Available *online* at : http://bit.ly/InfoTekJar

InfoTekJar : Jurnal Nasional Informatika dan Teknologi Jaringan

ISSN (Print) 2540-7597 | ISSN (Online) 2540-7600

Digital Signature untuk Menjaga Keaslian Data dengan Algoritma MD5 dan Algoritma RSA

Budi K. Hutasuhut [*], *Syahril Efendi, dan Zakarias Situmorang*

Program Studi S2 Teknik Informatika Fakultas Ilmu Komputer Dan Teknologi Informasi Universitas Sumatera Utara

KEYWORDS

Digital Signature, Message digest, RSA, MD5.

CORRESPONDENCE

E-mail: *hutasuhutbk@gmail.com*

A B S T R A C T

Penelitian ini bertujuan untuk menjaga keaslian data untuk memberikan jaminan kepada si penerima bahwa data tersebut bebas dari modifikasi yang dilakukan oleh pihak lain, dan jika terjadi suatu modifikasi terhadap data tersebut, maka si penerima akan mengetahui bahwa data tersebut tidak lagi terjaga keasliannya. Untuk menjaga keaslian data digunakan teknik digital signature dengan menggunakan algoritma MD5 sebagai algoritma fungsi hash untuk menghasilkan message digest, dan algoritma RSA sebagai algoritma kunci publik, dengan kombinasi kedua algoritma tersebut akan dihasilkan digital signature dari setiap data yang akan dijaga keasliannya.

PENDAHULUAN

Kerahasiaan dari data yang ada pada data dalam proses pengiriman sangat dibutuhkan untuk mencegah informasi diketahui oleh pihak-pihak yang tidak berhak. Selain masalah kerahasiaan, masalah keaslian data juga sangat diperlukan untuk dilakukan pengamanan, hal ini dikarenakan jika penerima menerima data paslu (data yang tidak sesuai dengan data yang dikirim oleh pengirim) akan perbedaan arti sehingga menimbulkan kekacauan dan kerugian di kedua belah pihak.

Digital signature merupakan salah satu teknik yang dapat digunakan untuk menjaga keaslian data, sehingga penerima mendapatkan jaminan untuk mengetahui bahwa data yang diterima merupakan data asli atau data palsu. Teknik ini dapat mencegah terjadinya penggunaan data palsu oleh penerima data. Setiap data yang diterima memiliki tanda tangan yang selalu berbeda dengan data lainnya, sehingga sedikit saja modifikasi yang dilakukan akan menyebabkan tanda tangan yang berubah sangat drastis.[1]

ALGORITMA MD5

Algoritma MD5 dikembangkan oleh Ronald L. Rivest pada tahun 1991 yang merupakan perbaikan dari algoritma MD4 yang diketahui tidak lagi aman. Algoritma MD5 merupakan salah satu algoritma fungsi hash yang dapat menerima *input* dengan panjang sembarang dan akan menghasilkan *output* berupa *message digest* dengan panjang 128 bit. [2]

MD5 dikembangkan dari algoritma MD, MD2, MD3 and MD4. MD5 telah digunakan dibanyak bidang untuk mengamankan keaslian data. Secara garis besar, proses pembuatan *message digest* pada MD5 meliputi tahapan sebagai berikut: [3]

1. Penambahan bit-bit pengganjal

   Pesan ditambah dengan sejumlah bit pengganjal sedemikian sehingga panjang pesan (dalam satuan bit) kongruen dengan 448 (mod 512).

