Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.2023107319
Vol. 10, No. 6, Desember 2023, hlm. 1213-1222 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 30/E/KPT/2018 e-ISSN: 2528-6579

KEAMANAN DATA MENGGUNAKAN SECURE HASHING ALGORITHM (SHA)-256 DAN RIVEST SHAMIR ADLEMAN (RSA) PADA DIGITAL SIGNATURE

Juniar Hutagalung[*1], Puji Sari Ramadhan[2], Sarah Juliana Sihombing[3]

[1,2,3]STMIK Triguna Dharma, Medan
Email: [1]juniarhutagalung991@gmail.com, [2]pujisariramadhan@gmail.com, [3]sarahjulianaasihombing@gmail.com
[*]Penulis Korespondensi

(Naskah masuk: 30 Mei 2023, diterima untuk diterbitkan: 27 November 2023)

Abstrak

Penelitian ini mempelajari bagaimana kriptografi menggunakan Secure Hash Algorthm 256 (SHA-256) dan Rivest Shamir Adleman (RSA) untuk menjaga keaslian Surat Keterangan Lulus (SKL) dengan menggunakan kode Quick Response (QR) berbasis web. Salah satu fungsi utama SHA-256 adalah menerima input data M dalam bentuk apa pun dan menghasilkan nilai hash h(M). RSA, algoritma yang memiliki asimetri, kunci publik dan kunci privat. Ini adalah sistem kriptografi terpopuler yang digunakan untuk memberikan tanda tangan digital, kerahasiaan, dan kunci. Salah satu fitur utama sistem otentikasi dokumen adalah mencetak dan verifikasi dokumen, menghasilkan tanda tangan digital dan mengubahnya menjadi kode QR. Dengan menggunakan sistem otentikasi dokumen berbasis web, orang dapat memastikan bahwa dokumen yang telah ditandatangani atau dibubuhi dengan kode QR. Ini membantu menjaga keaslian dokumen agar orang lain tidak dapat menyalinnya. Untuk mencegah pemalsuan surat, integritas data, dan keabsahan data, isi dokumen yang dilindungi disandikan menggunakan algoritma kriptografi SHA-256 dan RSA. Data yang digunakan yaitu gabungan nama depan + nis siswa (ELIA + 18001). Hasilnya kemudian dimasukkan ke dalam dokumen ringkasan terenkripsi. Algoritma kriptografi untuk SKL berhasil digunakan. Setelah menggunakan metode SHA-256 dan RSA, hasil akhir adalah "09C6423CF09C9E61BC09B29966DE6CB569A7DE2C4349FA2B52F 08EF39D07140F". Hasil dekripsi sebanding dengan hash e-dokumen, sehingga dokumen yang diverifikasi dianggap sah.

Kata kunci: *digital signature, kriptografi, QR Code, RSA, SHA-256, SKL*

IMPLEMENTATION OF SECURE HASHING ALGORITHM (SHA)-256 AND RIVEST SHAMIR ADLEMAN (RSA) ON DIGITAL SIGNATURE

Abstract

This research studies how cryptography uses the Secure Hash Algorthm 256 (SHA-256) and Rivest Shamir Adleman (RSA) to maintain the authenticity of the Pass Certificate (SKL) using a web-based Quick Response (QR) code. One of the main functions of SHA-256 is to accept M input data of any kind and output a hash value of h(M). RSA, an algorithm that has asymmetry, public key and private key. It is the most popular cryptographic system used to provide digital signatures, secrecy and keys. One of the main features of a document authentication system is printing and verifying documents, generating digital signatures and converting them into QR codes. By using a web-based document authentication system, people can ensure that documents have been signed or affixed with a QR code. This helps maintain the authenticity of documents so that others cannot copy them. To prevent letter forgery, data integrity, and data validity, the contents of protected documents are encoded using the SHA-256 and RSA cryptographic algorithms. The data used is a combination of the student's first name + nis (ELIA + 18001). The results are then entered into an encrypted summary document. The cryptographic algorithm for SKL was successfully used. After using the SHA-256 and RSA methods, the final result is "09C6423CF09C9E61BC09B29966DE6CB569A7DE2C4349FA2B52F 08EF39D07140F". The decryption result is comparable to the e-document hash, so the verified document is considered valid.

Keywords: *digital signature, cryptography, QR Code, RSA, SHA-256, SKL*

1. PENDAHULUAN

Salinan ijazah diperlukan untuk tujuan administratif saat mendaftar di sekolah atau universitas dan melamar pekerjaan. Siswa yang melewati ujian dan lulus sekolah biasanya tidak memiliki ijazah. Mendapatkan ijazah asli dari

Kementerian Pendidikan dan Kebudayaan membutuhkan waktu empat hingga lima bulan. Ini adalah peristiwa yang terjadi di Pusat Kegiatan Belajar Masyarakat Hati Nurani Baru (PKBM Hanuba) yang terletak di Medan. PKBM Hanuba adalah institusi pendidikan informal. Sekitar 250 siswa memperoleh gelar dari tiga program paket setiap tahun. Paket A adalah untuk SD, paket B untuk SMP, dan paket C untuk SMA. Bagi siswa yang lulus ujian akhir, sekolah akan memberikan Surat Keterangan Lulus (SKL).

SKL adalah ijazah sementara sampai ijazah resmi diterbitkan. Ijazah adalah dokumen yang sah secara hukum yang menunjukkan seseorang telah menyelesaikan pendidikan (Nugroho, 2020). Kepala sekolah menyetujui SKL, yang didasarkan pada ujian sekolah dan nasional. Siswa harus memiliki SKL untuk mendaftar di perguruan tinggi atau melamar pekerjaan.

Pemalsuan SKL sering dilakukan untuk keuntungan sendiri (Pramesti *et al.*, 2019). Memberikan bukti kualifikasi pendidikan tanpa mengikuti ujian, dapat merugikan banyak orang. Akibatnya, pengamanan diperlukan untuk mencegah ijazah palsu. Selain itu, sistem keamanan ini menawarkan layanan untuk pengecekan kehandalan dokumen dan keakuratan data secara cepat dan mudah. Kriptografi memainkan peran yang sangat penting untuk mengatasi masalah tersebut. Kriptografi memiliki prosedur untuk menyandikan (enkripsi dan dekripsi) data atau informasi agar aman, termasuk tanda tangan digital.

Tanda tangan digital tidak seperti tanda tangan orang yang dimasukkan ke komputer untuk scanning. Sebaliknya, mereka berfungsi sebagai penanda yang memastikan bahwa data tersebut asli asli (Nuraeni, et al., 2018). Tanda tangan digital menggunakan algoritma enkripsi untuk meringkas konten dokumen yang dilindungi dan memasukkan hasilnya ke dalam dokumen. Ringkasan isi dokumen dikumpulkan, dikodekan, dan dimasukkan dalam tanda tangan digital.

