rules for searching process, Good-Character rule and Bad-Character Rule. These two rules determined the direction of searching process. Bad-Character rule occurred if character from pattern is not same as in string. This condition will make comparison shift to left to next character of pattern. Good-Character rule occurred if character from pattern is same as in string after Bad-Character rule occured. This condition will make the next comparison shift right aligning two match characters. Those two rules searching make searching process faster than other exact string matching algorithm by avoiding not-necessary character comparison [8] .

2. RESEARCH METHODOLOGY

Research study conducted in five stages that shown as figure 1. First stage is image file segment identification. This stage have purpose to identify the jpeg/exif file structure. Image files as research object acquired from two smartphone types, Asus Z00UD and Samsung Galaxy A5. Each smartphone take 10 image. Images file taken with embedded camera apps from each smartphone with mode auto from indoor and outdoor sites.

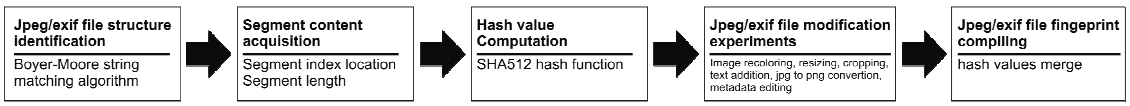

Figure 1. Research stages

Result from first stage is location index of each file parts. This location index will use on second stage as parameter to identified the beginning and end of file parts. Identification process conducted by use Boyer-Moore string matching algorithm for segment marker searching. Segment markers are data bit that located in the beginning of each jpeg/exif segments [9]. Tabel 1 shown segment marker value for each segments.

Table 1. Jpeg/exif segment marker

Segment	Segment Marker
SOI (Start Of Image)	ffd8
APP1 (Application-1)	ffe1
DQT (Define Quantization Table)	ffdb
SOF0 (Start Of Frame-0)	ffc0
DHT (Define Huffman Table)	ffc4
SOS (Start Of Scan)	ffda

Boyer-Moore string matching algorithm start pattern searching from most-right pattern character and shift to left until reach the left most character from pattern [10]. Figure 2 shown Boyer-Moore flowchart.

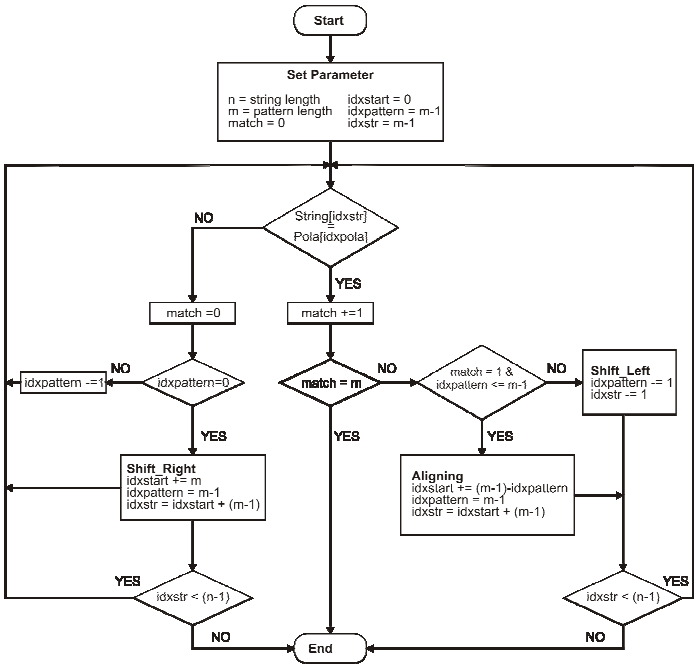

Figure 2. Boyer-Moore string matching flowchart

Boyer-Moore string matching algorithm have two searching stages. First stage is preprocessing, comprises of variabel assignment such as m for pattern length, n for string length idxstr for string character index, idxpatter for pattern character index and match for character match comparison that occured. Second stage is character comparison. The second stage executed as iteration loop that boundary by two condition. Looping will stop if match variabel have value same as m variable or comparison has reach the end of string. The segment location index identified by the last value of idxstr variable.