2. Penambahan nilai panjang semula

   Pesan yang telah diberi bit-bit pengganjal selanjutnya ditambah lagi dengan 64 bit yang menyatakan panjang pesan semula

3. Inisialisasi penyanggal (buffer) MD

   MD5 membutuhkan 4 buah penyangga (buffer) yang masing-masing panjangnya 32 bit. Keempat penyangga

DOI: https://doi.org/10.30743/infotekjar.v3i2.1019

ini diberi nama A, B, C, dan D. Setiap penyangga diinisialisasi dengan nilai-nilai (dalam notasi HEX) sebagai berikut:

A = 01234567

B = 89ABCDEF

C = FEDCBA98

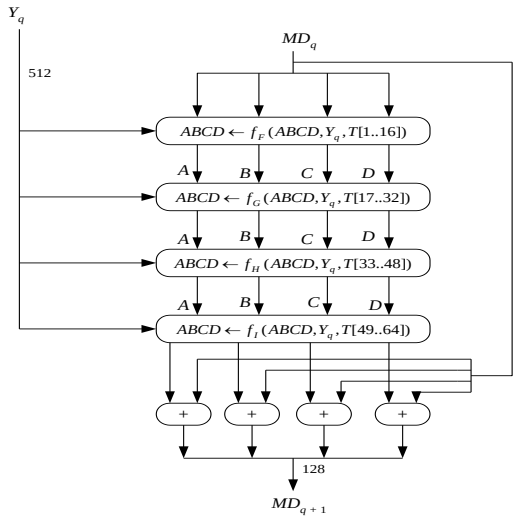
D = 76543210

4. Pengolahan pesan pada blok 512-bit

Pesan dibagi menjadi L buah blok yang masing-masing panjangnya 512 bit (Y_0 sampai Y_{L-1}).

Setiap blok 512-bit diproses bersama dengan penyangga MD menjadi keluaran 128-bit, dan ini disebut proses H_{MD5}

Proses tersebut dapat dilihat pada gambar di bawah: [3]



Gambar 1. Proses H_{MD5}

Proses H_{MD5} terdiri dari 4 buah putaran, dan masing-masing putaran melakukan operasi dasar *MD5* sebanyak 16 kali dan setiap operasi dasar memakai sebuah elemen *T*. Sehingga setiap putaran memakai 16 elemen Tabel *T*.

Fungsi yang digunakan pada setiap putaran :

F(x,y,z) = (x & y) | ((~x) & z)
G(x,y,z) = (x & z) | (y & (~z))
H(x,y,z) = x ^ y ^ z
I(x,y,z) = y ^ (x | (~z))

ALGORITMA RSA

Algoritma RSA diciptakan oleh Ron Rivest, Adi Shamir, dan Leonard Adleman, sesuai dengan nama penemunya, pada tahun 1970-an. Rancangan ini bergantung pada kerumitan dalam memfaktorisasi bilangan bulat (integer) yang berbeda dari penyelesaian algoritma diskrit. RSA hanya menggunakan operasi pemangkatan. RSA termasuk algoritma asimetri, yang berarti memiliki dua kunci, yaitu kunci publik dan kunci privat. [4]

Algoritma RSA terdiri dari 3 proses, yaitu : [5]

1. Pembangkit Kunci

Proses pembentukan atau pembangkitan kunci algoritma RSA yaitu :

1) Pilih dua bilangan prima acak ukuran besar, p dan q.

2) Hitung modulus sistem

$$n = p * q$$

3) Cari Totient Φ (n)

$$\Phi(n) = (p-1)(q-1)$$

4) Pilih kunci enkripsi e secara acak

Dimana $1 < e < \Phi(n)$, gcd $(e, \Phi(n)) = 1$

5) Selesaikan rumus berikut untuk menentukan kunci dekripsi d.

$d \equiv e^{-1} \pmod{\Phi(n)}$

dimana ekivalen dengan :

$e * d \equiv 1 \pmod{\Phi(n)}$, dimana $0 \leq d \leq n$

sehingga dihasilkan :

a. *Private key* = (d, n)

bersifat sangat rahasia, dan hanya penerima pesan yang boleh mengetahuinya.

b. *Public key* = (e, n)

Bersifat tidak rahasia, dan boleh disebarkan dengan bebas.