Dengan menggunakan fungsi hashing, dapat meringkas isi dokumen. Algoritma *Secure Hash Algorithm* (SHA) adalah salah satu contoh fungsi hashing yang terbukti, yang menghasilkan inti pesan (Prabowo & Afrianto, 2017). Sebuah algoritma yang dikenal sebagai pesan digest dapat mengubah teks atau pesan menjadi serangkaian karakter acak dengan jumlah karakter yang sama sama (Cahyono, 2018). Algoritma enkripsi kunci publik (asimetris), misalnya RSA, digunakan untuk mengkodekan nilai hash. Algoritma ini cocok untuk enkripsi dan tanda tangan. RSA biasanya digunakan untuk pertukaran kunci dan penandatanganan digital, tetapi algoritma enkripsi simetris (seperti AES) digunakan untuk mengenkripsi data yang sebenarnya karena lebih efisien (Anshori, et al., 2019). Hanya pemilik kunci privat yaitu pemilik dokumen dapat menggunakan algoritma kunci publik untuk mengenkripsi nilai hash dan memberikan tanda tangan digital. Penerima dokumen publik juga dapat divalidasi dengan kunci publik. Tanda tangan digital ditambahkan ke dokumen pada langkah terakhir.

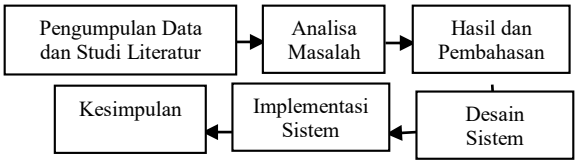
Beberapa peneliti menggunakan kriptografi karena dapat menggabungkan metode enkripsi simetris dan asimetris yang umum untuk mengamankan dokumen digital dengan efisiensi tinggi (Yonathan et al., 2021). Keakuratan sangat penting dalam kriptografi karena kesalahan kecil dalam enkripsi, dekripsi, atau tanda tangan digital dapat menghasilkan hasil yang salah (Syaifuddin, et al., 2021). Selama enkripsi dan dekripsi, kunci acak hanya digunakan sekali dalam kriptografi. (Silitonga & Pakpahan, 2021). Setiap teknik enkripsi, penerapan yang tepat, dan keamanan umum sangat penting. (Yuniati & A, 2020). Karakter dalam teks mengalami transformasi matematis ketika kriptografi digunakan untuk membuat prototipe enkripsi dan dekripsi kata sandi (Babu, 2017).

Setiap metode kriptografi harus mempertimbangkan elemen seperti kekuatan enkripsi RSA, ukuran pesan yang dapat disisipkan, dan kemungkinan kerentanannya terhadap berbagai serangan (Handoyo, et al., 2018). Kriptografi membuat data tidak hanya dienkripsi untuk keamanan, tetapi juga disembunyikan dalam media yang mungkin terlihat alami sehingga orang yang tidak berwenang tidak akan menemukannya (Rosalina & Hadisukmana, 2019). Kriptografi berbasis web tipe RSA digunakan untuk melindungi aplikasi dengan memasukkan kunci publik yang telah dibuat sebelumnya (Anjaswari, et al., 2020). Karena algoritma SHA-512 menggunakan token untuk otentikasi, itu dapat mempercepat dan meningkatkan keamanan (Setiawan & Purnamasari, 2020).

Pengamanan data dengan tanda tangan digital sangat penting untuk PKBM Hanuba Medan agar dapat memberikan layanan terbaik kepada siswa yang telah lulus ujian akhir.

2. METODE PENELITIAN

Untuk tandatangan digital, algoritma SHA-256 dan RSA digunakan. Tahapan penelitian terdiri dari pengumpulan data dan penelitian literatur, melakukan analisis masalah, hasil dan pembahasan, desain dan implementasi sistem, kemudian menarik kesimpulan, seperti yang ditunjukkan pada Gambar 1.



Gambar 1. Tahapan metode penelitian

2.1. Pengumpulan Data dan Studi Literatur

Peneliti melakukan observasi di PKBM Hanuba Medan dan mewawancarai Ketua dan pengelola lembaga, Bapak Jontar Sinaga, SE. Tabel 1 menunjukkan bagaimana data yang diambil dari SKL akan diubah menjadi tanda tangan digital.

Tabel 1. Data Awal

Nama Depan	Nomor Induk Siswa
ELIA	18001

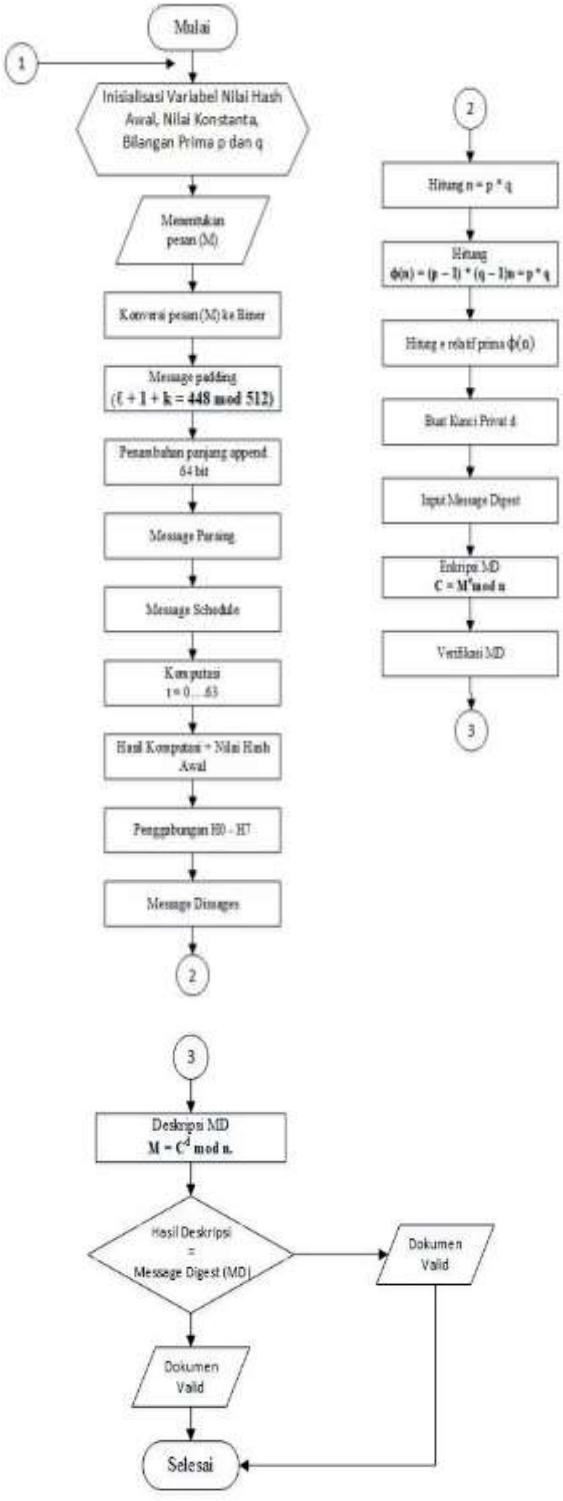
Melakukan penelitian lebih lanjut tentang literatur yang berkaitan dengan kriptografi dengan algoritma SHA-256 dan RSA. Algoritma ini digunakan untuk enkripsi dan dekripsi, serta pembangkit kunci publik dan privat.

