PENERAPAN DIGITAL SIGNATURE DAN KRIPTOGRAFI PADA OTENTIKASI SERTIFIKAT TANAH DIGITAL

Egi Cahyo Prabowo[1], Irawan Afrianto[2]

[1,2] Teknik Informatika – Universitas Komputer Indonesia
Jl. Dipatiukur 112-116 Bandung
E-mail : ghie21@ymail.com[1], irawan.afrianto@email.unikom.ac.id[2]

ABSTRAK

Digital signature merupakan suatu teknologi digital yang dapat disipkan pada suatu dokumen untuk menjaga otentifikasinya. Peneltian ini tertujuan menerapkan *digital signature* untuk menguji keutuhan dan otentikasi dokumen sertifikat tanah digital, serta dapat mendeteksi perubahan dokumen sertifikat tanah digital dari hasil manipulasi oleh orang yang tidak berhak. Salah satu cara untuk melakukan *digital signature* pada dokumen sertifikat tanah digital yaitu dengan menggunakan fungsi *hash*. Algoritma *hash* yang digunakan dalam penelitian adalah *Secure Hash Algorithm*-256 (SHA-256), sedangkan algoritma kunci publik yang digunakan adalah algoritma Rivest-Shamir-Adleman (RSA). Berdasarkan hasil pengujian yang telah dilakukan, dapat disimpulkan bahwa mengimplementasikan *digital signature* menggunakan fungsi *hash* algoritma SHA-256 dan algoritma RSA dapat memberikan layanan keamanan otentikasi dokumen pada sertifikat tanah digital sehingga dapat mencegah terjadinya pemalsuan dan manipulasi dokumen oleh orang yang tidak berhak.

Kata kunci : *Digital Signature*, Fungsi *Hash*, *Secure Hash Algorithm*-256 (SHA-256), Algoritma Rivest-Shamir-Adleman (RSA), Sertifikat Tanah.

1. PENDAHULUAN

Sertfikat tanah merupakan suatu dokumen yang penting bagi masyrakat. Didalamnya berikisi informasi mengenai luas tanah dan kepemilikannya. Hal inilah yang kemudian sering memunculkan manipualsi terhadap sertifikat tersebut, baik secara data maupun pencetakannya.

Guna meningkatkan pengamanan dan menjaga keaslian dari sertifkat tanah tersebut, akan dikembangkan suatu mekanisme pengambngan sertfikat tanah digital dengan menyertakan digital signature didalamnya untuk menjaga keutuhan dan otentikasi dari sertfikat tanah tersebut. *Digital signature* dapat berfungsi untuk menguji keutuhan dan otentikasi suatu dokumen digital, serta dapat mendeteksi perubahan dokumen dari hasil manipulasi [1]. Salah satu cara untuk melakukan *digital signature* pada dokumen yaitu dengan menggunakan fungsi *hash*, dari fungsi *hash* tersebut nantinya akan menghasilkan *message digest*.

Untuk menghasilkan *message digest* yang aman diperlukan sebuah fungsi *hash* yang mempunyai panjang nilai *hash* lebih banyak, dan SHA-256 merupakan algoritma *hash* yang aman dengan panjang 256 bit [3], sedangkan algoritma kunci publik yang biasa digunakan adalah algoritma kunci publik Rabin, algoritma ElGamal, dan algoritma RSA. Diantara ketiga algoritma tersebut, algoritma RSA merupakan algoritma yang tidak terlalu sederhana tetapi juga tidak terlalu rumit, sehingga algoritma RSA merupakan algoritma yang paling pas jika hendak mengimplementasikan algoritma kriptografi kunci publik [4].

Hal inilah yang melatarbelakangi penelitian ini untuk menghasilkan suatu aplikasi pengembangan sertifikat tanah digital dengan menerapakan *digital signature* didalamnya.

2. ISI PENELITIAN

2.1 Keamanan dan Kerahasiaan Data

Keamanan atau *security* adalah mekanisme dan teknik untuk melindungi sesuatu yang dapat berupa data atau informasi di dalam sistem. Pada dasarnya *security* adalah sistem yang digunakan untuk melindungi sistem dalam suatu jaringan keamanan agar tetap terjaga. Masalah keamanan dan kerahasiaan data merupakan salah satu aspek penting dari suatu sistem informasi. Dalam hal ini, sangat terkait dengan betapa pentingnya informasi tersebut diterima oleh orang yang berkepentingan. Informasi akan menjadi tidak *valid* lagi jika informasi tersebut di ketahui atau dibajak oleh orang yang tidak berhak [5].

Informasi atau data saat ini sudah menjadi sesuatu yang sangat penting bagi sebuah organisasi, perguruan tinggi, lembaga pemerintahan maupun individual, termasuk kemampuan dalam mengakses dan menyediakan informasi secara cepat serta akurat. Karena pentingnya sebuah informasi, seringkali informasi yang diinginkan hanya dapat diakses oleh orang tertentu misalnya pihak penerima yang diinginkan, dan jika informasi ini sampai

diterima oleh pihak yang tidak diinginkan akan berdampak kerugian pada pihak pengirim [6].