2. Enkripsi

Secara umum proses enkripsi dengan RSA dilakukan dengan rumus sebagai berikut :

$$C_i = P_i^e \bmod n$$

Tetapi pada penelitian ini, enkripsi dilakukan dengan menggunakan *private key*, bukan dengan *public key*, kebalikan dari proses enkripsi secara umum dengan RSA. Hal ini bertujuan agar hanya pengirim pesan yang dapat membangkitkan *digital signature* dari data yang dikirim. Jika saat proses pengiriman *intruder* melakukan *modification* atau *fabrication* terhadap data, maka intruder tidak akan dapat membangkitkan *digital signature* yang sesuai dikarenakan *intruder* tidak memiliki *private key* yang sesuai.

Sehingga proses enkripsi pada penelitian ini menjadi :

$$C_i = P_i^d \bmod n$$

3. Deskripsi

Secara umum proses dekripsi dengan RSA dilakukan dengan rumus sebagai berikut:

$$P_i = C_i^d \bmod n$$

Tetapi pada penelitian ini, proses deskripsi menggunakan *public key*, bukan *private key*. Hal ini dimaksudkan agar setiap penerima dapat menguji

keaslian file. Sehingga proses deskripsi pada penelitian ini menjadi :

$$P_i = C_i^e \bmod n$$

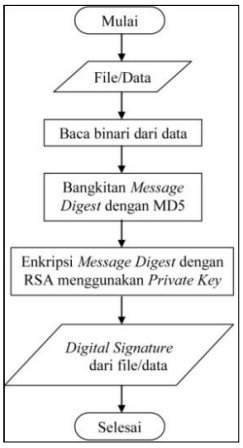
METODE PENELITIAN

Metode penelitian pada penelitian ini digambarkan pada diagram dibawah ini :

Gambar 2. Metode Penelitian

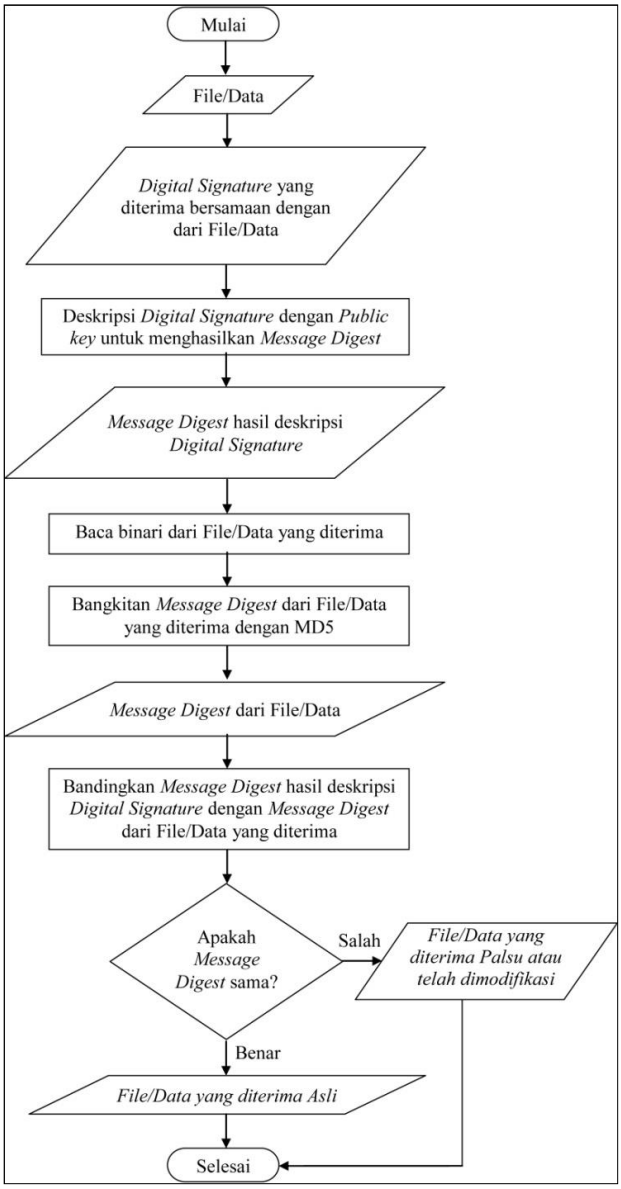
DESAIN PROSES

Proses yang dilakukan dalam membentuk *digital signature* pada data dapat dilihat pada *flowchart* dibawah:



