A Cryptographic Tamper Detection Approach for Storage and Preservation of Forensic Digital Data Based on SHA 384 Hash Function

Isaac Baffour Senkyire[1,2,3] , Quist-Aphetsi Kester[1,2,3]
[1]Computer Science Department, Ghana Communication Technology University, Ghana
[2]Cyber Security Division, CRITAC, Ghana
[3]Directorate of Information Assurance and Intelligence Research, CRITAC, Ghana
isenkyire@gctu.edu.gh, kquist-aphetsi@gctu.edu.gh

***Abstract*— The current age permits for legal, official, sensitive, and confidential documents to be exchanged using digital channels among stakeholders. In this digital age, new advances in technology are tremendously vital creating more sophisticated and intelligent tools in areas like informatics, electronics, and telecommunication. In the wake of these new advances, any individual can digitize any kind of document, and modern computing tools can alter all these using computational drawing tools, such as GIMP or Paint Shop Pro, without causing any distortions hence, tampered documents can be presented with the same quality as the original documents. Documents tampered, when used or distributed illegally can cause economic, political, legal and moral damages to individuals and organizations. In this paper, we propose to detect the tampering of documents using SHA-384 hash function..**

Keywords- Cryptographic, tamper detection, SHA – 384, hash function, content authentication

I. INTRODUCTION

Almost all documents produced recently are in a digital form and stored in a specific file format such as Open Document Format (ODF) or Portable Document Format (PDF), among others, due to storage space reduction and rapid access that such file formats provide[1][2]. Many official, legal and confidential documents such as administrative and government documents, certificates, diploma are most often exchanged among stakeholders via digital channels for diverse purposes[3]. Most of these documents are very sensitive to be transferred online as regards contents, structure, syntax, and semantics[4]. Documents of such nature are used in government agencies, financial and educational institutions[3] military organizations and intelligence agencies[5]. Though some of these documents with specific file format such as PDF include some security mechanisms[2], advancement in third party applications, and smart devices default mechanisms coupled with technology advancement has made it easily accessible for malicious and unauthorized persons[6] to intercept, break and tamper the content of such documents during transmission yet present the same quality as the original document.

These tampered documents can cause economic, political, legal and moral damages to persons or institutions when they are used or distributed illegally[2]. Hence, several techniques and algorithms have been proposed by researchers for the security of information in these documents such as; content authentication, tamper detection, integrity verification, owner identification, copyright protection, and access control[4].

This paper focuses on tamper detection using a cryptographic approach hence, we propose SHA 384 to detect the tampering of forensic digital data.

The next sections of this paper consider related works done on preserving the authenticity and integrity of documents, the methodology, results and the conclusion.

II. RELATED WORKS

[5] proposed robust visible digital stamp (RVDS) to secure and protect documents that are important or sensitive using low-power computers that do not need any network connection type. The RVDS proposed by [5] converts the information in a document into a coded form of a customized quick response code (QR code). [5] then bound the code to the document for the verification process. The RVDS applies a combination of keys for the encryption and the process of authentication checks. [5] model (RVDS) does not require a network connection hence, it makes it an off-line process that guards users from the danger of losing their privacy. [7] proposed WAT-based image hashing and cryptographically created digital signature to authenticate paper documents. [7] calculated the image hash by extracting robust image features in wave atom transform (WAT) domain. They then encrypted the hash value in the person's original image with the private key of authorities that are trusted to form a digital signature which is encoded in a QR code printed on the document. [7] decrypted the digital signature extracted from the QR code with a public key of authorities trusted for identity verification. Their proposed approach provides the verification of printed documents offline without the need of online network access or database. [8] proposed an unsupervised approach to automatically detect forged documents by detecting the geometric distortions introduced during the process of forgery to scanned or re-engineered documents. They focused on the detection of the distortions done on fixed document parts such as headers, and footers that are mostly found on invoices. [8] used the matching quality between all documents pairs, and performed an outlier detection on the summed matching quality to spot the tampered document. They evaluated the quality of their approach on two public data sets, which recorded a true positive rate from 0.7 to 1.0. [9] proposed a feature extraction method for detecting the tampering of a printed

