Jurnal CyberTech

Vol.x. No.x, September 201x, pp. xx~xx
P-ISSN : 9800-3456
E-ISSN : 2675-9802

Implementasi Digital Signature Untuk E-Bill Pada Aplikasi Ikulapay Menggunakan Metode SHA Dan DSA

Beni Fatmadi *, Nurcahyo Budi Nugroho, Sri Murniyanti****

* Sistem Informasi, STMIK Triguna Dharma
** Sistem Informasi, STMIK Triguna Dharma

Article Info

Article history:
Received Jun 12th, 2021
Revised Aug 20th, 2021
Accepted Aug 26th, 2021

Keyword:
Tanda tangan digital
Digital signature
SHA (*Secure Hash Algorithm*)
DSA (*Digital Signature Algorithm*)
Tagihan digital

ABSTRACT

Tagihan digital saat ini merupakan sebuah kebutuhan, hal ini akan mempermudah pihak instansi untuk mendistribusikan e-bill melalui berbagai media komunikasi elektronik. Dengan adanya kemudahan tersebut maka data integrity, dan authentication merupakan hal yang sangat penting untuk menjaga keamanan dan kerahasian data tagihan oleh pihak yang tidak bertanggung jawab demi kepentingan pribadinya sendiri dengan cara memanipulasi atau mengubah data tagihan. Adapun cara untuk menangani permasalahan tersebut dengan mengubah pesan menjadi sebuah kode. Teknologi sistem informasi yang dapat menjaga data integrity dan authentication dalam keadaan tetap aman ialah digital signature. SHA (Secure Hash Algorithm) dan DSA (Digital Signature Algorithm) merupakan salah satu jenis algoritma signature. Pemanfaatan dari algoritma SHA ialah dapat melindungi pesan pada saat distribusi dengan menghitung nilai hash pada data e-bill, sehingga setiap e-bill memiliki nilai hash yang berbeda beda dan tidak akan menghasilkan nilai hash yang sama, sedangkan DSA dapat memberikan jaminan othentikasi pengirim dan juga penerima.

Corresponding Author:
Nama : Beni Fatmadi
Sistem Informasi
STMIK Triguna Dharma
Email: benipatmadi@gmail.com

1. PENDAHULUAN

Perkembangan sistem informasi yang setiap tahunnya semakin maju dan berkembang pada akhirnya akan membuat sebagian besar instansi harus mengikuti perubahan teknologi informasi, termasuk perubahan kegiatan transaksi yang tadinya bersifat manual menjadi serba *digital*. Teknologi informasi terus berkembang dari tahun ke tahun, pada akhirnya, pembayaran menggunakan uang *cash* dalam berbagai jenis pembayaran atau transaksi kini sudah semakin berkurang, dikarenakan adanya teknologi informasi yang disebut *e-money* atau uang elektronik. Transfer antar bank dinilai lebih efektif dan lebih banyak dipilih konsumen terutama saat melakukan transaksi secara *online*, karena tidak lagi harus saling bertemu.

Tagihan dalam bentuk kertas saat ini menjadi media pembayaran, namun potensial dalam personalisasi sangat terbatas. Hal tersebut juga sedang berlangsung dibanyaknya perusaahaan dan perguruan tinggi. Adanya perkembangan teknologi informasi harus diimbangi dengan kebutuhan suatu perusahaan dan perguruan tinggi. Begitu juga dengan sistem informasi tagihan, tagihan yang terkomputerisasi akan

mengurangi waktu proses pembuatan tagihan tersebut [1]. CV. Deacas merupakan sebuah perusahaan teknologi informasi yang memproduksi aplikasi-aplikasi untuk kalangan pendidikan. Tujuan utama aplikasi-aplikasi yang dibuat CV. Deacas ialah meningkatkan nilai akreditas kampus dan sekolah. Salah satu produk aplikasi CV. Deacas yaitu aplikasi ikulapay, aplikasi ikulapay adalah aplikasi keuangan sekolah, untuk memberikan kemudahan bagi sekolah dalam mengelola keuangan, mencatat dan melihat laporan keuangan. Pencatatan keuangan bertujuan supaya terlaksananya manajemen keuangan yang baik, karena pencatatan keuangan yang tidak baik akan berakibat fatal dalam pengambilan keputusan kedepan. Aplikasi ikulapay belum memiliki keamanan yang kuat dalam mengamankan *e-bill* (elektronik tagihan) yang dimana isinya tersebut merupakan tagihan uang sekolah. Hal ini mengakibatkan adanya permasalahan yaitu perubahan isi tagihan uang sekolah oleh pihak yang tidak bertanggung jawab. Hal tersebut tentunya berdampak *negatif* terhadap sekolah atau kampus yang menggunakan aplikasi ikulapay. Dari kondisi tersebut, disarankan menggunakan *digital signature*.