Kriptografi adalah bidang yang menyelidiki data dengan membuat angka acak yang tidak dapat dipahami oleh orang lain. Kriptografi yang baik memastikan kerahasiaan, integritas, non-repudiation, dan kerahasiaan data (Kusuma , et al., 2018). Algoritma klasik dan modern digunakan dalam kriptografi (Fauziah, et al., 2018). RSA pada dasarnya adalah metode enkripsi dan otentikasi Internet, jadi tidak perlu mengirimkan kunci pribadi ke seluruh Internet. RSA menggunakan proses pembangkitan kunci, enkripsi, dan dekripsi (Yousif, 2018). Algoritma RSA menggunakan kunci dan blok enkripsi yang bervariasi. Faktor pertama digunakan untuk membuat kunci dan menyebarkannya secara publik, sedangkan kunci privat disimpan untuk pengirim dan penerima (Paramita & Sudibyo, 2021). Algoritma RSA banyak digunakan untuk enkripsi dan dekripsi informasi dalam kriptografi asimetris/kunci publik (Khan , et al., 2018). RSA adalah algoritma kriptografi terbaik saat ini, yang memastikan komunikasi aman melalui jaringan (Nisha & Farik, 2017).

Fungsi hash digunakan oleh layanan keamanan jaringan untuk otentikasi pesan dan keutuhan data (Sembiring, et al., 2019). Salah satu fungsi hash satu arah adalah SHA-256. Ini dianggap aman karena tidak memungkinkan mendapatkan pesan yang berhubungan dengan intisari pesan yang sama (Mulyadi , et al., 2018). Setelah menerima input pesan, algoritma SHA-256 dapat melakukan padding (Anugrah, et al., 2019). Jadwal pesan dapat diubah dengan algoritma SHA-256 (Sulastri & Putri, 2018). Fungsi hash SHA-256 dan algoritma RSA digunakan untuk menghasilkan digital signature (Lorien & Wellem, 2021).

2.2. Analisa Masalah

Untuk menyelesaikan masalah tanda tangan digital SKL pada PKBM Hanuba Medan, langkah pertama adalah menentukan titik masalah sebenarnya. Gambar 2 menunjukkan diagram alur kerja yang digunakan untuk menerapkan metode SHA-256 dan RSA dalam implementasi tanda tangan digital.



Gambar 2. Flowchart algoritma SHA-256 dan RSA

2.3. Fungsi Hash

SHA diciptakan oleh National Security Agency, SHA adalah fungsi hash satu arah yang diterbitkan oleh National Institute of Standards and Technology sebagai Federal Information Processing Standard pada tahun 1993. Sebelumnya dikenal sebagai SHA-0, generasi berikutnya, SHA-1 dirilis pada tahun berikutnya, yang merupakan perbaikan dari algoritma

SHA-0. Pada tahun 2002, empat versi tambahan dari algoritma ini dirilis: SHA-224, SHA-256, SHA-384, dan SHA-512 (FIPS 180-3, 2008).

Parameter fungsi hash yang sering digunakan ditunjukkan dalam Tabel 2.

Tabel 2. Parameter Fungsi Hash

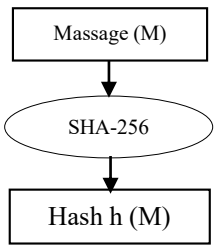
Parameter	Keterangan
M	Pesan
h	Fungsi *hash*
Y	Merupakan *h(M)* atau *message digest*

Tabel 3 menampilkan daftar algoritma fungsi hash berikut yang umum digunakan.

Tabel 3. Daftar Algoritma Fungsi Hash

Algoritma	Output Size	Internal State Size	Block Size	Word Size		Collision
HAVAL	256/224 /192/160/128	256	1024	64	32	Yes
MD2	128	384	128	No	8	Almost
MD4	128	128	512	64	32	Yes
MD5	128	128	512	64	32	Yes
PANAMA	256	8736	256	No	32	With flaws
RIPEMD	128	128	512	64	32	Yes
RIPEMD-128/256	128/256	128/256	512	64	32	No
RIPEMD-160/320	160/320	160/320	512	64	32	No
SHA-0	160	160	512	64	32	Yes
SHA-1	160	160	512	64	32	With flaws
SHA-256/224	256/224	256	512	64	32	No
SHA-512/384	512/384	512	1024	128	64	No
Tiger(2)-192/160/128	192/160 /128	192	512	64	64	No
VEST-4/8	160/256	176/304	8	80	1	No
VEST-16/32	320/512	424/680	8	88	1	No
WHIRLPOOL	512	512	512	256	8	No

Gambar 3 menunjukkan fungsi utama SHA-256.



Gambar 3. Fungsi SHA-256

2.4. *Rivest Shamir Adleman (RSA)*

Tiga peneliti, Ronald Rivest, Adi Shamir, dan Len Adleman, adalah pencipta algoritma kriptografi kunci publik RSA, yang digunakan secara luas untuk mengamankan pengiriman data (Sinlae, Ngaga and Mau, 2018).

Beberapa parameter, yaitu p, q, n, ø (n), e, d, K publik dan K privat, diperlukan untuk membangkitkan kunci RSA. Keterangan untuk masing-masing parameter disajikan dalam Tabel 4.

Tabel 4. Parameter Pembangkit Kunci RSA

Parameter	Keterangan
P,Q	Bilangan prima
N	Merupakan hasil dari $p \times q$
ø (n)	$(p-1) \times (q-1)$
E	gcd (ø (n), e) = 1
D	e^{-1} *(mod ø (n))*
K publik	Menggunakan algoritma *extended euclid (e,n)*
K privat	*D*

Tabel 5 menunjukkan algoritma enkripsi RSA.

Tabel 5. Algoritma Enkripsi RSA

Algortima Enkripsi RSA	
Input	*Kpublik = (e,n)*
Output	$C = P^e \bmod n$

Tabel 6 menunjukkan algoritma dekripsi RSA, yang menggunakan kunci privat, merupakan fungsi eksponensial modular n.