978-1-6654-2567-4/21/$31.00 ©2021 IEEE
DOI 10.1109/ICCMA53594.2021.00033

document. [9] calculated alphabets and numbers' median point, then they printed and scanned to verify the probability of the collision, the uniformity of the distribution, invariability during D/A and A/D transform, and the invariability at the time of the ordinal change of paper of their feature extracting method. The method proposed by [9] showed lower probability of the collision as compared with the existing one, their method also showed a half skew of the existing one as regards the uniformity of the distribution. Their method recorded an invariability of about thirtieth of the one of the existing one during D/A and A/D transformation. Finally [9] method showed an eighteenth invariability during the ordinal change of paper of the existing one. [9] compared their proposed method that uses median points as a feature to the existing method that leveraged on the area of the black dots as a feature and found their method to be better than the existing one. [10] proposed a tamper locating algorithm for DOCX document content authentication. [10] embedded an authentication watermark unrelated to the text content into the main setting file of the document which is named document.xml by the segmentation of the display character. In order to identify the integrity of the text content, [10] needed to detect whether the embedded watermark was the same as the authentication watermark. They therefore proposed a tamper calculation in their experiment, and it was concluded that their algorithm could always detect a tamper anytime the text was modified and the tampered places could also be located.

Varied techniques and methods have been used over time to secure forensic data , we seek to use SHA-384 to detect document tamper and the next section highlights the SHA-384 methodology.

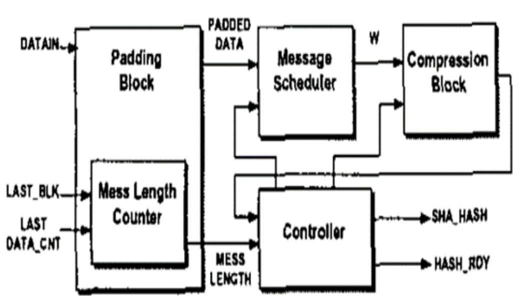

Figure 1. Outline of SHA-384 Design[11]

III. METHODOLOGY

SHA-384 operates on a message in 1024-bit blocks like the SHA-512 but it produces a 384-bit message digest. The maximum message length acceptable by the SHA-384 algorithm is 2^128bits. The SHA-384 algorithm consists of message padding, a message scheduler, compression block and controller. The controller controls the flow of data in the design. The padding block produces the padded 1024-bit data blocks needed by the message scheduler. The message length is calculated using counter and signals indicating the arrival of the last data block and its length. The SHA-384 algorithm is illustrated in Figure 1 and it is carried out as follows;

- Pad the message to length = 896 mod 1024
- Append the message length as a 128-bit binary number
- Parse message into Nx1024-bits blocks of data
- Initialize 8 x 64-bits words, A, B, C, D, E, F, G, and H such that,

A = 152 dcef8f07e9599
B = b74 1854efdb4a4fa
C = 4a4 f4b7d4b58e1ba
D = b9d bbccd59501d8e
E = 115 8e4b8a4786110
F = 00b 316736632cff7
G = 9a2 962637ad0c572
H = 71d 5901a3070d159 (1)

- Perform 80 iterations of the SHA-384 processing function, outlined in Figure 2, on the first 1024-bit data block
- The resulting 384-bit output initializes A, B, C, D, E, F, G, and H for the processing function of the next data
- After all N data blocks have been processed, the final output forms the 384-bit message digest

In the SHA-384 processing function, K_t are a sequence of eighty 64-bit constants. The message schedule W_t consists of 64-bit values such that,

$$W_t = \begin{cases} M_t & 0 \le t \le 15 \\ \sigma_1^{384}(W_{t-2}) + W_{t-7} + \sigma_0^{384}((W_{t-15}) + W_{t-16} & 16 \le t \le 79 \end{cases} \quad (2)$$