Penelitian ini akan memanfaatkan *digital signature* untuk *e-bill*. *Digital signature* dapat memecahkan sebuah permasalahan diantaranya adalah melakukan sistem pengamanan *file* [2], dan juga dapat melakukan autentikasi pada dokumen [3]. Dari jurnal tersebut dapat diketahui bahwasannya *digital signature* dapat menyelesaikan masalah yang bersifat memanipulasi atau mengubah isi dari sebuah dokumen. Dalam *digital signature* dapat diadopsi dengan beberapa metode diantaranya adalah metode SHA dan DSA. Untuk menegaskan bahwasanya metode SHA dan DSA dapat diterapkan ke dalam permasalahan yang bersifat mengamankan sebuah dokumen diambil beberapa referensi. Dalam beberapa referensi metode SHA dan DSA dapat diterapkan ke dalam beberapa persoalan yang bersifat mengamankan dokumen diantaranya: implementasi *secure e-health system* yang dimana untuk menikatkan pelayanan medis dan mengurangi biaya serta menjaga privasi tetap aman [4]. Selain itu dalam referensi lain metode SHA dan DSA juga dapat menyelesaikan masalah pemalsuan sebuah dokumen dengan melakukan pemeriksaan intergritas [5] .

Dari referensi-referensi tersebut terlihat metode SHA dan DSA dapat dinyatakan sebagai solusi untuk menyelesaikan permasalahan *e-bill* yang berisi tagihan uang sekolah. Kedepannya hal baru yang ada di aplikasi ikulapay adalah sistem terpadu yang berbasis web yang mengadopsi metode SHA dan DSA yang mampu menyelesaikan masalah khususnya dalam menjaga privasi dan keamanan tagihan *digital*. Penelitian ini juga diharapkan dapat memberikan solusi kepada CV. Deacas untuk dapat bertumbuh menjadi perusahaan yang lebih besar dan lebih baik kedepannya. Berdasarkan kondisi tersebut maka diangkatlah judul penelitian yaitu: **"Implementasi Digital Signature Untuk E-Bill Pada Aplikasi Ikulapay Menggunakan Metode Sha Dan Dsa Studi Kasus Di CV. Deacas".**

2. METODE PENELITIAN

Model yang digunakan adalah model *waterfall*. Model *waterfall* merupakan suatu model dalam pengembangan *software* dimana pengerjaannya dilakukan secara sistematis dan berurutan.

Terdapat 5 tahap dalam pengembangan sistem model *waterfall* sebagai berikut:

1. Analisis Kebutuhan

   Pengembang sistem diperlukan suatu komunikasi yang bertujuan untuk memahami *software* yang dibutuhkan. Informasi ini diperoleh dengan wawancara dan observasi.
2. Desain Sistem

   Pada proses desain, berfokus pada struktur data, arstitektur perangakt lunak, representasi dan detail algoritma *procedural*.
3. Penerapan

   Pada tahap ini terjadi proses perancangan desain ke bentuk yang dapat dimengerti oleh mesin, dengan menggunakan kode-kode bahasa pemograman. Kode program yang dihasilkan masih berupa modul-modul kecil yang nantinya akan digabungkan pada tahap berikutnya.

4. Integrasi Dan Pengujian
   Ditahap ini dilakukan penggabungan modul-modul yang sudah dibuat dan dilakukan pengujian, dilakukan untuk mengetahui apakah *software* yang telah dibuat sesuai dengan desainnya dan fungsinya pada *software* terdapat kesalahan atau tidak.
5. Operasi Dan Pemeliharaan
   Ini merupakan tahap terakhir dalam model waterfall. *Software* yang sudah jadi dijalankan serta dilakukan pemeliharaan atau perawatan. Pemeliharaan termasuk dalam memperbaiki kesalahan-kesalahan yang tidak ditemukan pada langkah-langkah sebelumnya [6].

3. ANALISA DAN HASIL

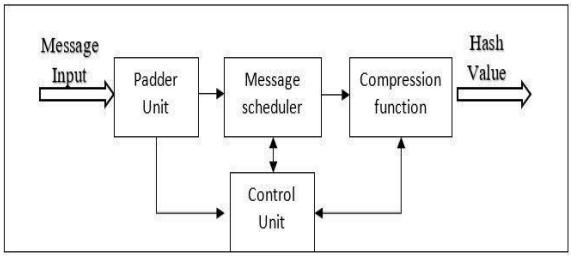
Adapun dalam analisa ini menggunakan kombinasi dari dua metode *digital signature* diantaranya yaitu SHA-256 dan DSA. Pemanfaatan algoritma SHA-256 akan melindungi pesan saat proses distribusi yaitu dengan menghitung nilai *hash* pada dokumen *e-bill* dan algoritma DSA dapat memberikan jaminan *othentikasi* pengirim maupun penerima dokumen.