Gambar 3. Proses Digital signature File/Data

Sedangkan proses dalam melakukan pengujian keaslian data dapat dilihat pada flowchart dibawah ini :

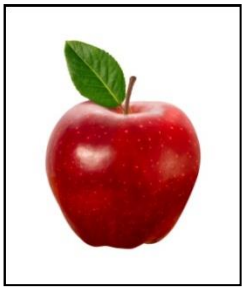


Gambar 4. Proses Pengujian keaslian File/Data

HASIL DAN PENGUJIAN

Desain proses diimplementasikan dengan bahasa pemrograman PHP yang dijalankan dengan *browser* Google Chrome. *Web server* yang adigunakan adalah XAMPP 1.7.3.

Pengujian dilakukan dengan beberapa jenis file. Pada pengujian pertama dilakukan pengujian pada file citra, dengan nama file Apple.jpg.



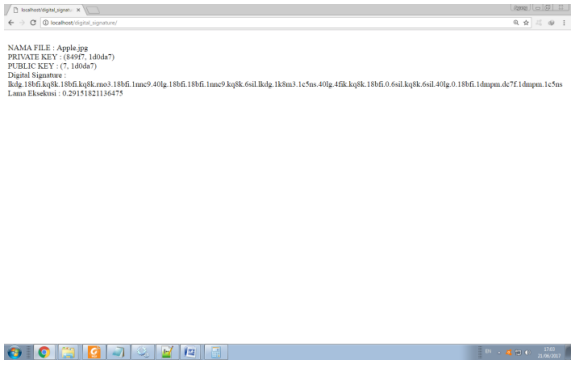
Gambar 5. Apple.jpg

DOI: https://doi.org/10.30743/infotekjar.v3i2.1019

Program akan secara otomatis membangkitkan *private key* dan *public key* dalam bentuk bilangan heksadesimal. Adapun *digital signature* yang dihasilkan akan dalam bentuk bilangan berbasis 32. Yaitu suatu bilangan yang terdiri dari kombinasi dari simbol dibawah ini : 0,1,2,3,4,5,6,7,8,9,a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v

Nilai desimal yang didapat pada *digital signature* akan dikonversikan secara otomatis oleh program kedalam sistem bilangan berbasis 32.

Pada percobaan pertama didapatkan :



Gambar 6. Hasil Pembangkitan *Digital signature* File Apple.jpg

NAMA FILE : Apple.jpg
PRIVATE KEY : (849f7, 1d0da7)
PUBLIC KEY : (7, 1d0da7)
Digital signature :
lkdg.18bfi.kq8k.18bfi.kq8k.rno3.18bfi.1nnc9.40lg.18bfi.18bfi.
1nnc9.kq8k.6sil.lkdg.1k8m3.1c5ns.40lg.4fik.kq8k.18bfi.0.6sil
.kq8k.6sil.40lg.0.18bfi.1dmpm.dc7f.1dmpm.1c5ns
Lama Eksekusi : 0.29151821136475

File Apple.jpg dapat dikirim dengan aman dengan menyertakan *public key*, dan *digital signature* yang telah dibangkitkan. *Private key* bersifat sangat rahasia.

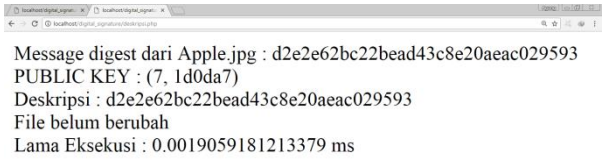
Penerima akan menerima file Apple.jpg, kemudian penerima akan memeriksa apakah file Apple.jpg mengalami perubahan atau modifikasi selama proses pengiriman. Penerima dapat membangkitkan *message digest* dari file Apple.jpg yang diterima, lalu mendeskripsi *digital signature* yang disertakan dengan menggunakan *public key* yang juga telah disertakan bersama saat pengiriman file Apple.jpg. Hasil deskripsi akan dibandingkan dengan *message digest* yang berhasil dibangkitkan, jika hasil deskripsi dari *digital signature* dengan *message digest* adalah sama, maka dapat dipastikan file Apple.jpg belum mengalami perubahan atau modifikasi, jika ternyata perbandingannya berbeda, maka dapat dipastikan file Apple.jpg telah mengalami perubahan atau modifikasi.