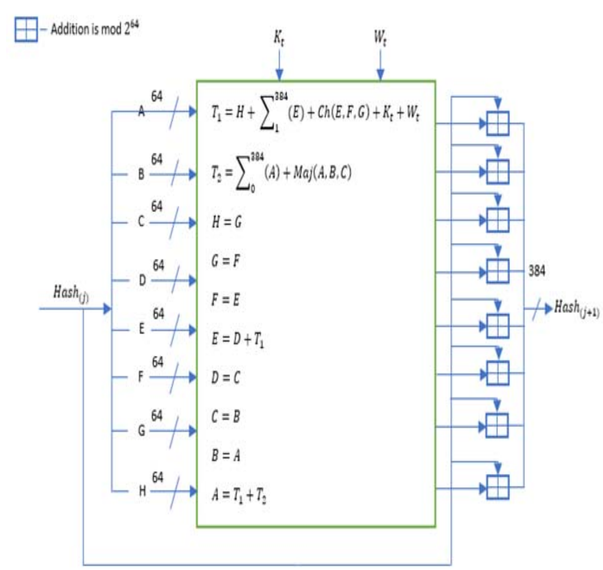

Figure 2. SHA-384 Processing Function[11]

The logical functions used in the message schedule and processing function are,

$$Ch(x,y,z) = (x\ AND\ y) \oplus (\bar{x}\ AND\ z) \qquad (3)$$
$$Maj(x,y,z) = (x\ AND\ y) \oplus (x\ AND\ z) \oplus (y\ AND\ z) \qquad (4)$$
$$\Sigma_0^{384}(x) = ROT_{RGT-28}(x) \oplus ROT_{RGT-34}(x) \oplus ROT_{RGT-39}(x) \qquad (5)$$
$$\Sigma_1^{384}(x) = ROT_{RGT-14}(x) \oplus ROT_{RGT-18}(x) \oplus ROT_{RGT-42}(x) \qquad (6)$$
$$\sigma_0^{384} = ROT_{RGT-1}(x) \oplus ROT_{RGT-8}(x) \oplus SHF_{RGT-7}(x) \qquad (7)$$
$$\sigma_1^{384} = ROT_{RGT-19}(x) \oplus ROT_{RGT-61}(x) \oplus SHF_{RGT-6}(x) \qquad (8)$$

Where $ROT_{RGT-n}(word)$ is a circular rotation of a word by n positions to the right and $SHF_{RGT-n}(word)$ is the right shifting of a word by n positions.

A shift register with a design mechanism of 16-stages is used to implement the SHA-384 message scheduler, as illustrated in Figure 3. The shift registers used in the implementation is informed by the diagrammatical representation of hash algorithms NIST provided. The 64-bit padded message blocks are loaded unto the registers over 16 clock cycles. On the next clock register 15 is then replaced with the outcome of equation (2). This process progresses for 80 clock cycles. Since the results of W_t is obtained from between registers 14 and 15, and not from the results of register 0, an initial 16 clock cycle delay is circumvented.

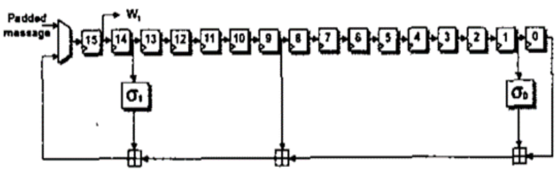

Figure 3. SHA-384 Message Scheduler Design[11]

In the compression block architecture, the shift register design methodology can also be used as illustrated in Figure 4. The compression block is the application of the processing function. In the design, 8 registers are used to hold the values of A to H as they are updated on each cycle. The functions $Ch(E,F,G)$, $Maj(A,B,C), \Sigma_0(A)$ and $\Sigma_1(E)$ are as represented in the equations (3), (4), (5), and (6) respectively[11].

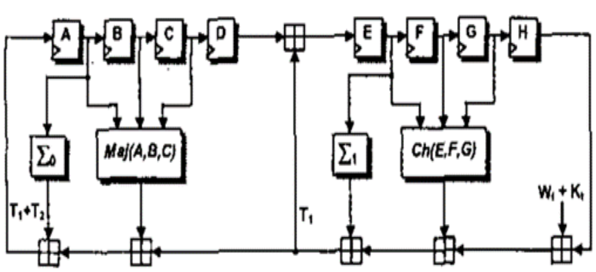

Figure 4. SHA -384 Compression Block Design[11]

With the proposed approach, a document D is divided into n parts $D_1, D_2, \dots, D_n$ and hashed as follows; D_1 is picked and a hash of H_1 is created, D_2 is also picked and combined with H_1 and a hash of H_2 is created, the hashing process continues till a final hash of H_n which combines D_n and hash H_{n+1} is produced as shown in Figure 5.