Tabel 6. Algoritma Denkripsi RSA

Algortima Dekripsi RSA	
Input	*Kprivat = d* *Kpublik = (e,n)*
Output	$P = C^d \bmod n$

3. HASIL DAN PEMBAHASAN

Tahap hasil dan diskusi ini mencakup penerapan Secure Hash Algorthm 256 (SHA-256) dan Rivest Shamir Adleman (RSA), seperti yang ditunjukkan di bawah ini.

3.1. Penerapan SHA-256

Proses atau tahapan algoritma SHA-256, yaitu:

1. Menentukan pesan (M) yang diambil dari data SKL. Data yang digunakan yaitu gabungan nama depan + nis siswa.
   Contoh : ELIA + 18001.
   M = ELIA18001.
   Ubah pesan (M) menjadi bilangan biner, sehingga menjadi :
   E = 01000101 1 = 00110010
   L = 01001100 8 = 00111000
   I = 01001001 0 = 00110000
   A = 01000001 0 = 00110000

1 = 00110010

Maka M = 01000101 01001100 01001001 01000001 00110010 00111000 00110000 00110000 00110010

2. *Padding*, yaitu penambahan bit pengganjal sehingga total panjangnya 512 bit.
   M = 01000101 01001100 01001001 01000001 00110010 00111000 00110000 00110000 00110010
   Sehingga ℓ = 72 bit.
   Kemudian tambahkan bit ‘1’ dan sisanya adalah bit ‘0’ sejumlah k. Dimana k :

   ℓ + 1 + k = 448 mod 512

   k = ℓ + 1 = 448 mod 512
   = 72 + 1 = 448 mod 512
   = 73 = 448 mod 512
   = 448 – 73 = **375**

   Maka banyak bit ‘0’ yang ditambahkan sebanyak 375 bit. Pada akhir pesan yang di *padding* dijumlahkan
   ℓ = 72 = 01001000.
   Pada Tabel 7 terlihat hasil pesan yang di *padding* [$M^{(0)}$], sebagai berikut.

Tabel 7. Hasil Padding Pesan

0100	0100	0100	0100	0011	0011	0011	0011
0101	1100	1001	0001	0010	1000	0000	0000
0011	1000	0000	0000	0000	0000	0000	0000
0010	0000	0000	0000	0000	0000	0000	0000
0000	0000	0000	0000	0000	0000	0000	0000
0000	0000	0000	0000	0000	0000	0000	0000
0000	0000	0000	0000	0000	0000	0000	0000
0000	0000	0000	0000	0000	0000	0000	0000
0000	0000	0000	0000	0000	0000	0000	0000
0000	0000	0000	0000	0000	0000	0000	0000
0000	0000	0000	0000	0000	0000	0000	0000
0000	0000	0000	0000	0000	0000	0000	0000
0000	0000	0000	0000	0000	0000	0000	0000
0000	0000	0000	0000	0000	0000	0000	0000
0000	0000	0000	0000	0000	0000	0000	0100
0000	0000	0000	0000	0000	0000	0000	1000

3. *Parsing* pesan, yaitu setiap blok 512 bit dibagi 16 buah *word* 32 bit. Hasil parsing pesan dapat di lihat pada Tabel 8.

Tabel 8. Hasil Parsing Pesan

$M_0^{(0)}$	01	01	01	11	01	10	01	00
	00	01	00	00	00	01	00	01
$M_1^{(0)}$	00	00	00	10	00	00	00	00
	11	10	11	00	11	00	11	00
$M_2^{(0)}$	00	00	10	00	00	00	00	00
	11	10	00	00	00	00	00	00
$M_3^{(0)}$	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00
$M_4^{(0)}$	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00
$M_5^{(0)}$	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00
$M_6^{(0)}$	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00
$M_7^{(0)}$	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00
$M_8^{(0)}$	01	01	01	11	01	10	01	00
	00	01	00	00	00	01	00	01
$M_9^{(0)}$	00	00	00	10	00	00	00	00
	11	10	11	00	11	00	11	00
$M_{10}^{(0)}$	00	00	10	00	00	00	00	00
	11	10	00	00	00	00	00	00
$M_{11}^{(0)}$	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00
$M_{12}^{(0)}$	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00
$M_{13}^{(0)}$	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00
$M_{14}^{(0)}$	00	00	00	00	00	00	00	00
	00	00	00	00	00	00	00	00
$M_{15}^{(0)}$	00	00	00	00	00	00	01	10
	00	00	00	00	00	00	00	00

4. *Message Schedule*. Langkah ini diawali dengan mengubah setiap blok pesan menjadi bilangan *hexadesimal* sesuai pada Tabel 9 sebagai berikut:

Tabel 9. Hasil Message Schedule

454C 4941	3238 3030	3280 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000
0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000
BAC 0 AE37	4B5 D 3CD 0	7573 9A13	39E0 FB8 A	BE56 88F1	62B A 23F2	9549 7D43	1060 81B 9
DC8 A 2722	C663 4C2 F	9123 12E7	8DF A 0C1F	4DC 5 9663	DCF 8 C704	9F5C 3B8 B	AB8 0 6D6 6
232A 2770	4C21 37C C	C219 B6D 5	B7C 3 1E6F	E93E D7A E	19D A A40 A	4225 65E2	29D F CFF F
02AE FDC 5	C046 C95 C	97E4 12F8	323C 185A	168E B8C 7	B880 D61 6	51A9 8724	F283 0564
D4FF 08CE	A49 A 07C5	59A1 5E51	06A3 34A A	E4B3 C952	FDF E 8D7 4	DAF 9 F99A	C87 E 38D B
A9A E 5538	EF0 C 0452	C0B 0 5BA E	BF48 41F4	4B04 8FE1	DFC 5 618 D	683D CBD 2	2397 88F A

5. Inisialisasi Variabel dan Konstanta
   Variabel awal pada fungsi *hash* SHA-256 sebagai berikut :
   a = $H_0^{(0)}$ = 6A09E667
   b = $H_1^{(0)}$ = BB67AE85
   c = $H_2^{(0)}$ = 3C6EF372
   d = $H_3^{(0)}$ = A54FF53A
   e = $H_4^{(0)}$ = 510E527F
   f = $H_5^{(0)}$ = 9B056887
   g = $H_6^{(0)}$ = 1F83D9AB
   h = $H_7^{(0)}$ = 5BE0CD19
   Pada Tabel 10 terlihat nilai konstanta pada fungsi *hash* SHA-256.