Second stage is segment content acquisition. To acquire segment content, need two parameters, starting index and length of content. The starting index provided by segmen index location. Segment length compute from substraction between index location values from two adjacent segments. The third stage is hash value computation. The computation conducted for every segment. Hash value computation with SHA512 hash function consist of three stages, preprocessing, hash computation and hash value compilation. Figure 3 shown the stages sequences.

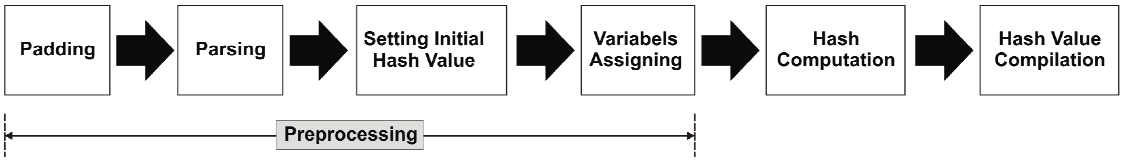

Figure 3. SHA512 processes

The preprocessing stages consist of four process, padding, parsing, setting initial hash value and variable assigning [3]. The padding process is the adjustment of the size of input data so that the processed data has a size of multiples of 512 bits. The parsing process is to divide the data bits into groups of data with 64-bit size. The process of determining initial hash values and register assignments arranged as shown in Table 2.

Table 2. Initial hash value inside eight registers

Register	Hash Value	
a	$H_{(0)}^0$	6a09e667f3bcc908
b	$H_{(1)}^0$	bb67ae8584caa73b
c	$H_{(2)}^0$	3c6ef372fe94f82b
d	$H_{(3)}^0$	a54ff53a5f1d36f1
e	$H_{(4)}^0$	510e527fade682d1
f	$H_{(5)}^0$	9b05688c2b3e6c1f
g	$H_{(6)}^0$	1f83d9abfb41bd6b
h	$H_{(7)}^0$	5be0cd19137e2179

The eight registers (a, b, c, d, e, f, g, h) and blocks of input data are used in computing the hash value according to the sequence shown in Figure 4.

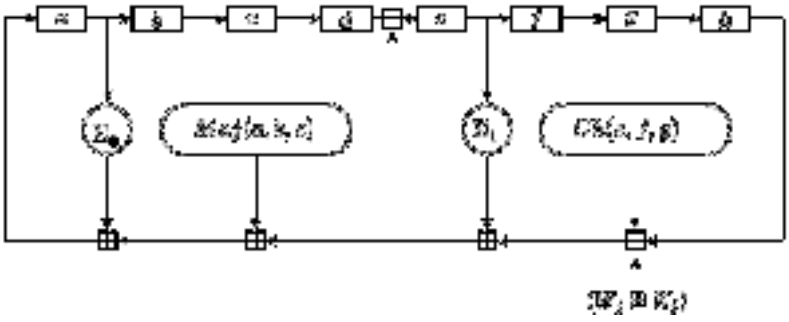

Figure 4. Hash computation

The hash value computing executed iteratively as much as the block of data bits generated from the parsing stage. The mathematical equation in Figure 4 used according to what is shown in equations 1 to 6.

$$Ch(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z) \quad (1)$$

$$Maj(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z) \quad (2)$$

$$\sum 0(x) = S^{28}(x) \oplus S^{34}(x) \oplus S^{39}(x) \quad (3)$$

$$\sum 1(x) = S^{14}(x) \oplus S^{18}(x) \oplus S^{41}(x) \quad (4)$$

$$\sigma_0(x) = S^{1}(x) \oplus S^{8}(x) \oplus R^{7}(x) \quad (5)$$

$$\sigma_1(x) = S^{19}(x) \oplus S^{61}(x) \oplus R^{6}(x) \quad (6)$$

The results of the computing process are eight hash values stored in eight registers. Figure 5 shows the results of calculating the hash value for the input "abc" string.