2.2 Aspek Keamanan Informasi

Keamanan informasi adalah bagaimana kita dapat mencegah penipuan, atau paling tidak mendeteksi adanya penipuan di sebuah sistem yang berbasis informasi, untuk melindungi informasi dari pengaksesan, penggunaan, penyebaran, perusakan, perubahan, dan penghancuran tanpa otorisasi yang sah. Untuk itu diperlukanlah sebuah pendekatan dalam melakukan pengamanan pada informasi, seperti melakukan enkripsi, *steganografi*, *cipher*, *digital signature* dan *hashing* terhadap informasi tersebut.

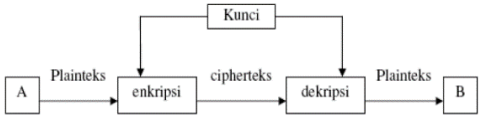
Aspek keamanan komputer adalah bentuk pertimbangan yang menyatakan sebuah komputer bisa dinyatakan aman. Aspek-aspek keamanan di dalam kriptografi adalah sebagai berikut: [7]

1. *Confidentiality* (kerahasiaan) yaitu merupakan usaha untuk menjaga informasi dari orang yang tidak berhak mengakses.
2. *Authentication* (otentikasi) yaitu agar penerima informasi dapat memastikan keaslian pesan tersebut datang dari orang yang dimintai informasi
3. *Non-repudiation* (nir penyangkalan) yaitu merupakan hal yang yang bersangkutan dengan si pengirim, si pengirim tidak dapat mengelak bahwa dia lah yang mengirim informasi tersebut.
4. *Integrity* (data Integritas) yaitu keaslian pesan yang dikirim melalui sebuah jaringan dan dapat di pastikan bahwa informasi yang dikirim tidak di modifikasi oleh orang yang tidak berhak dalam perjalanan informasi tersebut.

2.3 Kriptografi

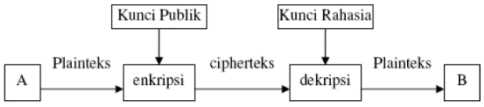
Kriptografi adalah ilmu yang mempelajari bagaimana supaya pesan atau dokumen kita aman, tidak bisa dibaca oleh pihak yang tidak berhak. Terdapat dua jenis algoritma kriptografi berdasar jenis kuncinya yaitu: [8]

1. Algoritma simetri disebut juga sebagai algoritma konvensional adalah algoritma yang menggunakan kunci enkripsi yang sama dengan kunci dekripsinya. Algoritma simetrik sering juga disebut sebagai algoritma kunci rahasia, algoritma kunci tunggal, atau algoritma satu kunci. Pada Gambar 1 dinyatakan bahwa kriptografi simetris pada saat di enkripsi dan di dekripsi menggunakan kunci yang sama.



Gambar 1. Proses Kriptografi Simetri

2. Algoritma asimetri didesain sedemikian sehingga kunci yang digunakan untuk enkripsi berbeda dari kunci yang digunakan untuk dekripsi, pada Gambar 2 terlihat kunci untuk enkripsi tidak sama dengan kunci untuk dekripsi. Kunci untuk enkripsi tidak rahasia, sehingga dinamakan juga kunci publik (*public key*), sedangkan kunci untuk dekripsi rahasia, sehingga dinamakan kunci privat (*privat key*). Pengirim pesan mengenkripsi pesan dengan menggunakan kunci publik si penerima pesan, hanya penerima pesan yang dapat mendekripsi pesan menjadi plainteks semula dengan menggunakan kunci privatnya. Contoh algoritma kriptografi asimetris diantaranya adalah RSA (Rivest-Shamir-Adleman), Rabin, dan ElGamal.



Gambar 2. Proses Kriptografi Asimetri

2.4 *Digital Signature*

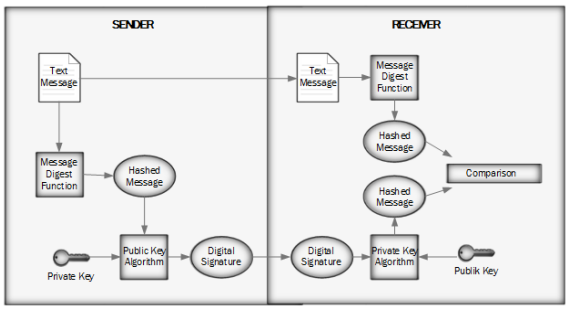
Salah satu konsep pada kriptografi modern adalah *digital signature*. Cara kerja dan kegunaan *digital signature* mirip dengan tanda tangan dalam versi nyata, yaitu untuk memberikan kepastian keaslian dan persetujuan dokumen oleh penanda tangan. Dalam *digital signature*, "tanda tangan" adalah dalam bentuk digital yang digunakan untuk mensahkan sebuah dokumen digital.