Tabel 10. Nilai Konstanta

428A 2F98	7137 4491	B5C 0FB CF	E9B5 DBA 5	3956 C25 B	59F1 11F1	923F 82A4	AB1 C5E D5
D807 AA9 8	1283 5B0 1	2431 85B E	550C 7DC 3	72B E5D 74	80DE B1FE	9BD C06 A7	C19 BF17 4

		B5C	E9B5	3956			AB1
428A	7137	0FB	DBA	C25	59F1	923F	C5E
2F98	4491	CF	5	B	11F1	82A4	D5
E49	EFB	0FC1	240C	2DE	4A74	5CB	76F9
B69	E478	9DC	A1C	92C6	84A	0A9	88D
C1	6	6	C	F	A	DC	A
	A83			C6E	D5A	06C	
983E	1C6	B003	BF59	00B	7914	A635	1429
5152	6D	27C8	7FC7	F3	7	1	2967
	2E1	4D2		650	766A		
27B7	B21	C6D	5338	A73	0AB	81C2	9272
0A85	38	FC	0D13	54	B	C92E	2C85
A2B	A81	C24	C76	D19			
FE8	A66	B8B	C51	2E81	D699	F40E	106A
A1	4B	70	A3	9	0624	3585	A070
	1E37		34B0	391C	4ED8	5B9	
19A4	6C0	2748	BCB	0CB	AA4	CCA	682E
C116	8	774C	5	3	A	4F	6FF3
	78A		8CC	90B	A450	BEF	
748F	5636	84C8	7020	EFF	6CE	9A3F	C671
82EE	F	7814	8	FA	B	7	78F2

6. Komputasi hash: menggunakan fungsi SHA-256 untuk melakukan proses komputasi untuk t = 0, sampai t = 63 sesuai dengan fungsinya. Fungsi SHA-256 adalah sebagai berikut.

For t = 0 to t = 63 :
{
$T_1 = h + \sum_1^{(256)}(e) + Ch(e,f,g) + K_t^{(256)} + Wt$
$T_2 = \sum_0^{(256)}(a) + Maj(a,b,c)$
h = g, g = f, f = e, e = d + T_1
d = c, c = b, b = a, a = $T_1 + T_2$

$\sum_1^{(256)}(e)$ = (e *ROTR* 6) ⊕ (e *ROTR* 11) ⊕ (e *ROTR* 25)
$\sum_0^{(256)}(a)$ = (e *ROTR* 2) ⊕ (e *ROTR* 13) ⊕ (e *ROTR* 22)
$K_t^{(256)}$ = Konstanta SHA-256
Ch (e, f, g) = (e ^ f) ⊕ (~e ^ g)
Maj (a, b, c) = (a ^ b) ⊕ (a ^ c) ⊕ (b ^ c)
ROTR = *Rotate Right*
⊕ = fungsi *XOR*
^ = fungsi *AND*

Pada Tabel 11 terlihat proses komputasi pada fungsi *hash* SHA-256.

Tabel 11. Proses Komputasi Fungsi Hash SHA-256

	A	B	C	D	E	F	G	H
	C8C	6A0	BB6	3C6	FCA	510	9B0	1F8
t=	A	9	7	E	8	E	5	3
0	123	E66	AE8	F37	6AC	527	688	D9A
	6	7	5	2	3	F	C	B
	0FD	C8C	6A0	BB6	5627	FCA	510	9B0
t=	0	A	9	7		8	E	5
1	A63	123	E66	AE8	D985	6AC	527	688
	8	6	7	5		3	F	C
	501	0FD	C8C	6A0	9DCD	562	FC	510
t=	A	0	A	9		7	A8	E
2	C2E	A63	123	E66	9677	D98	6A	527
	3	8	6	7		5	C3	F
	19E	501	0FD	C8C	D87	9DC	562	FCA
t=	6	A	0	A	F	D	7	8
3	A0F	C2E	A63	123	AFA	967	D98	6AC
	D	3	8	6	C	7	5	3

Setelah di dapat ke 64 putaran dari komputasi fungsi *hash*, kemudian dilakukan penjumlahan hasil putaran ke 64 (t = 63) dengan variabel awal *hash* SHA-256. Maka hasil yang didapatkan yaitu :

$H_0^{(0)}$ = 9FBC5BD5 + 6A09E667 = 09C6423C
$H_1^{(0)}$ = 3534EFDC + BB67AE85 = F09C9E61
$H_2^{(0)}$ = 7F9ABF27 + 3C6EF372 = BC09B299
$H_3^{(0)}$ = C18E777B + A54FF53A = 66DE6CB5
$H_4^{(0)}$ = 18998BAD + 510E527F = 69A7DE2C
$H_5^{(0)}$ = A84491A4 + 9B056887 = 4349FA2B
$H_6^{(0)}$ = 336CB548 + 1F83D9AB = 52F08EF3
$H_7^{(0)}$ = 412646F6 + 5BE0CD19 = 9D07140F

7. Penggabungan (H)
$H_0 \| H_1 \| H_2 \| H_3 \| H_4 \| H_5 \| H_6 \| H_7$
09C6423C‖F09C9E61‖BC09B299‖66DE6CB5 ‖69A7DE2C‖4349FA2B‖52F08EF3 ‖9D07140F
8. Nilai *Hash*
09C6423CF09C9E61BC09B29966DE6CB569 A7DE2C4349FA2B52F08EF39D07140F.

3.2. Penerapan RSA

Pembangkit Kunci:
1. Bangkitkan dua bilangan prima (p dan q),
2. Cari n = p * q
   n = 7 * 13
   n = 91
3. Cari ϕ(n) = (p – 1) * (q – 1)
   ϕ(n) = (7 – 1) * (13 – 1)
   ϕ(n) = 72
4. Pilih e dengan nilai relatif prima terhadap ϕ(n), yaitu 5, 5 relatif prima terhadap 72.
5. Tentukan kunci privat d dengan a. Oleh karena itu, nilai d ditemukan melalui proses berikut: a = 0 = d = 1/5 (tidak memenuhi), a = 0 = d = 73/5 (tidak memenuhi), dan a = 2 = d = 145/5 = 29 (memenuhi). Dengan demikian, kunci publik (n = 91, e = 5) dan kunci privat (n = 91, e = 29) diperoleh.

Setelah dokumen elektronik diubah menjadi paket pesan, langkah berikutnya adalah mengenkripsi paket pesan menggunakan algoritma RSA dan kunci privat yang diciptakan. Contoh pesan yang akan dienkripsi adalah hasil hash dari contoh hash sebelumnya, yang berisi "09C6423CF09C9E61BC09B29966DE6CB569A7DE2C4349FA2B52F08EF39D07140F".