3.1. SHA-256 (*Secure Hash Algorithm*)

National Security Agency (NSA) mendesain fungsi *hash* satu arah yang merupakan fungsi dari SHA lalu, dipublikasikan oleh *National Institute of Standards and Technology* (NIST) sebagai *Federal Information Processing Standard* (FIPS) disebut sebagai SHA-0 pada tahun 1993, SHA-1 dipublikasikan setelah dua tahun kemudian yang merupakan generasi selanjutnya dari perbaikan algoritma SHA-0. Pada tahun 2002 ada empat variasi, yaitu SHA-224, SHA-256, SHA-384, dan SHA-512, keempat variasinya disebut sebagai SHA-2. Dapat dinyatakan aman karena SHA secara komputasi *message digest* yang dihasilkan dari isi pesan tidak dapat ditemukan, dan tidak dapat menghasilkan *message digest* yang sama dari dua pesan yang berbeda.

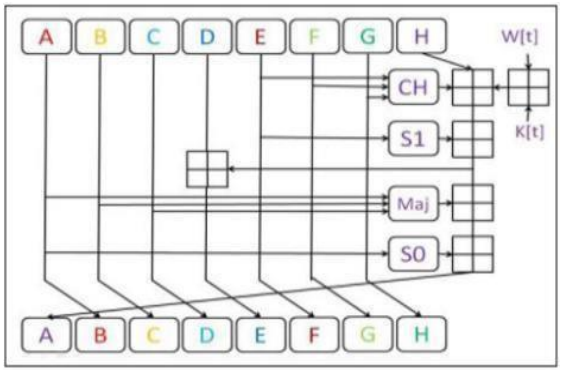
SHA-256 mengubah pesan masukan kedalam *messgage digest* 256 *bit* merupakan cara kerja algoritma SHA-256. Pesan masuk yang panjangnya lebih pendek dari 2^{64} *bit* berdasarkan *secure hash signature* standart, wajib dioperasikan oleh 512 *bit* dalam kelompok dan menjadi sebuah *message digest* 256 *bit* [7].

Berikut merupakan arsitektur sederhana dari SHA-256 sebagai berikut [8]:



Gambar 1. Arsitektur Sederhana SHA-256

Berikut merupakan jalur komputasi SHA-246:



Gambar 2. Jalur Komputasi SHA-256

3.2. DSA (*Digital Signature Algorithm*)

Algoritma sidik *digital* yang disebut *Digital Signature Algorithm* (DSA) diumumkan oleh NIST (*The National Institute of Standard and Technology*) pada bulan agustus 1991. DSA dijadikan sebagai bakuan (*standard*) dari *Digital Signature Standard* (DSS). DSA termasuk kedalam sistem kriptografi, sebagai halnya pada algoritma kriptografi kunci publik, DSA menggunakan dua buah kunci yaitu kunci publik dan kunci privat. Pembentukan sidik *digital* menggunakan kunci privat pengirim, sedangkan verifikasi menggunakan kunci publik pengirim. DSA menggunakan fungsi *hash* SHA untuk mengubah pesan menjadi *message digest* yang berukuran 160 *bit*. DSA mempunyai dua fungsi yaitu sebagai pembentukan tanda tangan (*signature generation*) dan pemeriksaan keabsahan tanda tangan (*signature verification*) [9].

3.3. Penerapan Dengan Metode

Berikut meruapakan data yang digunakan sebagai sampel untuk penelitian:

Tabel 1. Data Awal

No. Bill	Nominal
327	Rp. 150. 000

Data awal yang diperoleh selanjutnya akan dilakukan tahapan *hasing* dengan penerapan SHA-256 yaitu:

1. Pada data sampel *no.bill* terdapat sebuah pesan teks "327", pesan tersebut diubah ke dalam bentuk *binner*, 3=00000011, 2=00000010. 7=00000111. Maka pesan, M = 00000011 00000010 00000111, panjang pesan, 1 = 24 *bit*.
2. *Message Padding*

   $l + 1 + k \equiv 448 \; mod \; 512$

   $24 + 1 + k \equiv 448 \; mod \; 512$

   $k \equiv 448 - 25 \; mod \; 512$

   $k \equiv 423$

   Karena k=423 maka banyak *bit* 0 yang akan ditambahkan adalah sebanyak 423 *bit*, setelah itu ditambahkan jumlah panjang pesan pada akhir pesan yang di *padding* sebanyak 8 *bit* dengan nilai 1 = 00011000.