Percobaan kedua bertujuan untuk memeriksa file Apple.jpg yang diterima apakah telah termodifikasi atau tidak.

Hasil percobaan kedua dengan *digital signature* :

lkdg.18bfi.kq8k.18bfi.kq8k.rno3.18bfi.1nnc9.40lg.18bfi.18bfi.1
nnc9.kq8k.6sil.lkdg.1k8m3.1c5ns.40lg.4fik.kq8k.18bfi.0.6sil.kq
8k.6sil.40lg.0.18bfi.1dmpm.dc7f.1dmpm.1c5ns

adalah sebagai berikut :



Gambar 7. Hasil Pengujian Keaslian File Apple.jpg

Message digest dari Apple.jpg :
d2e2e62bc22bead43c8e20aeac029593
PUBLIC KEY : (7, 1d0da7)
Deskripsi :
d2e2e62bc22bead43c8e20aeac029593
File belum berubah

Lama Eksekusi : 0.0019059181213379 ms

Hasil percobaan kedua menunjukkan bahwa file Apple.jpg tidak mengalami modifikasi, sehingga file Apple.jpg yang diterima masih terjaga integritas data nya.

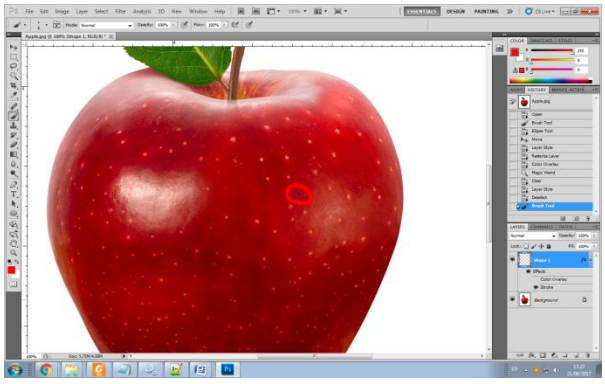
Pada gambar 7. hasil percobaan kedua dapat dilihat, *message digest* yang dihasilkan dari file Apple.jpg yang diterima adalah :

d2e2e62bc22bead43c8e20aeac029593

Sedangkan hasil deskripsi *digital signature* dari file Apple.jpg yang asli memberikan hasil yang sama dengan *message digest* dari file Apple.jpg yang diterima, yaitu :

d2e2e62bc22bead43c8e20aeac029593

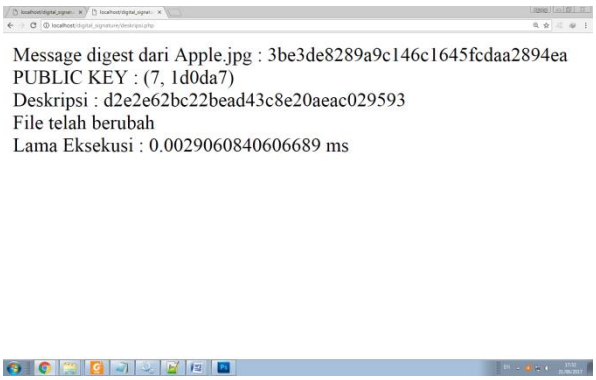
Pada percobaan ketiga, file Apple.jpg akan dilakukan sedikit modifikasi dengan photoshop dengan menambahkan sebuah titik biru yang sangat kecil.



Gambar 8. Modifikasi pada File Apple.jpg

Setelah file apple.jpg dimodifikasi, maka akan dilakukan percobaan keempat untuk mengetahui apakah jika terjadi sedikit saja modifikasi terhadap file akan membatalkan *digital signature* atau tanda tangan digital-nya.