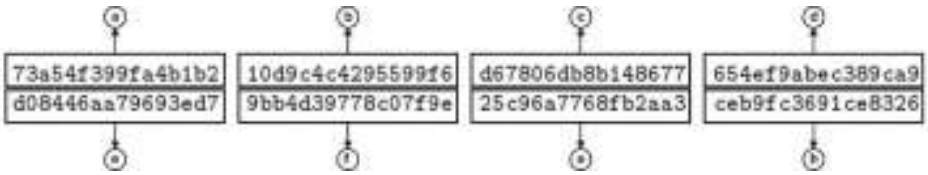

Figure 5. Hash value inside eight registers

Each hash value in eight registers is then summed with the initial hash value as shown in Figure 6.

```
H1 = 6a09e667f3bcc908 + 73a54f399fa4b1b2 = ddaf35a193617aba
H2 = bb67ae8584caa73b + 10d9c4c4295599f6 = cc417349ae204131
H3 = 3c6ef372fe94f82b + d67806db8b148677 = 12e6fa4e89a97ea2
H4 = a54ff53a5f1d36f1 + 654ef9abec389ca9 = 0a9eeee64b55d39a
H5 = 510e527fade682d1 + d08446aa79693ed7 = 2192992a274fc1a8
H6 = 9b05688c2b3e6c1f + 9bb4d39778c07f9e = 36ba3c23a3feebbd
H7 = 1f83d9abfb41bd6b + 25c96a7768fb2aa3 = 454d4423643ce80e
H8 = 5be0cd19137e2179 + ceb9fc3691ce8326 = 2a9ac94fa54ca49f
```

Figure 6. Summation intial hash value with registers hash value

The final result of the hash value of SHA512 is a combination of eight hash values from Figure 6 composed sequentially as shown in Figure 7.

```
ddaf35a193617aba cc417349ae204131 12e6fa4e89a97ea2 0a9eeee64b55d39a
2192992a274fc1a8 36ba3c23a3feebbd 454d4423643ce80e 2a9ac94fa54ca49f
```

Figure 7. SHA512 hash value from string “abc”

Research’s third stage is modification experiments that consist six types file modification. Image recoloring, resizing, cropping conducted with ACDSee Pro.8 application. Metadata manipulation conducted with Hex Editor Neo. Image file format conversion from jpeg to png conducted by using FormatFactory application and image text addition conducted by using Paint application. This third stage have purpose to identified altered segments caused by image modification. Identification conducted by comparing each segments hash value from original image with hash values from modified image as shown in figure 8.

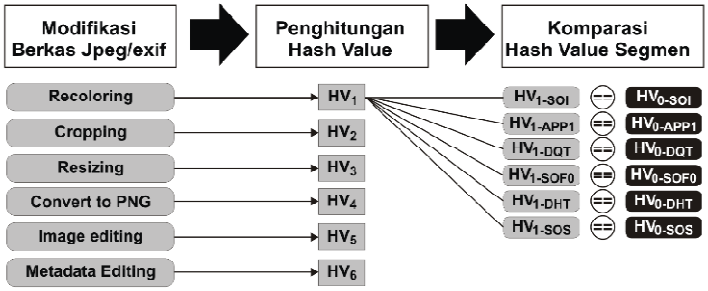

Figure 8. Hash value comparison between original and modified image

Before the image file is modified, the hash value is calculated from the original file (HV0). The hash value of the original file is used in comparison to the hash value of the modified file (HV1, HV2, HV3, HV4, HV5, HV5, HV6). Each modified form will have the value hash of each segment compiled. The comparison of hash values is done for each of the same segments. The hash value that has changed in each form of modification will be used as the compiler of the fingerprint file at a later stage. The last stage is message digest compiling. Each hash value from third stage compile into one string file to form one message digest.

3. RESULTS AND DISCUSSION

The result of jpeg / exif file segment identification shown in Table 3 and 4. Table 3 shown segment location index for image file from Asus Z00UD smartphone.