   Berikut ini merupakan hasil dari *padding* pesan:

Tabel 2. *Message Padding*

00000011	00000010	00000111	10000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000

3. *Parsing*

   Tahapan selanjutnya adalah membagi setiap blok 512 *bit* menjadi 16 buah *word* 32 *bit* sebagi berikut:

Tabel 3. *Parsing*

$M_0^{(0)}$	0000	0011	0000	0010	0000	0111	1000	0000
$M_0^{(1)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(2)}$	0000	0000	0000	0000	0000	0000	0000	0000

$M_0^{(3)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(4)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(5)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(6)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(7)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(8)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(9)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(10)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(11)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(12)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(13)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(14)}$	0000	0000	0000	0000	0000	0000	0000	0000
$M_0^{(15)}$	0000	0000	0000	0000	0000	0000	0001	1000

4. *Message Schdule*

Selanjutnya menyiapkan *message schdule* dari 16 buah *word* hasil *parsing* tadi menggunakan rumus berikut:

$$Wt = \begin{cases} M_t^{(t)} & 0 \le t \le 15 \\ \sigma_1^{(256)}(W_{i-2}) + W_{i-7} + \sigma_0^{(256)}(W_{i-15}) + W_{i-16}, & 16 \le t \le 63 \end{cases} \quad (1)$$

Gambar 3. Rumus *Message Schdule*

Dimana:

$\sigma_1^{(256)}(W_{i-2}) = ((W_{i-2})ROTR\ 17) \square ((W_{i-2})ROTR\ 19) \square ((W_{i-2})SHR10)$ (2)

$\sigma_0^{(256)}(W_{i-15}) = ((W_{i-15})ROTR\ 7) \square ((W_{i-15})ROTR\ 18) \square ((W_{i-15})SHR3)$ (3)

Keterangan:

Wt = Blok pesan yang baru Mt = Blok pesan yang lama

Wi-2 = Blok pesan dari W ke i-2 Wi-15 = Blok pesan dari W ke i-15

ROTR = *Rotate Right* SHR = *Shift Right*

□ = *Operator* XOR

Berikut ini merupakan hasil dari *message schdule*:

Tabel 4. *Message Schdule*

W0	03020780	W16	00000000	W32	00000000	W48	00000000
W1	00000000	W17	00000000	W33	00000000	W49	00000000
W2	00000000	W18	00000000	W34	00000000	W50	00000000
W3	00000000	W19	00000000	W35	00000000	W51	00000000
W4	00000000	W20	00000000	W36	00000000	W52	00000000
W5	00000000	W21	00000000	W37	00000000	W53	00000000
W6	00000000	W22	00000000	W38	00000000	W54	00000000
W7	00000000	W23	00000000	W39	00000000	W55	00000000
W8	00000000	W24	00000000	W40	00000000	W56	00000000
W9	00000000	W25	00000000	W41	00000000	W57	00000000
W10	00000000	W26	00000000	W42	00000000	W58	00000000
W11	00000000	W27	00000000	W43	00000000	W59	00000000
W12	00000000	W28	00000000	W44	00000000	W60	00000000
W13	00000000	W29	00000000	W45	00000000	W61	00000000

W14	00000000	W30	00000000	W46	00000000	W62	00000000
W15	00000018	W31	00000000	W47	00000000	W63	00000000

5. Inisialisasi *Variabel* Dan Konstanta

Masing-masing dari 64 *word* yang diberi label W0, W1,..W63 tadi kemudian diproses dengan algoritma fungsi *hash* SHA-256. Dalam proses tersebut, adalah membuat 8 variabel yang diberikan nilai untuk nilai awal dari $H_0^{(0)} - H_7^{(0)}$ di awal masing-masing fungsi *hash*.

Dibawah ini merupakan nilai-nilai awal *variabel* tersebut adalah sebagai berikut :

Tabel 5. Nilai Awal SHA-256

$A = H_{0(0)}$	6a09e667
$B = H_{1(0)}$	bb67ae85
$C = H_{2(0)}$	3c6ef372
$D = H_{3(0)}$	a54ff53a
$E = H_{4(0)}$	510e527f
$F = H_{5(0)}$	9b05688c
$G = H_{6(0)}$	1f83d9ab
$H = H_{7(0)}$	5be0cd19

Berikut ini merupakan nilai konstanta dari SHA-256 :

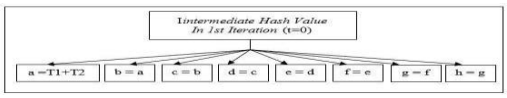
Tabel 6. Nilai Konstanta

428A2F98	71374491	B5C0FBCF	E9B5DBA5	3956C25B	59F111F1	923F82A4	AB1C5ED5
D807AA98	12835B01	243185BE	550C7DC3	72BE5D74	80DEB1FE	9BDC06A7	C19BF174
E49B69C1	EFBE4786	0FC19DC6	240CA1CC	2DE92C6F	4A7484AA	5CB0A9DC	76F988DA
983E5152	A831C66D	B00327C8	BF597FC7	C6E00BF3	D5A79147	06CA6351	14292967
27B70A85	2E1B2138	4D2C6DFC	53380D13	650A7354	766A0ABB	81C2C92E	92722C85
A2BFE8A1	A81A664B	C24B8B70	C76C51A3	D192E819	D6990624	F40E3585	106AA070
19A4C116	1E376C08	2748774C	34B0BCB5	391C0CB3	4ED8AA4A	5B9CCA4F	682E6FF3
748F82EE	78A5636F	84C87814	8CC70208	90BEFFFA	A4506CEB	BEF9A3F7	C67178F2