1. Menggunakan tabel ASCII untuk mengubah setiap karakter menjadi desimal. ”48 57 67 54 52 50 51 67 70 48 57 67 57 69 54 49 66 67 48 57 66 50 57 57 54 54 68 69 54 67 66 53 54 57 65 55 68 69 50 67 52 51 52 57 70 65 50 66 53 50 70 48 56 69 70 51 57 68 48 55 49 52 48 70”
2. Kemudian ubah ke chipertext menggunakan rumus $C = M^e \bmod n$.

Pada Tabel 12 terlihat hasil *chipertext*, seperti di bawah ini.

Tabel 12. Hasil Chipertext

$48^5 \bmod 91 = 55$	$57^5 \bmod 91 = 57$	$67^5 \bmod 91 = 58$	$54^5 \bmod 91 = 45$
$52^5 \bmod 91 = 26$	$50^5 \bmod 91 = 85$	$51^5 \bmod 91 = 25$	$67^5 \bmod 91 = 58$
$70^5 \bmod 91 = 70$	$48^5 \bmod 91 = 55$	$57^5 \bmod 91 = 57$	$67^5 \bmod 91 = 58$
$57^5 \bmod 91 = 68$	$69^5 \bmod 91 = 62$	$54^5 \bmod 91 = 45$	$49^5 \bmod 91 = 56$
$66^5 \bmod 91 = 40$	$67^5 \bmod 91 = 58$	$48^5 \bmod 91 = 55$	$57^5 \bmod 91 = 57$
$66^5 \bmod 91 = 40$	$50^5 \bmod 91 = 85$	$57^5 \bmod 91 = 57$	$57^5 \bmod 91 = 57$
$54^5 \bmod 91 = 45$	$54^5 \bmod 91 = 45$	$68^5 \bmod 91 = 87$	$69^5 \bmod 91 = 62$
$54^5 \bmod 91 = 45$	$67^5 \bmod 91 = 58$	$66^5 \bmod 91 = 40$	$53^5 \bmod 91 = 79$
$54^5 \bmod 91 = 45$	$57^5 \bmod 91 = 57$	$65^5 \bmod 91 = 39$	$55^5 \bmod 91 = 48$
$68^5 \bmod 91 = 87$	$69^5 \bmod 91 = 62$	$50^5 \bmod 91 = 85$	$67^5 \bmod 91 = 58$
$52^5 \bmod 91 = 26$	$51^5 \bmod 91 = 25$	$52^5 \bmod 91 = 26$	$57^5 \bmod 91 = 57$
$70^5 \bmod 91 = 70$	$65^5 \bmod 91 = 39$	$50^5 \bmod 91 = 85$	$66^5 \bmod 91 = 40$
$53^5 \bmod 91 = 79$	$50^5 \bmod 91 = 85$	$70^5 \bmod 91 = 70$	$48^5 \bmod 91 = 55$
$56^5 \bmod 91 = 49$	$69^5 \bmod 91 = 62$	$70^5 \bmod 91 = 70$	$51^5 \bmod 91 = 25$
$57^5 \bmod 91 = 57$	$68^5 \bmod 91 = 87$	$48^5 \bmod 91 = 55$	$55^5 \bmod 91 = 48$
$49^5 \bmod 91 = 56$	$52^5 \bmod 91 = 26$	$48^5 \bmod 91 = 55$	$70^5 \bmod 91 = 70$

Sehingga diperoleh nilai *ciphertext* nya yaitu : "55 57 58 45 26 85 25 58 70 55 57 58 68 62 45 56 40 58 55 57 40 85 57 57 45 45 87 62 45 58 40 79 45 57 39 48 87 62 85 58 26 25 26 57 70 39 85 40 79 85 70 55 49 62 70 25 57 87 55 48 56 26 55 70"

Proses verifikasi dokumen dimulai dengan mengubah tanda tangan digital menjadi hash yang baru dengan menggunakan kunci publik. Kemudian, hasil dekripsi dibandingkan dengan message digest dokumen. Dokumen harus sah jika sama.

Contoh proses verifikasi e-dokumen adalah sebagai berikut:

1. Dengan menggunakan persamaan M = Cd mod n, ubah chiphertext dokumen elektronik menjadi pesan baru.
   C = 55 57 58 45 26 85 25 58 70 55 57 58 68 62 45 56 40 58 55 57 40 85 57 57 45 45 87 62 45 58 40 79 45 57 39 48 87 62 85 58 26 25 26 57 70 39 85 40 79 85 70 55 49 62 70 25 57 87 55 48 56 26 55 70.
   d = 29
   Hasil chipertext ditampilkan pada Tabel 13.

Tabel 13. Hasil Perubahan Chipertext

$55^{29} \bmod 91 = 48$	$57^{29} \bmod 91 = 57$	$58^{29} \bmod 91 = 67$	$45^{29} \bmod 91 = 54$
$26^{29} \bmod 91 = 54$	$85^{29} \bmod 91 = 50$	$25^{29} \bmod 91 = 51$	$58^{29} \bmod 91 = 67$
$70^{29} \bmod 91 = 70$	$55^{29} \bmod 91 = 48$	$57^{29} \bmod 91 = 57$	$58^{29} \bmod 91 = 67$
$68^{29} \bmod 91 = 57$	$62^{29} \bmod 91 = 69$	$45^{29} \bmod 91 = 54$	$56^{29} \bmod 91 = 49$
$40^{29} \bmod 91 = 66$	$58^{29} \bmod 91 = 67$	$55^{29} \bmod 91 = 48$	$57^{29} \bmod 91 = 57$
$40^{29} \bmod 91 = 66$	$85^{29} \bmod 91 = 50$	$57^{29} \bmod 91 = 57$	$57^{29} \bmod 91 = 57$
$45^{29} \bmod 91 = 54$	$45^{29} \bmod 91 = 54$	$87^{29} \bmod 91 = 68$	$62^{29} \bmod 91 = 69$
$45^{29} \bmod 91 = 54$	$58^{29} \bmod 91 = 67$	$40^{29} \bmod 91 = 66$	$79^{29} \bmod 91 = 53$
$45^{29} \bmod 91 = 54$	$57^{29} \bmod 91 = 57$	$39^{29} \bmod 91 = 65$	$48^{29} \bmod 91 = 55$
$87^{29} \bmod 91 = 68$	$62^{29} \bmod 91 = 69$	$85^{29} \bmod 91 = 50$	$58^{29} \bmod 91 = 67$
$26^{29} \bmod 91 = 52$	$25^{29} \bmod 91 = 51$	$26^{29} \bmod 91 = 52$	$57^{29} \bmod 91 = 57$
$70^{29} \bmod 91 = 70$	$39^{29} \bmod 91 = 65$	$85^{29} \bmod 91 = 50$	$40^{29} \bmod 91 = 66$
$79^{29} \bmod 91 = 53$	$85^{29} \bmod 91 = 50$	$70^{29} \bmod 91 = 70$	$55^{29} \bmod 91 = 48$

Sehingga diperoleh nilai "48 57 67 54 52 50 51 67 70 48 57 67 57 69 54 49 66 67 48 57 66 50 57 57 54 54 68 69 54 67 66 53 54 57 65 55 68 69 50 67 52 51 52 57 70 65 50 66 53 50 70 48 56 69 70 51 57 68 48 55 49 52 48 70".