Berikut hasi percobaan keempat :



Gambar 9. Hasil Percobaan keempat

Hasil percobaan keempat menunjukkan bahwa file telah berubah, *digital signature* atau tanda tangan digital yang diberikan telah terbatalkan oleh file Apple.jpg yang diterima, sehingga file Apple.jpg yang telah diterima telah mengalami modifikasi sehingga integritas data dari file Apple.jpg telah diserang. Pada gambar 9, hasil percobaan keempat dapat dilihat, *message digest* yang dihasilkan dari file Apple.jpg yang diterima adalah :

3be3de8289a9c146c1645fcdaa2894ea

Sedangkan hasil deskripsi *digital signature* dari file Apple.jpg yang asli adalah :

d2e2e62bc22bead43c8e20aeac029593

Dapat dilihat, sebuah perubahan dengan menambahkan sebuah titik biru kecil pada file Apple.jpg akan memberikan perubahan yang sangat besar terhadap *message digest* yang dihasilkan, sehingga penerima akan dapat memastikan bahwa file Apple.jpg telah mengalami perubahan, dan tidak layak digunakan. Untuk pengujian terhadap file dengan berbagai format lain dapat dilihat pada tabel dibawah ini :

Tabel 1. Hasil percobaan terhadap beberapa format file.

No	Format File	Size dan Key	Key	*Digital signature*	Waktu Proses (ms)
1	.docx	78.9 KB	*private key* : (ab007, 257483) *public key* : (7, 257483)	74ds.1h19m.9bfp.2aj23.1i711.1apct.s4g4.1.1r76v.s2bc.74ds.s4g4.2aj23.28pj5.28pj5.0.mi54.74ds.28pj5.28pj5.1h19m.28pj5.1ovdl.1ovdl.1apct.1ovdl.1i711.28pj5.28pj5.74ds.1r76v.1h19m	0.60454392433167
2	.xlsx	34.9 KB	*private key* : (1a9123, 1d442d)	0.1m39l.1jek1.vcda.e8lt.9h6f.1hcrc.1jek1.1hcrc.1.17kg7.1jek1.1hcrc.d7ob.722v.vcda.1.1qaau.1m39l.1	1.22123599052 43
			public key : (b, 1d442d)	7kg7.e8lt.722v.9h6f.hv2q.hv2q.1gce6.e8lt.0.vcda.vcda.1hcrc.e8lt	
3	.txt	9.03 KB	*private key* : (5530d, 1aa997) *public key* : (5, 1aa997)	p2si.14r7t.0.lrh6.ge99.42ka.0.42ka.1cts9.0.1a6k1.14r7t.ge99.14s9p.1cadj.1cts9.p2si.42ka.h2gf.1a6k1.5do.5do.lrh6.h2gf.5do.1cts9.0.1cadj.14s9p.1cadj.1.5do	0.19954609870911
4	.rar	2.68 MB	*private key* : (167aa1, 258355) *public key* : (5, 258355)	1hku7.1hku7.10pi1.16cet.21ui8.qmi3.t09q.79fu.1fch4.qmi3.21ui8.16cet.sqkf.1fch4.1gn77.1h98b.1qdbe.t09q.1gn77.qmi3.qmi3.1hku7.1qdbe.qmi3.21ui8.79fu.1gn77.1.79fu.dpnv.sqkf.21ui8	1.34321117 40112
5	.mp3	4.15 MB	*private key* : (1d88ad, 314735) *public key* : (5, 314735)	1mm15.of0.1mmpu.sk4.1mmpu.10bdg.10bdg.84up.1vfmu.1qlnc.2mg22.2mg22.lpll.1qlnc.1qlnc.2dujb.0.41qq.1vfmu.0.sk4.1vfmu.1qlnc.1mmpu.1mmpu.1qlnc.lpll.1vfmu.10bdg.84up.lpll.84up	1.75189900 39825
6	.pdf	132 KB	*private key* : (173737, 1b2081) *public key* : (7, 1b2081)	b120.1.9553.1.irtp.1.g7n7.2rqc.0.1lo6s.9553.0.0.2rqc.2fih.apr5.apr5.0.2rqc.g7n7.71ol.15018.71ol.apr5.irtp.b120.9553.2fih.apr5.13h63.irtp.g7n7	0.84202504 15802
7	.mp4	3.54 MB	*private key* : (d5e1b, 14161f) *public key* : (3, 14161f)	thd7.heb2.2ssq.2ssq.13562.2.2ssq.1.10pus.10d0u.0.2.mcg3.heb2.1.2.81hd.10pus.81hd.13562.thd7.8qu9.10d0u.1.du5f.1.2ssq.9uug.2ssq.du5f.1.9uug	0.59433388 710022