6. *Hash Computation*

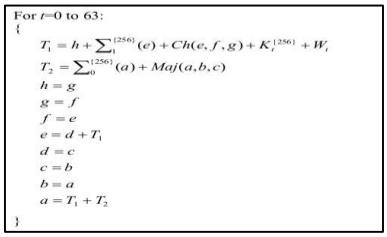
Algoritma ini melakukan perhitungan sebanyak 64 kali putaran untuk setiap perhitungan blok. Delapan variabel yang diberi A, B, C, ..., H tadi nilainya terus terus berganti selama perputaran sebanyak 64 kali putaran.

Dibawah ini merupakan bentuk perhitungan dari *hash computation* sebagai berikut:



Gambar 4. *Intermediate Hash Value In 1st Iteration* Rumus

Berikut ini merupakan rumus perhitungan putaran:



Gambar 5. Rumus Perhitungan Putaran

Rumus untuk menghasilkan nilai T1 adalah:

$T1 = h + S1(e) + ch(e,f,g) + kt + wt$

Dimana:

h = Nilai h pada iterasi sebelumnya

S1(e) = $(e \ggg 6) \oplus (e \ggg 11) \oplus (e \ggg 25)$

Ch (e, f,g) = $(e \,\&f) \oplus ((\neg\, e)\& g)$

Kt = Nilai konstanta

Wt = Nilai W pada Message Schadule

Maka nilai T1 adalah sebagai berikut :

h=5BE0CD19

(e) = (510E527F $\ggg$6) $\oplus$ (510E527$F\ggg$11) $\oplus$ (510E527$F\ggg$11) = 3587272B

$ch(e,f,g)$ = (510E527F & 9B05688C $\oplus$ ((¬ 510E527F) & 1F83D9AB)) = 1F85C98C

$T1$ = 5BE0CD19 + 3587272B + 1F85C98C + 428$A2F$98 + 03020780 = F679F4E7

Rumus untuk menghasilkan nilai T2 adalah sebagai berikut :

$T2 = S0(a) + Maj(a,b,c)$

Dimana:

S0(a) = $(a \ggg 2) \oplus (a \ggg 13) \oplus (a \ggg 22)$

Maj (a,b,c) = $(a \,\&\, b) \oplus (a \,\&\, c) \oplus (b \,\&\, c)$

Maka nilai T2 adalah sebagai berikut :

$S0(a)$ = (6A09E667$\ggg$2) $\oplus$ (6A09E667$\ggg$13) $\oplus$ (6A09E667$\ggg$22)

$S0(a) = CE20B47E$

(a,b,c) = (6A09E667 & BB67AE85) $\oplus$ (6A09E667 & 3C6EF372) $\oplus$(BB67AE85 & 3C6EF372)

(a,b,c) = 3A6FE667

$T2 = CE20B47E$+3$A6FE$667 = 08909AE5

Karena nilai T1 dan T2 telah didapatkan, maka nilai a adalah sebagai berikut:

$a = T1 + T2$

a = F679F4E7 + 08909AE5 = FF0A8FCC

Tabel 7. Nilai Putaran T=0 Dan T=63

	a	b	c	d	e	f	g	h
init	6a09e667	bb67ae85	3cbef372	a54ff53a	510e527f	9b05688c	1f83d9ab	5be0c019
t=0	FF0A8FCC	6a09e667	bb67ae85	3cbef372	a54ff53a	510e527f	9b05688c	1f83d9ab
t=63	98C2BD19	7FEEA059	D55E0681	5E799BD1	E8B6DF87	837523AF	57AD9011	E15B46F2

Setelah didapat ke 64 putaran dari *hash computation*, kemudian pada tahap ini dilakukan proses penjumlahan hasil putaran yang ke-64 dengan *initial hash value*

$H_0^{(0)}$ = 98C2BD19 + 6a09e667 = 02cca380

$H_1^{(0)}$ = 7FEEA059 + bb67ae85 = 3b564ede

$H_2^{(0)}$ = D55E0681 + 3c6ef372 = 11ccf9f3

$H_3^{(0)}$ = 5E799BD1 + a54ff53a = 03c9910b

$H_4^{(0)}$ = E8B6DF87 + 510e527f = 39c53206

$H_5^{(0)}$ = 837523AF + 9b05688c = 1e7a8c3b

$H_6^{(0)}$ = 57AD9011 + 1f83d9ab = 773169bc

$H_7^{(0)}$ = E15B46F2 + 5be0cd19 = 3d3c140b

7. Penggabungan H0-H7

Selanjutnya hasil akhir SHA-256 didapat dari penggabungan delapan variabel yang tadi sudah dikomputasi = 02cca380 3b564ede11ccf9f303c9910b39c532061e7a8c3b773169bc3d3c140b