Table 3. Segment index location for image file from smartphone Asus Z00UD

No.	JPEG/Exif file	Segment Index					
		SOI	APP1	DQT	SOF0	DHT	SOS
1	P_20180723_141211	0	4	26400	26844	26726	27630
2	P_20180731_134049	0	4	26368	61656	26694	27598
3	P_20180823_124724	0	4	26378	26664	26704	27610
4	P_20180905_085850	0	4	25350	25636	25676	26580
5	P_20190110_100735	0	4	26318	26602	26642	27548
6	P_20190324_100013	0	4	34212	25544	25584	26490
7	P_20190324_100040	0	4	25378	25662	25704	26608
8	P_20190324_114023	0	4	25278	25769	25809	26713
9	P_20190324_115302	0	4	25348	25789	25792	26713
10	P_20190324_121005	0	4	25405	25663	25564	26580

The location index of the SOI and APP1 segment in all jpeg / exif files in Table 3 has the same values 0 and 4. This is because the SOI segment is located in the initial bit of the image file and consists of only four bits containing the segment marker segment, ffd8. The location index and length of the segments DQT, SOF0, DHT and SOS in ten jpeg / exif files have different values. This makes the index location of a segment of a jpeg / exif file not be used to identify the location of the segment in another file. Therefore, the Boyer-Moore algorithm matching string is always used to identify the location of the segment for each time the message digest is compiled. The identification of the location of the jpeg / exif file segment from the Samsung Galaxy A5 smartphone is shown in Table 4.

Table 4. Segment index location for image file from smartphone Samsung Galaxy A5

No.	JPEG/Exif file	Segment Index					
		SOI	APP1	DQT	SOF0	DHT	SOS
1	01_20180825_131753	0	4	2000	2286	2326	3218
2	02_20171225_111236	0	4	2000	2286	2326	3218
3	03_20171201_124906	0	4	2000	2286	2326	3218
4	04_20171201_130420	0	4	2000	2286	2326	3218
5	05_20180825_134942	0	4	2000	2286	2326	3218
6	06_20181213_172235	0	4	2000	2286	2326	3218
7	07_20190114_154205	0	4	2000	2286	2326	3218
8	08_20190114_154209	0	4	2000	2286	2326	3218
9	09_20190114_154215	0	4	2000	2286	2326	3218
10	10_20190114_154220	0	4	2000	2286	2326	3218

The location index of the SOI and APP1 segment in all jpeg / exif files in Table 4 has the same values 0 and 4. This is because the SOI segment is located in the initial bit of the image file and segment location index on the jpeg/exif file from the Samsung Galaxy A5 smartphone shows that each file has the same segment location. Therefore, identification of the location of the jpeg/exif segment that will conducted in the future does not require a search from the start.
The location index of Tables 3 and 4 used to calculate the length of each segment. This calculation compilshed by finding the difference in location index values from two adjacent segments as formulated in equation 7.

$$P_{segmen(n)} = Index(n) - Index(m) \qquad (7)$$

An example of the calculation of segment length for jpeg / exif documents from the Asus Z00UD smartphone shown in Table 5.

Table 5. Image file Segment length

No.	File name	Smartphone type	Segment Length (bit)					
			SOI	APP1	DQT	SOF0	DHT	SOS
1	P_20180723_141211	Asus Z00UD	4	26396	444	118	904	9181254
2	P_20180823_124724	Asus Z00UD	4	26374	286	40	906	5386020
3	P_20180905_085850	Asus Z00UD	4	25346	286	40	904	5319634
4	P_20190324_100013	Asus Z00UD	4	34208	8668	40	906	3197654
5	P_20190324_100202	Asus Z00UD	4	63584	38180	40	906	5009834
6	01_20180825_131753	Samsung G.A5	4	1996	286	40	892	10805918
7	02_20171225_111236	Samsung G.A5	4	1996	286	40	892	7275684
8	03_20171201_124906	Samsung G.A5	4	1996	286	40	892	6686880
9	04_20171201_130420	Samsung G.A5	4	1996	286	40	892	6598378
10	05_20180825_134942	Samsung G.A5	4	1996	286	40	892	4788372

The APP1 segment length in the first file in Table 3 has a value of 26396 obtained using equation 7, which is the result of a reduction between the DQT segment location index and the APP segment1. The results of the long calculation of all segments show that the SOS segment has the largest segment length. This is because the SOS segment contains image data that is the main data from the Image file. The greater the size of the jpeg / exif document, the longer the SOS segment will be. SOS segment length calculation is done by operating a reduction between the overall image bit length and the SOS segment location index. The bit length of image data obtained from the length calculation of the converted file at the beginning of the identification phase of the segment location that is also stored in the n variable in the search process with Boyer-Moore string matching algorithm.