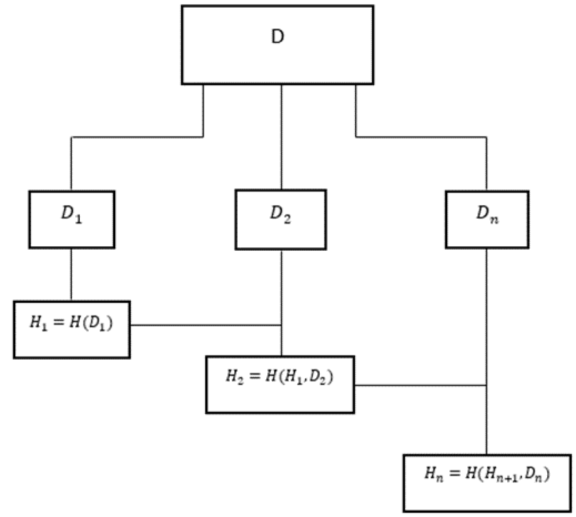

Figure 5. Document Hashing Process

IV. RESULTS

The image below is a sampled digital image of a forensic scene. The image is segmented into the RGB channels with their hash values computed.

Figure 6. Sample image of RGB image

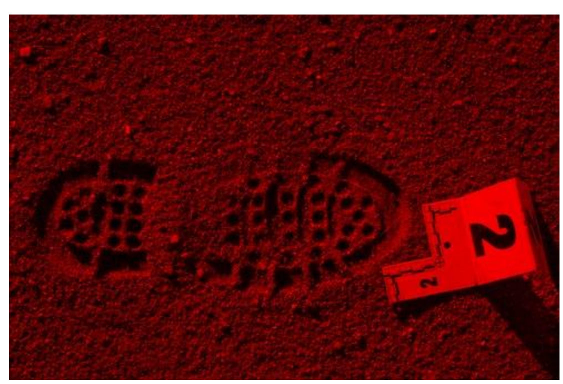

Figure 7. Sample image of R channel of the RGB image

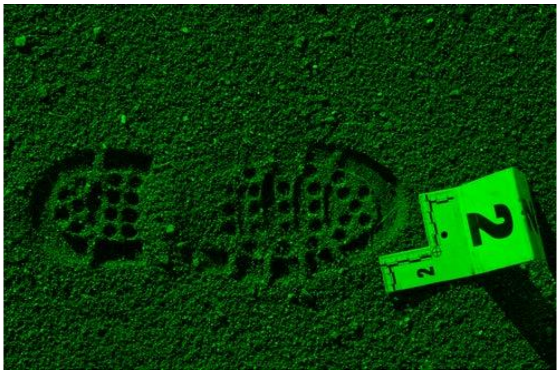

Figure 8. Sample image of G channel of the RGB image

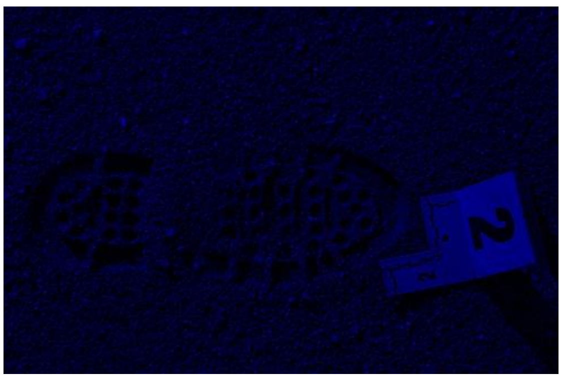

Figure 9. Sample image of B channel of the RGB image

A. SHA-384 Values of RGB

hex:
d531c9aae838677ded3f130b8a5caec0a4d918c9fa555e543c996d8b42b84f05f87dcaae9283b85cca0b4e61565f9fda

HEX:
D531C9AAE838677DED3F130B8A5CAEC0A4D918C9FA555E543C996D8B42B84F05F87DCAAE9283B85CCA0B4E61565F9FDA

h:e:x:
d5:31:c9:aa:e8:38:67:7d:ed:3f:13:0b:8a:5c:ae:c0:a4:d9:18:c9:fa:55:5e:54:3c:99:6d:8b:42:b8:4f:05:f8:7d:ca:ae:92:83:b8:5c:ca:0b:4e:61:56:5f:9f:da

base64:
1THJqug4Z33tPxMLilyuwKTZGMn6VV5UPJlti0K4TwX4fcqukoO4XMoLTmFWX5/a

B. SHA-384 Values of R

hex:
1e64c417c67362fe6a7f43a47c53f283597429e5af5956523550621b675597673d7f713e827ae686b242a52ba50da411