8. Pembangkit Sepasang Kunci
   a. p dan q adalah bilangan prima, dimana p dengan panjang L *bit* atau $512 \leq L \leq 1024$ dengan kelipatan 64, $(p-1)\ Mod\ q = 0$.
      p = 59419 dan q = 3301, $(59419-1)\ Mod\ 3301 = 0$
   b. Menghitung parameter $g = h^{(p-1)/q}\ Mod\ p$ yang bersifat publik, dimana $1 < h < p-1$ dan $h^{(p-1)/q}\ Mod\ p > 1$.
      h = 3501
      g = $3501^{(59419-1)/3301}\ Mod\ 59419 = 21628$
   c. Menentukan nilai sembarang untuk parameter *x* atau kunci privat yang merupakan bilangan bulat, dimana $x < q$.
      x = 2201 (Sebagai kunci privat)
   d. Menghitung nilai pada kunci publik $y = g^x\ Mod\ p$.
      $y = 21628^{2201}\ Mod\ 59419$
      y = 34431 (Sebagai kunci publik)
9. Pemberian *Signature* (proses tanda-tangan) pada dokumen
   Input : Pesan (M) dan kunci privat (x)
   Output : Pesan (M) dan tanda-tangan (r, s)
   a. Ubah nilai *hash* dari 347 kedalam bentuk bilangan bulat, yaitu sebagai berikut : H(m) = 02cca3803b564ede11ccf9f303c9910b39c532061e7a8c3b773169bc3d3c140b
      H(m)
      =1266190940157953728708675617671981068445245955135603539862874207519917216779
   b. Menentukan bilangan acak $k < q$
      k = 1011
   c. Menghitung r dan s tanda-tanda dari pesan, yaitu sebagai berikut :
      r = $(g^K\ Mod\ p)\ Mod\ q$
      = $(2128^{1011}\ Mod\ 59419)\ Mod\ 3301$
      r = 547
      s = $(k^{-1}(\mathrm{H(m)} + \mathrm{x} * \mathrm{r}))\ Mod\ q$. k^{-1} merupakan *invers k Mod q*
      k^{-1} = 1515
      Maka
      s=(1515(1266190940157953728708675617671981068445245955135603539862874207519917216779 + 2021 *547) *Mod* 3301
      s = 222
   d. Pesan (M) dapat dikirim beserta tanda-tangan r dan , digital signature merupakan gabungan pesan *r* dan *s* = 547222.
10. Verifikasi *Digital Signature*
    Teorema 1 : (Pembuktian v = r)
    Jika $M' = M, r' = r$, dan $s' = s$ pada verifikasi tandatangan, maka *v = r'. '*
    a. s^{-1}= *Invers s Mod q*
       = *Invers* 222 *Mod* 3301 = 342
    b. *w* =s^{-1} *Mod q*
       = 342 *Mod* 3301 = 342
    c. *u1* = (H(m) * w) *Mod q*
       =(1266190940157953728708675617671981068445245955135603539862874207519917216779 * 342) *Mod* 3301
       = 1191

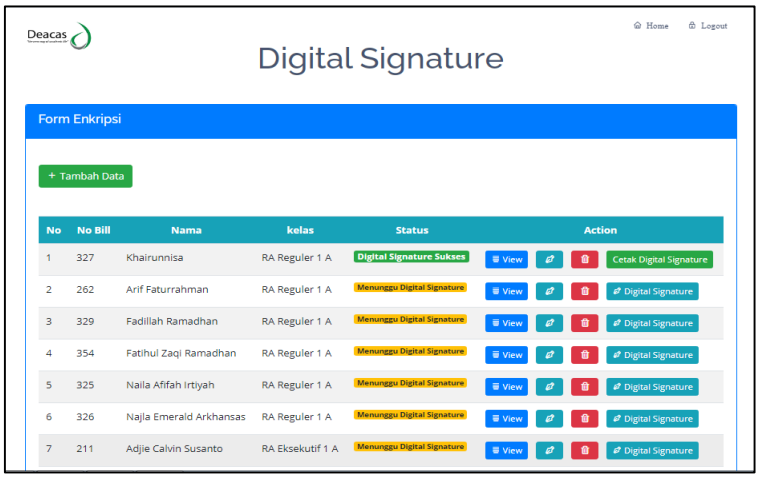
d. $u2 = (r * w) \ Mod \ q$
$= (547 * 342) \ Mod \ 3301 = 2218$