ANALISA KEAMANAN

Tanpa *private key*, maka tidak dapat sembarang pihak yang dapat membangkitkan *digital signature* dari suatu data. Jika penyerang menggunakan sembarang *private key*, maka pasangan *public key* yang digunakan oleh penerima data tidak akan cocok. Karena berbeda *private key* akan memberikan pasangan *public key* yang berbeda pula.

DOI: https://doi.org/10.30743/infotekjar.v3i2.1019

Algoritma MD5 dapat menerima masukan dengan panjang sembarang, dan akan menghasilkan *message digest* dengan panjang yang tetap, yaitu sepanjang 128-bit. Dengan keluaran sepanjang 128-bit, kemungkinan untuk mencari pasangan data dengan *message digest* yang sama adalah 2^{128}. Jika penyerang mampu melakukan operasi percobaan sebanyak 1 triliun operasi per detik. Maka dibutuhkan waktu $2^{128}/10^{12}$ detik atau 10.790.283 Triliun Tahun untuk mencari pasangan data yang sama.

KESIMPULAN

Dari percobaan yang telah dilakukan, *digital signature* dapat memberi keamanan terhadap keaslian data, sehingga penerima terhindar dari penggunaan data-data palsu yang telah dimodifikasi oleh penyerang. Sedikit saja perubahan yang terjadi terhadap data, akan mengubah secara signifikan terhadap *digital signature* yang ada, sehingga secara otomatis akan membatalkan *digital signature* yang diberikan sebelumnya kepada data.

Digital signature dapat dibangkitkan dengan kombinasi algoritma MD5 dan RSA. Kombinasi kedua algoritma tersebut dapat memberikan peningkatan keamanan pada keaslian data. Algoritma MD5 memiliki keluaran 128-bit, sehingga memiliki 2^{128} kombinasi yang mungkin. Algoritma RSA mencegah agar tidak sembarang orang dapat membangkitkan *digital signature* dari data yang ada, hanya pengirim yang memiliki *private key* yang dapat membangkitkan *digital signature*. Sehingga kombinsai dari kedua algoritma ini akan memberikan tingkat keamanan yang tinggi

REFERENSI

[1] E. Noroozi,. 2013. Secure *Digital signature* Schemes Based on Hash Functions. International Journal of Innovative Technology and Exploring Engineering (IJITEE)

[2] Neyole M. Jacob,. 2016. Vulnerability of data security using MD5 function in php database design. International Journal of Science and Engineering (IJSE)

[3] Priyanka Walia,. 2014. Implementation of New Modified MD5-512 bit Algorithm for Cryptography. International Journal of Innovative Research in Advanced Engineering (IJIRAE)

[4] Ravindra Babu Kallam,. 2011. An Enhanced RSA *Public key* Cryptographic Algorithm. International Journal of Advanced Research in Computer Science (IJARCS)

[5] Sombir Singh,. 2013. A Performance Analysis of DES and RSA Cryptography. International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)

[6] Marc Schneider,.-. A Robust Content Based Digital Signature For Image Authentication. Columbia University

[7] Shivendra Singh,. 2015. Survey on Techniques Developed using Digital Signature: Public key Cryptography. International Journal of Computer Applications

[8] D. Shiva Rama Krishna,. 2015. Providing Security to Confidential Information Using Digital signature.