HEX:
1E64C417C67362FE6A7F43A47C53F283597429E5AF5956523550621B675597673D7F713E827AE686B242A52BA50DA411

h:e:x:
1e:64:c4:17:c6:73:62:fe:6a:7f:43:a4:7c:53:f2:83:59:74:29:e5:af:59:56:52:35:50:62:1b:67:55:97:67:3d:7f:71:3e:82:7a:e6:86:b2:42:a5:2b:a5:0d:a4:11

base64:
HmTEF8ZzYv5qf0OkfFPyg1l0KeWvWVZSNVBiG2dVl2c9f3E+gnrmhrJCpSulDaQR

C. SHA-384 Values of G

hex:
dea7484dc2a71d81ece3dd8989baa625e97b752a58f74ed642e431439fb6d33b21ed4dbadf3df37f67658377dc0d2b8a

HEX:
DEA7484DC2A71D81ECE3DD8989BAA625E97B752A58F74ED642E431439FB6D33B21ED4DBADF3DF37F67658377DC0D2B8A

h:e:x:
de:a7:48:4d:c2:a7:1d:81:ec:e3:dd:89:89:ba:a6:25:e9:7b:75:2a:58:f7:4e:d6:42:e4:31:43:9f:b6:d3:3b:21:ed:4d:ba:df:3d:f3:7f:67:65:83:77:dc:0d:2b:8a

base64:
3qdITcKnHYHs492JibqmJel7dSpY907WQuQxQ5+20zsh7U263z3zf2dlg3fcDSuK

D. SHA-384 Values of B

hex:
f1c0236487b5943153f2b4e236c377244bcb413fac0092d82382767ba3cab707a5d5050287314d479672bc7307139a26

HEX:
F1C0236487B5943153F2B4E236C377244BCB413FAC0092D82382767BA3CAB707A5D5050287314D479672BC7307139A26

h:e:x:
f1:c0:23:64:87:b5:94:31:53:f2:b4:e2:36:c3:77:24:4b:cb:41:3f:ac:00:92:d8:23:82:76:7b:a3:ca:b7:07:a5:d5:05:02:87:31:4d:47:96:72:bc:73:07:13:9a:26

base64:
8cAjZIe1lDFT8rTiNsN3JEvLQT+sAJLYI4J2e6PKtwel1QUChzFNR5ZyvHMHE5om

TABLE I. COMPUTED HASH VALUES FROM RGB CHANNELS

Image type	SHA-384 Values			
	hex	*HEX*	*h:e:x*	*base64*
RGB	d531c9a ae83867 7ded3f1 30b8a5c aec0a4d 918c9fa 555e543 c996d8b 42b84f0 5f87dcaa	D531C9AAE8 38677DED3F 130B8A5CAE C0A4D918C9 FA555E543C 996D8B42B8 4F05F87DCA AE9283B85C CA0B4E6156 5F9FDA	d5:31:c9:aa:e8: 38:67:7d:ed:3f: 13:0b:8a:5c:ae: c0:a4:d9:18:c9: fa:55:5e:54:3c: 99:6d:8b:42:b8: 4f:05:f8:7d:ca:a e:92:83:b8:5c:c a:0b:4e:61:56:5 f:9f:da	1THJqug4 Z33tPxM LilyuwKT ZGMn6V V5UPJlti0 K4TwX4f cqukoO4 XMoLTm FWX5/a