e. $\mathbf{v} = ((g^{u1} * y^{u2}) \ Mod \ p) \ Mod \ q)$
$= ((21628^{1191} * 34431^{2218}) \ Mod \ 59419) \ Mod \ 3301$
$V = 547$

f. Karena $v = r$, maka tanda tangan dinyatakan asli

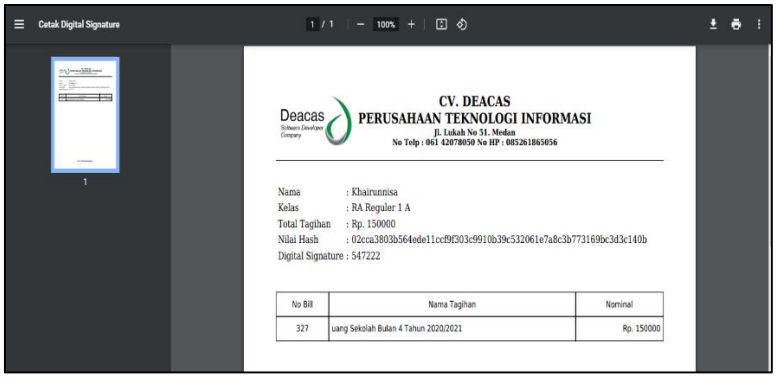
3.4 Pengujian

Adapun memulai pengujian sistem dilakukan pada *form digital signature* untuk memasukan data yang dimana data tersebut di *input* ke dalam *form* tambah data tagihan yang isinya berupa, nomor *bill*, nama, kelas, nama tagihan, total tagihan. Untuk mendapatkan *digital signature* tinggal klik tombol *digital signature* dibagian data tagihan yang ingin di *digital signature*, sistem secara otomatis akan melakukan proses enkripsi dan melakukan pembentukan kunci beserta *digital signature*. setelah itu maka statusnya berubah menjadi *digital signature* sukses, untuk memastikan data tagihan tersebut sudah di *digital signature* klik tombol *view*.



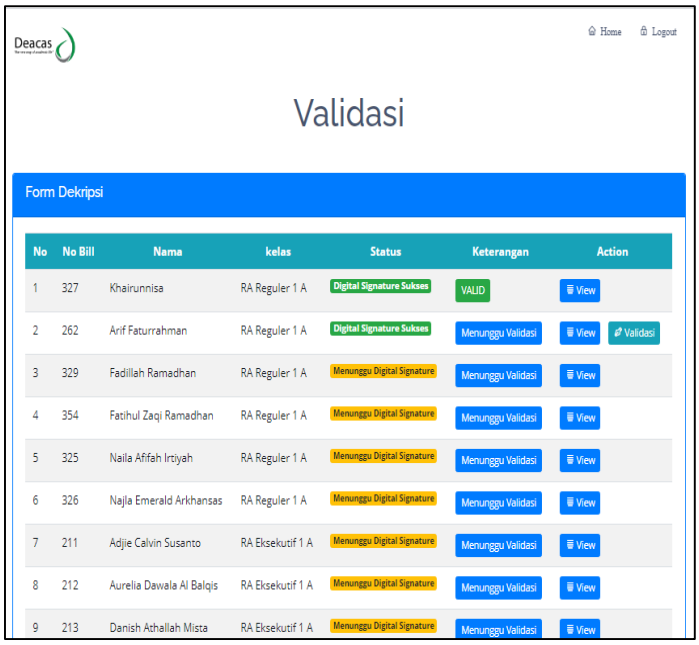
Gambar 6 *Form Digital Signature*

Dimana hasil *digital signature* tersebut merupakan hasil dari metode SHA dan DSA. Hasil *digital signature* dapat diberikan kepada siswa dengan cara cetak *digital signature*.



Gambar 7. Tampilan Cetak *Digital Signature* Dan *E-Bill*

Lanjut pada *form* validasi, yang dimana *form* tersebut untuk melakukan validasi dengan tujuan untuk mengetahui keabsahan dari dokumen tersebut, dengan cara klik tombol validasi, tombol validasi akan tampil apabila data tagihan sudah di *digital signature*. Yang dimana *digital signature* tersebut didapat dari hasil proses enkripsi. Setelah mengklik tombol validasi, maka akan melakukan proses dekripsi, dan akan menampilakan keterangan berupa valid atau tidak valid, untuk melihat informasi keabsahan dengan cara klik tombol *view*, maka akan tampil informasi berupa VALID, dan jika dokumen tersebut sudah dipalsukan atau diubah maka akan tampil informasi keabsahan berupa TIDAK VALID.