The location index value and segment length then used as parameters for content acquisition for each segment. Figure 9 shows the application interface for the acquisition and calculation of hash values for each segment.

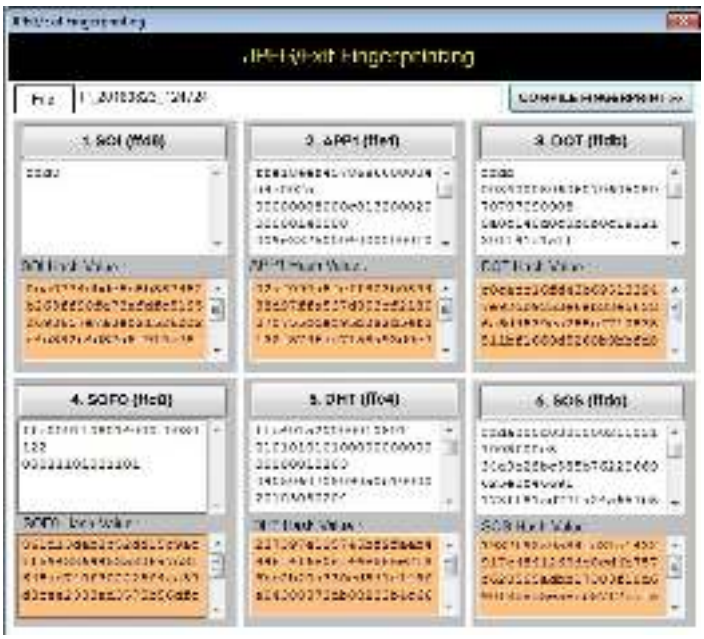

Figure 9. Application interface for segment content acquisition and hashing

Six hash values from six jpeg/segments form figure 9 have possibility to use as message digest elements that will conduct on research fourth stage. Elements selection process purposes is

to determined segments that affected if image file altered by comparing each hash values from original and modified image. Figure 10 shown hash values comparison from recoloring image.

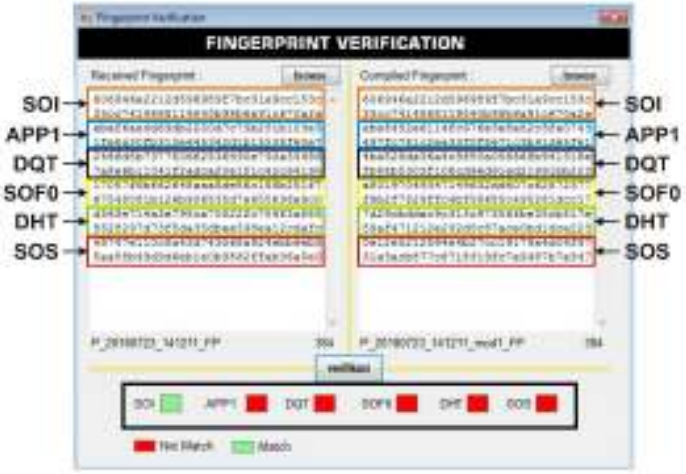

Figure 10. Application interface for hash values comparison

The left part of Figure 11 shows the hash value of six segments in the original file. The right section shows the hash value of five segments in the recoloring modification file. The comparison results per segment consist of two conditions, "Match" if the hash values of the two segments are equal and "Not Match" if not the same. The comparison exemplified in Figure 10 produces only the value of the SOI segment hash value. This shows that four other segments experienced changes in content when recoloring modifications occurred. The affected segments from each modified experiments shown in Table 6.