Image type	SHA-384 Values			
	hex	*HEX*	*h:e:x*	*base64*
	e9283b85cca0b4e61565f9fda			
R	1e64c417c67362fe6a7f43a47c53f283597429e5af5956523550621b675597673d7f713e827ae686b242a52ba50da411	1E64C417C67362FE6A7F43A47C53F283597429E5AF5956523550621B675597673D7F713E827AE686B242A52BA50DA411	1e:64:c4:17:c6:73:62:fe:6a:7f:43:a4:7c:53:f2:83:59:74:29:e5:af:59:56:52:35:50:62:1b:67:55:97:67:3d:7f:71:3e:82:7a:e6:86:b2:42:a5:2b:a5:0d:a4:11	HmTEF8ZzYv5qf0OkfFPyg1l0KeWvWVZSNVBiG2dVl2c9f3E+gnrmhrJCpSulDaQR
G	dea7484dc2a71d81ece3dd8989baa625e97b752a58f74ed642e431439fb6d33b21ed4dbadf3df37f67658377dc0d2b8a	DEA7484DC2A71D81ECE3DD8989BAA625E97B752A58F74ED642E431439FB6D33B21ED4DBADF3DF37F67658377DC0D2B8A	de:a7:48:4d:c2:a7:1d:81:ec:e3:dd:89:89:ba:a6:25:e9:7b:75:2a:58:f7:4e:d6:42:e4:31:43:9f:b6:d3:3b:21:ed:4d:ba:df:3d:f3:7f:67:65:83:77:dc:0d:2b:8a	3qdITcKnHYHs492JibqmJel7dSpY907WQuQxQ5+20zh7U263z3zf2dlg3fcDSuK
B	f1c0236487b5943153f2b4e236c377244bcb413fac0092d82382767ba3cab707a5d5050287314d479672bc7307139a26	F1C0236487B5943153F2B4E236C377244BCB413FAC0092D82382767BA3CAB707A5D5050287314D479672BC7307139A26	f1:c0:23:64:87:b5:94:31:53:f2:b4:e2:36:c3:77:24:4b:cb:41:3f:ac:00:92:d8:23:82:76:7b:a3:ca:b7:07:a5:d5:05:02:87:31:4d:47:96:72:bc:73:07:13:9a:26	8cAjZIe1lDFT8rTiNsN3JEvLQT+sAJLYI4J2e6PKtwel1QUChzFNR5ZyvHMHE5om

TABLE II. COMPUTED CHAIN HASH VALUES FROM RGB CHANNELS

Image type	SHA-384 Values			
	hex	*HEX*	*h:e:x*	*base64*
H_1(RGB)	d531c9aae838677ded3f130b8a5caec0a4d918c9fa555e543c996d8b42b84f05f87dcaae9283b85cca0b4e61565f9fda	D531C9AAE838677DED3F130B8A5CAEC0A4D918C9FA555E543C996D8B42B84F05F87DCAAE9283B85CCA0B4E61565F9FDA	d5:31:c9:aa:e8:38:67:7d:ed:3f:13:0b:8a:5c:ae:c0:a4:d9:18:c9:fa:55:5e:54:3c:99:6d:8b:42:b8:4f:05:f8:7d:ca:ae:92:83:b8:5c:ca:0b:4e:61:56:5f:9f:da	1THJqug4Z33tPxMLilyuwKTZGMn6VV5UPJlti0K4TwX4fcqukoO4XMoLTmFWX5/a
$H_2(H_1$,R)	3cb96902547e0c8bac2f8114a1a573f0a5284d822fdca23e4fb1ac0344712be8a8b00cc60abb9a54a66dceba065b3c90	3CB96902547E0C8BAC2F8114A1A573F0A5284D822FDCA23E4FB1AC0344712BE8A8B00CC60ABB9A54A66DCEBA065B3C90	3c:b9:69:02:54:7e:0c:8b:ac:2f:81:14:a1:a5:73:f0:a5:28:4d:82:2f:dc:a2:3e:4f:b1:ac:03:44:71:2b:e8:a8:b0:0c:c6:0a:bb:9a:54:a6:6d:ce:ba:06:5b:3c:90	PLlpAlR+DIusL4EUoaVz8KUoTYIv3KI+T7GsA0RxK+iosAzGCruaVKZtzroGWzyQ
$H_2(H_1$,G)	93c5b3841a81425b13a205450c844f192de916d16bbb96205ac744784d42b808fd294f39c069f62786c06f922698e07e	93C5B3841A81425B13A205450C844F192DE916D16BBB96205AC744784D42B808FD294F39C069F62786C06F922698E07E	93:c5:b3:84:1a:81:42:5b:13:a2:05:45:0c:84:4f:19:2d:e9:16:d1:6b:bb:96:20:5a:c7:44:78:4d:42:b8:08:fd:29:4f:39:c0:69:f6:27:86:c0:6f:92:26:98:e0:7e	k8WzhBqBQlsTogVFDIRPGS3pFtFru5YgWsdEeE1CuAj9KU85wGn2J4bAb5ImmOB+
$H_3(H_2$,B)	f867628350f802d9820de080ca4c38dbc8d339564567d6e837e6cde6d1609304ea1a98ca07e3c922ee00da43dafedb1f	F867628350F802D9820DE080CA4C38DBC8D339564567D6E837E6CDE6D1609304EA1A98CA07E3C922EE00DA43DAFEDB1F	f8:67:62:83:50:f8:02:d9:82:0d:e0:80:ca:4c:38:db:c8:d3:39:56:45:67:d6:e8:37:e6:cd:e6:d1:60:93:04:ea:1a:98:ca:07:e3:c9:22:ee:00:da:43:da:fe:db:1f	+Gdig1D4AtmCDeCAykw428jTOVZFZ9boN+bN5tFgkwTqGpjKB+PJIu4A2kPa/tsf