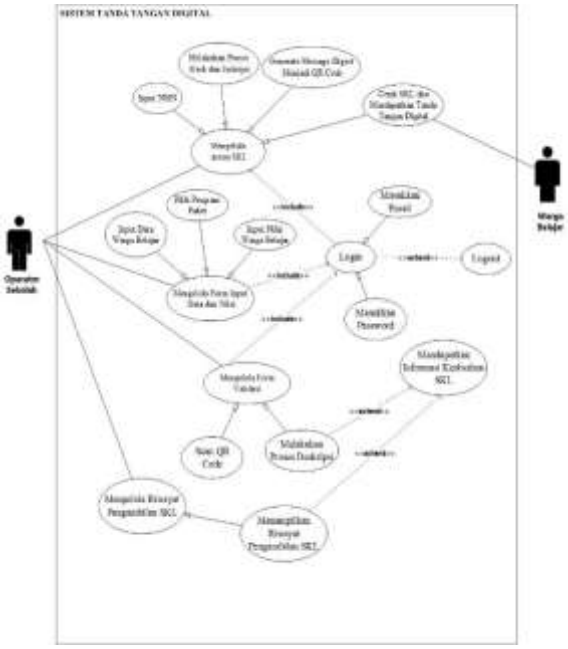
Berikut hasilnya menjadi karakter **"09C6423CF09C9E61BC09B29966DE6CB569A7DE2C4349FA2B52F08EF39D07140F".**

Bandingkan hasil dekripsi dengan *hash* e-dokumen. Karena hasil dekripsi sama dengan *hash* e-dokumen maka dokumen yang diverifikasi adalah dokumen yang sah.

3.3. Desain Sistem

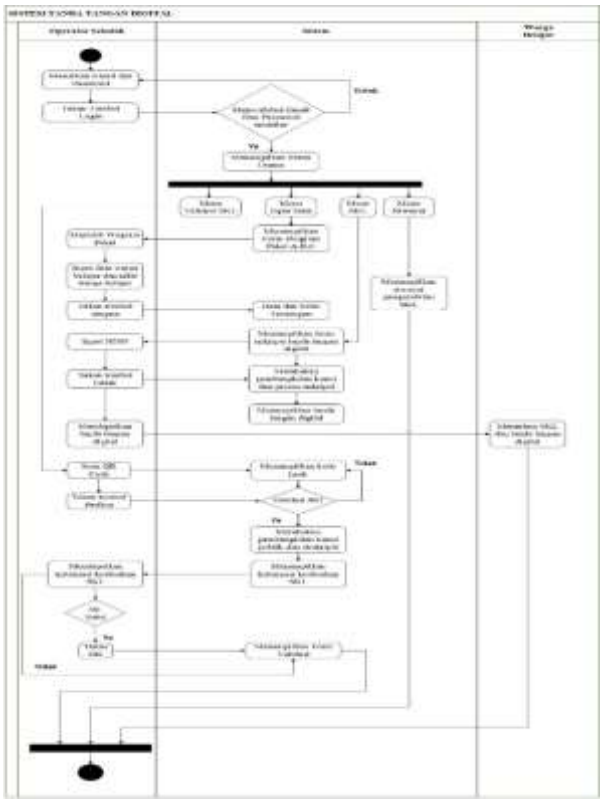
1. Skenario Login
   Aktor: Operator sekolah
   Deskripsi: Skenario ini menggambarkan tindakan form login yang muncul saat aplikasi dibuka.
2. Skenario Mengelola Menu Input Data
   Aktor: Operator sekolah
   Deskripsi: Skenario ini menggambarkan tindakan form input data yang muncul di halaman operator sekolah.
3. Skenario Mengelola Form SKL
   Aktor: Operator sekolah
   Deskripsi: Skenario ini menggambarkan tindakan form input data yang muncul di halaman operator sekolah.
4. Skenario Validasi SKL
   Aktor: Operator sekolah
   Deskripsi: Skenario ini menjelaskan aktifitas dari formulir validasi SKL yang ditampilkan di halaman operator sekolah.
5. Skenario Riwayat SKL
   Aktor: Operator sekolah
   Deskripsi: Skenario ini menjelaskan aktifitas saat menu riwayat dipilih.

Gambar 4 berikut menunjukkan use case diagram sistem yang dirancang untuk penelitian ini.



Gambar 4. Use case diagram tanda tangan digital

Gambar 5 menunjukkan activity diagram sistem yang dirancang berdasarkan use case diagram.



Gambar 5. Activity diagram tanda tangan digital

3.4. Implementasi Sistem

Berbagai bentuk sistem yang dibangun dibahas dalam implementasi sistem dalam penelitian ini. Bentuk menu utama terlihat pada Gambar 6.

Gambar 6. Form menu utama

Halaman SKL berfungsi sebagai halaman untuk proses enkripsi serta memperoleh tanda tangan digital. Di bawah ini adalah Gambar 7 tampilan halaman SKL sebagai berikut :

Gambar 7. Form SKL

Halaman validasi SKL berfungsi untuk melakukan proses dekripsi guna memverifikasi tanda tangan digital serta memperoleh informasi keabsahan dari dokumen. Di bawah ini adalah Gambar 8 tampilan halaman validasi SKL sebagai berikut :

Gambar 8. Form validasi SKL

Operator sekolah akan melakukan *submit* terhadap data yang telah ditambahkan dan akan tersimpan di dalam *database.* Untuk memberikan tanda tangan digital atau proses enkripsi serta mencetak SKL dilakukan melalui *form* SKL. Di dalam *form* SKL operator sekolah akan menambahkan nomor NISN sebagai kunci untuk melakukan proses enkripsi lalu memilih tombol cetak. Berikut ini adalah Gambar 9 tampilan *form* SKL :

Gambar 9. Form SKL

Proses enkripsi akan dilakukan oleh sistem dan sistem akan mencetak SKL sesuai dengan keterangan data yang telah ditambahkan. SKL yang akan dicetak akan berupa *file* PDF. Berikut ini Gambar 10 merupakan tampilan SKL yang telah dicetak :

Gambar 10. Tampilan cetak form SKL

Operator akan melakukan pemeriksaan keaslian setelah SKL diunduh dan diserahkan kepada siswa. Ini akan mencegah keluhan tentang tanda tangan digital. Untuk membaca QR Code, aplikasi QR Scanner diperlukan. Aplikasi ini akan membaca QR Code dan mengeluarkan teks sebagai hasil scan. Hasil scan QR code ditampilkan di Gambar 11.