Table 6. Affected segment for image modification

No.	Modification Experiments	Affected Segment					
		SOI	APP1	DQT	SOF0	DHT	SOS
1	Recoloring	-	✓	✓	✓	✓	✓
2	Metadata Modification	-	✓	-	-	-	-
3	Resizing	-	✓	✓	✓	✓	✓
4	Convert to PNG	✓	✓	✓	✓	✓	✓
5	Text addition	✓	✓	✓	✓	✓	✓
6	Cropping	-	✓	✓	✓	✓	✓

Result from table 6 categorized into three groups. First group shown that metadata modification experiment affected APP1 segment only. Second group shown that SOI segment altered for image file conversion and text/object addition experiments. Third group shown all segment altered for image display modification. Each result group used as file fingerprint components. First component is SOI hash value and second component is APP1 hash value. Third component selected from four segments (DQT, SOF0, DHT, SOS) based on segments content and segment length.

SOS segment have the most length content from other segments. This condition made segment marker searching process need more time because Boyer-Moore string matching algorithm have *O(mn)* time complexity for worst case condition [7]. SOF0 have smallest size than others except SOI segment. SOF0 store image information such image dimension and number of color components that always change if image altered as in 1st, 2nd, 5th and 6th experiments. Figure 11 shown jpeg/exif file message digest that arranged from three hash values, SOI, APP1 and SOF0.

Figure 11. Example of jpeg/exif message digest structure

Hash value from each segments have 128 bit length. This caused of number format from application is hexadecimal which in binary can compute by 128 x 4.

4. CONCLUSION

Jpeg/exif message digest consist of three hash values that represent for modified experiments. SOI hash values use for identifying file convertion and text addition modification. APP1 hash values use for identifying metadata editing modification and SOF0 hash value for identifying recoloring, resizing and cropping modification. Identifying those three segment will make segments searching faster for Boyer-Moore string matching algorithm. Compilation of Message digest from SHA512 hash value have advantage because of its small size and cannot decrypt. Future research hope can developed message digest compilation for another file types that common use in digital communication such as video and audio.

REFERENCES

[1] H. Wijayanto, I. Riadi, and Y. Prayudi, "Encryption EXIF Metadata for Protection Photographic Image of Copyright Piracy," *Int. J. Res. Comput. Commun. Technol.*, vol. 5, no. 5, 2016.

[2] A. S. Sukarno, "Pengembangan Aplikasi Pengamanan Dokumen Digital Memanfaatkan Algoritma Advance Encryption Standard, RSA Digital Signature dan Invisible Watermarking," *Pros. Semin. Nas. Apl. Teknol. Inf. 2013*, pp. 1–8, NaN-5022, 2013.

[3] NIST, *FIPS PUB 180-4 Secure Hash Standard (SHS)*, no. August. Gaithersburg: National Institute of Standards and Technology, 2015.

[4] I. Riadi and M. Sumagita, "Analysis of Secure Hash Algorithm (SHA) 512 for Encryption Process on Web Based Application," *Int. J. Cyber-Security Digit. Forensics*, vol. 7, no. 4, 2018.

[5] L. Refialy, E. Sediyono, and A. Setiawan, "Pengamanan Sertifikat Tanah Digital Menggunakan Digital Signature SHA-512 dan," *JUTISI*, vol. 1, pp. 229–234, 2015.

[6] N. Jiji and T. Mahalakshmi, "Survey of Exact String Matching Algorithm for Detecting Patterns in Protein Sequence," *Adv. Comput. Sci. Technol.*, vol. 10, no. 8, pp. 2707–2720, 2017.

[7] K. Al-Khamaiseh and S. Al-Shagarin, "A Survey of String Matching Algorithms," *Int. J. Eng. Res. Appl.*, vol. 4, no. June 2015, pp. 144–156, 2014.

[8] D. R. Candra and K. D. Tania, "Application of Knowledge Sharing Features Using the algorithm Boyer-moore On Knowledge Management System (KMS)," *J. Sist. Inf.*, vol. 9, no. 1, pp. 1216–1221, 2017.

[9] A. L. . Sandoval, D. M. . Gonzales, L. J. . Villaba, and J. Hernandez-Castro, "Analysis of errors in exif metadata on mobile devices," *Multimed Tools Appl*, no. 74, pp. 4735–4763, 2015.

[10] N. Jiji and T. Mahalaksmi, "An Efficient String Matching Algorithm for Detecting Pattern Using Forward and Backward Searching Approach," *Int. J. Comput. Sci.*, vol. 6, no. 2, pp. 16–26, 2018.