Table I consist of the independent hash values of RGB , R, G and B channels whilst table II consist of the connected hash values of the various channels.

V. CONCLUSION

The proposed approach makes it easy for experts to detect the tampering of documents using SHA-384 hash function because a change in pixel values of the image will affect the hash values. The strength of the connected hash value table is dependent on the length of the chain of hashes. Hence, multiple linked based connected hashes of evidence in a centralized or distributed database is encouraged for higher security.

REFERENCES

[1] L. Rosales-Roldan, M. Cedillo-Hernández, M. Nakano-Miyatake, and H. Pérez-Meana, "Watermarking-based tamper detection and recovery algorithms for official documents," in 2011 8th International Conference on Electrical Engineering, Computing Science and Automatic Control, 2011, pp. 1–6.

[2] M. González-Lee, M. Nakano-Miyatake, H. Pérez-Meana, and G. Sánchez-Pérez, "Script format document authentication scheme based on watermarking techniques," J. Appl. Res. Technol., vol. 13, no. 3, pp. 435–442, 2015.

[3] C. Vinh Loc, T. Cao De, J.-C. Burie, and J.-M. Ogier, "Content Region Detection and Feature Adjustment for Securing Genuine Documents," in 2020 12th International Conference on Knowledge and Systems Engineering (KSE), 2020, pp. 103–108.

[4] F. N. Al-Wesabi, K. Mahmood, and N. NEMRI, "A zero watermarking approach for content authentication and tampering detection of Arabic text based on fourth level order and word mechanism of Markov model," J. Inf. Secur. Appl., vol. 52, no. 102473, 2020.

[5] H. Al-Maksousy and H. Abdulhussein, "Robust Visible Digital Stamp for Instant Documents Authentication and Verification," in International Conference of Electromechanical Engineering and its Applications (ICEMEA-2020), 2020, vol. 765, no. 1.

[6] K. Quist-Aphetsi and I. Baffour Senkyire, "Validating of digital forensic images using SHA-256," in 2019 International Conference on Cyber Security and Internet of Things, ICSIoT 2019, 2019, pp. 118–121.

[7] F. Ahmad and L. M. Cheng, Paper Document Authentication Using Print-Scan Resistant Image Hashing and Public-Key Cryptography, vol. 11611 LNCS. Springer International Publishing, 2019.

[8] J. Van Beusekom and F. Shafait, "Distortion measurement for automatic document verification," in Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, 2011, pp. 289–293.

[9] Y. Takahashi, T. Yamada, and S. Susaki, "The estimation of the new feature extracting method for tamper detection in printed documents," in Proceedings - 2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, PACIIA 2008, 2008, vol. 2, pp. 43–47.

[10] G. Xin, X. Qi, and C. Ding, An Improved Tamper Detection and Location Scheme for DOCX Format Documents, vol. 11066 LNCS. Springer International Publishing, 2018.

[11] M. Mcloone and J. V Mccanny, "Efficient Single-Chip Implementation of SHA-384 and SHA-512," in 2002 IEEE International Conference on Field-Programmable Technology, 2002 (FPT), 2002, pp. 311–314..