Gambar 8. *Form* Validasi

4. KESIMPULAN

Adapun kesimpulan dari penelitian ini yaitu sebagai berikut:

1. Sistem dapat melakukan proses pembangkitan sepasang kunci secara tetap atau tidak berubah dan sistem juga dapat melakuan proses pembangkitan sepasang kunci secara *random* atau nilainya berubah-ubah.
2. Sistem dapat melakukan proses pembuatan *digital signature* atau tanda tangan pada dokumen, yang hasil dari *digital signature* diberikan kepada siswa.
3. Sistem dapat melakukan verifikasi, sehingga mampu mendeteksi keabsahan atau keaslian dari dokumen *e-bill.*
4. Sistem dapat diimplementasikan dengan menerapkan algoritma SHA (*Secure Hash Algorithm*) dan DSA (*Digital Signature Algorithm*).
5. Dapat menambah pengetahuan dalam merancang suatu sistem dengan menggunakan metode SHA dan DSA, dan juga dapat memberikan informasi bagi pembaca penelitian ini dalam permasalahan *digital signature*.

UCAPAN TERIMA KASIH

Penulis mengucapkan terimakasih kepada program studi S1 Sistem Informasi STMIK Triguna Dharma yang telah memberikan dukungan dalam penyelesaian tulisan ini.

REFERENSI

[1] A. Delvina, “Penggunaan Tanda Tangan Elektronik dalam Pengajuan Pembiayaan berdasarkan Prinsip Syariah,” *J. Akunt. Bisnis dan Ekon.*, vol. 05, no. 01, pp. 1305–1318, 2019.

[2] M. Ihwani, “Model Keamanan Informasi Berbasis Digital Signature Dengan Algoritma Rsa,” *CESSJournal Comput. Eng. Syst. Sci.*, vol. 1, no. 1, pp. 15–20, 2016.

[3] Sugiyatno and P. D. Atika, “Digital Signature Dengan Algoritma Sha-1 Dan Rsa Sebagai Autentikasi,” *J. Cendikia*, vol. 16, no. 2, pp. 74–83, 2018.

[4] S. Adleman, D. A. N. Digital, and S. Algorithm, “Implementasi secure e-health system berbasis rivest shamir

adleman dan," vol. 001, pp. 95–106, 2015.
[5] P. P. Sarjana and U. Diponegoro, "E-Dokumen Dengan Metode Hybrid : Biometrik Tandatangan Dan Dsa (Digital Signature Algorithm)," pp. 1–150, 2011.
[6] M. Susilo, "Rancang Bangun Website Toko Online Menggunakan Metode Waterfall," *InfoTekJar (Jurnal Nas. Inform. dan Teknol. Jaringan)*, vol. 2, no. 2, pp. 98–105, 2018, doi: 10.30743/infotekjar.v2i2.171.
[7] D. V. S. Y. Sakti, N. Agani, and M. Hardjianto, "Pengamanan Sistem Menggunakan One Time Password Dengan Pembangkit Password Hash SHA-256 dan Pseudo Random Number Generator (PRNG) Linear Congruential Generator (LCG) di Perangkat Berbasis Android," *J. Budi Luhur*, vol. 13, no. 1, pp. 1–10, 2016.
[8] Z. Panjaitan, E. F. Ginting, and Y. Yusnidah, "Modifikasi SHA-256 dengan Algoritma Hill Cipher untuk Pengamanan Fungsi Hash dari Upaya Decode Hash," *J. SAINTIKOM (Jurnal Sains Manaj. Inform. dan Komputer)*, vol. 19, no. 1, pp. 53–61, 2020.
[9] M. B. Y. Diki Arisandi, Sukri, "PEMERIKSAAN INTEGRITAS DOKUMEN DENGAN DIGITAL SIGNATURE ALGORITHM," *J. Ilm.*, vol. 16, no. 1, pp. 73–82, 2016.

BIBLIOGRAFI PENULIS

	**Beni Fatmadi** lahir pada tahun 1998 di Kedai Durian, saat ini sedang menempuh studi Sistem Informasi di STMIK Triguna Dharma. Sejak tahun 2020 hingga saat ini bekerja sebagai programmer di CV. Deacas. Menyelesaikan Program Kreativitas Mahasiswa (PKM-P) bersama rekan lainya pada tahun 2020.
	**Nurcahyo Budi Nugroho, S.Kom., M.Kom.** Merupakan dosen tetap STMIK Triguna Dharma bidang Sistem Informasi, Beliau aktif megampu mata kuliah dibidang Program Website, Dekstop dan Mobile, aktif dalam mengembangkan Mutu Mahasiswa dalam bidang Pemrograman dan Teknik Algoritma Pemrograman.
	**Sri Murniyanti, S.S., M.M.** Merupakan dosen tetap STMIK Triguna Dharma bidang Sistem Informasi, Beliau aktif mengampu mata kuliah dibidang Prinsip Manajemen Bisini, Teknik Pemasaran dan Technopreneur.