Gambar 11. Hasil scan QR code

Pengecekan dilakukan dengan memasukkan digit kode verifikasi dari hasil scan ke aplikasi QR Scanner ke form validasi SKL. Proses pengecekan kode hasil scan digambarkan pada Gambar 12.

Gambar 12. Hasil validasi SKL

Gambar 13. Tampilan SKL asli

Jika digit kode verifikasi yang dimasukkan valid, maka SKL masih terjaga integritasnya dengan memunculkan informasi data SKL asli. Berikut Gambar 13 merupakan gambaran SKL asli.

Tidak ada informasi yang akan ditampilkan oleh sistem jika digit kode verifikasi yang dimasukkan tidak valid.

4. KESIMPULAN

Dalam penelitian ini, metode SHA-256 dan RSA digunakan untuk menjaga keaslian Surat Keterangan Lulus (SKL) melalui penggunaan tanda tangan digital dan kode tanggapan cepat (QR). Pengujian sistem menunjukkan bahwa penggunaan metode ini untuk mengamankan dokumen dengan QR code dan tanda tangan digital yang digunakan dalam penelitian ini dapat menjamin keaslian dokumen yang telah ditandatangani atau dibubuhi dengan QR code. Hasil pengujian menunjukkan bahwa tanda tangan digital Surat Keterangan Lulus (SKL) dapat memenuhi tiga syarat umum tanda tangan digital: otentikasi, integritas, dan keabsahan. Hasil menunjukkan bahwa algoritma SHA-256 dan RSA dapat diterapkan dengan sukses pada tanda tangan digital SKL.

DAFTAR PUSTAKA

ANJASWARI, I. A., ANDRYANA, S. & GUNARYATI, A., 2020. E-Voting Application Security Using Web-Based Cryptography RSA type. *Mantik,* 3(4), pp. 42-48.

ANSHORI, Y., DODU, A. Y. E. & P. WEDANANTA, D. M., 2019. Implementasi Algoritma Kriptografi Rivest Shamir Adleman (RSA) pada Tanda Tangan Digital. *Techno.COM ,* 18(2), pp. 110-121.

ANUGRAH, Y., ICHSAN, M. H. H. & KUSYANTI, A., 2019. Implementasi Algoritme SHA-256 Menggunakan Protokol MQTT pada Budidaya Ikan Hias. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* 3(4), pp. 4066-4073.

BABU, S. A., 2017. Modification Affine Ciphers Algorithm For Cryptography Password. *International Journal of Research In Science & Engineering,* 3(2), p. 346–351.

CAHYONO, A., 2018. Aplikasi Digital Signature Untuk Pengaman E-Document Di Pg. Pesantren Baru Menggunakan Algoritma Dsa. *Interciencia,* 489(20), p. 313–335.

FAUZIAH, N. A., RACHMAWANTO, E. H., SETIADI, D. R. I. M. & SARI , C. A., 2018. Design And Implementation Of Aes And Sha-256 Cryptography For Securing Multimedia File Over Android Chat Application. *International Seminar On*

Research Of Information Technology And Intelligent Systems (Isriti).

FIPS 180-3. 2008. Secure Hash Standard (SHS). National Institute of Standards and Technology. Information Technology Laboratory. USA.

HANDOYO, A. . E. ET AL., 2018. Teknik Penyembunyian dan Enkripsi Pesan pada Citra Digital dengan Kombinasi Metode LSB dan RSA. *Jurnal Teknologi dan Sistem Komputer,* 6(1), p. 37–45.

KHAN , A. G., BASHARAT , S. & RIAZ, M. U., 2018. Analysis of asymmetric cryptography in information security based on computational study to ensure confidentiality during information exchange. *International Journal of Scientific & Engineering Research Volume 9, Issue 10, October-2018,* 9(10), p. 992–999.

KUSUMA , E. J., SARI, C. A., RACHMAWANTO, E. H. & MOSES SETIADI, D. R. . I., 2018. A Combination of Inverted LSB, RSA, and Arnold Transformation to get Secure and Imperceptible Image Steganography. *J. ICT Res. Appl.,* 12(2), pp. 103-122.

LORIEN, A. & WELLEM, T., 2021. Implementasi Sistem Otentikasi Dokumen Berbasis Quick Response (QR) Code dan Digital Signature. *RESTI,* 5(4), pp. 663 - 671.

MULYADI , A. Y., NUGROHO, P. E. & J.P, R. R., 2018. Implementasi Algoritma AES 128 dan SHA – 256 Dalam Pengkodean pada Sebagian Frame Video CCTV MPEG-2. *JATIKOM,* 1(1), pp. 33-39.

NISHA, S. & FARIK, M., 2017. RSA Public Key Cryptography Algorithm-A Review. *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH,* 6(7), pp. 187-191.

NUGROHO, A. H., 2020. Validasi Ijazah Dengan Menggunakan Watermarking Dan Qr Code Pada Fakultas Teknik Unis Tangerang. *Jutis,* 4(2), p. 9–15.

NURAENI, F., AGUSTIN, Y. H. & MUHARAM , I. M., 2018. Implementasi Tanda Tangan Digital Menggunakan RSA dan SHA-512 Pada Proses Legalisasi Ijazah. *Konferensi Nasional Sistem Informasi,* p. 864–869.

PARAMITA, C. & SUDIBYO, U., 2021. Kriptografi Audio MP3 Menggunakan RSA dan Transposisi Kolom. *RESTI,* 5(3), pp. 483 - 488.

PRABOWO, E. C. & AFRIANTO, I., 2017. Penerapan Digital Signature Dan Kriptografi Pada Otentikasi Sertifikat Tanah Digital. *Jurnal Ilmiah Komputer dan,* 6(2), p. 83–90.

PRAMESTI, A.L., SAPUTRO, N.D.& NOVITA. M., 2019. Sistem Informasi Surat Pengganti Ijasah Sementara Berbasis Qr Code, *Science And Engineering National Seminar*, 4(Sens 4), pp. 69–73.

ROSALINA, R. & HADISUKMANA, N., 2019. Implementation of Securing Data in the Cloud using Combined Cryptography and Steganography. *Jurnal Teknik Informatika dan Sistem Informasi,* 5(3), p. 317–327.

SEMBIRING, H., MANIK , F. Y. & TENGKUZA, T., 2019. Penerapan Algoritma Secure Hash Algorithm (SHA) Keamanan Pada Citra. *MEANS,* 4(1), pp. 33-36.