Prinsip yang digunakan dalam tanda tangan digital ini adalah dokumen yang dikirimkan harus ditandatangani oleh pengirim dan tanda tangan bisa diperiksa oleh penerima untuk memastikan keaslian dokumen yang dikirimkan. Fungsinya adalah untuk melakukan validasi terhadap data yang dikirim. Tanda tangan digital menggunakan algoritma yang disebut dengan istilah *hashing algorithm*. Fungsi tersebut akan menghasilkan sebuah kombinasi karakter yang yang unik yang disebut *message digest*. dengan cara ini pengirim bertanggungjawab terhadap isi dokumen dan dapat di cek keaslian dokumen oleh penerima.

Keunikannya adalah jika di tengah perjalanan data mengalami modifikasi, penghapusan maupun di sadap diam-diam oleh *hacker* walaupun hanya 1 karakter saja, maka *message digest* yang berada pada si penerima akan berbeda dengan yang dikirimkan pada awalnya. Keunikan lainnya adalah *message digest* tersebut tidak bisa dikembalikan lagi ke dalam bentuk awal seperti sebelum disentuh dengan fungsi algoritma, sehingga disebut sebagai *one-way hash* [2].

Fungsi utama dari tanda tangan digital pada pada aspek keamanan kriptografi adalah *non-repudiation* atau anti penyangkalan dimana apabila dokumen

valid maka pengirim tidak bisa menyangkal bahwa keberadaan dokumen benar dikirim oleh pengirim yang bersangkutan.



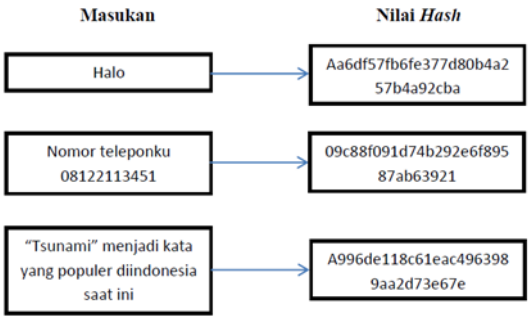
Gambar 3. Skema Digital Signature

Cara kerja *digital signature* seperti yang terlihat pada Gambar 3 adalah sebagai berikut:

1. Sender melakukan proses *hashing algorithm* untuk menghasilkan *message digest* dari sebuah pesan yang terdapat dalam sebuah dokumen yang akan dikirim.
2. Setelah dilakukan *hashing*, Sender melakukan *sign* terhadap *message digest* dengan menggunakan kunci publik yang digunakan untuk membentuk *digital signature*.
3. Kemudian Sender mengirimkan *digital signature* bersama dokumen tersebut kepada Receiver.
4. Receiver menerima pesan yang dikirimkan oleh Sender.
5. Setelah itu Receiver mengverifikasi pesan yang dikirimkan oleh Sender. Pada proses verifikasi tersebut pesan di *hashing* terlebih dahulu sehingga menghasilkan *message digest* dan *digital signature* akan di *unsign* menggunakan kunci private. Jika *message digest*-nya sama, maka pesan ini adalah asli dan pesan berasal dari pengirim yang sebenarnya. Bila pesan telah diubah oleh pihak luar, maka *message digest* juga ikut berubah.

2.5 Fungsi Hash

Fungsi *hash* adalah suatu fungsi yang menerima masukan berupa string yang panjangnya sembarang dan mengonversi masukan tersebut menjadi string yang mempunyai panjang tetap (*fixed*) dan umumnya menjadi lebih kecil dari panjang semula. Keluaran dari fungsi *hash* disebut juga nilai *hash* atau pesan ringkas (*message digest*). Fungsi *hash* merupakan fungsi satu arah (*one way function*) yang dapat menghasilkan ciri (*signature*) dari data. Perubahan satu bit saja akan mengubah keluaran *hash* secara drastis. Fungsi *hash* biasanya digunakan untuk menjamin integritas dan *digital signature*.



Gambar 4. Penggunaan Fungsi Hash

Pada Gambar 4, memperlihatkan fungsi *hash* yang mengubah suatu string dengan panjang berapapun menjadi sebuah *message digest* yang memiliki panjang tetap. Fungsi *hash* pada dasarnya bekerja satu arah, berarti pesan asli atau pesan semula akan diubah menjadi sebuah *message digest*, namun *message digest* yang dihasilkan tidak dapat dikembalikan menjadi pesan asli atau pesan semula kembali. Pengirim dan penerima memiliki cara sehingga keutuhan data dapat diselidiki, parameter dari fungsi *hash* yang biasa digunakan dapat dilihat pada Tabel 1 serta daftar algoritma fungsi *hash* pada Tabel 2.

Tabel 1. Parameter Fungsi Hash

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

Tabel 2. Daftar Algoritma Fungsi Hash

Algoritma	Output size	Internal state size	Block size	Lenght 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

Fungsi *hash h* memiliki sifat-sifat sebagai berikut:

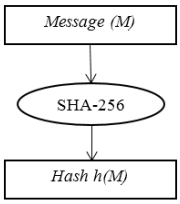
1. *Preimage resistant*
   Jika diberikan suatu nilai *hash y*, akan sulit mencari *M* sedemikian sehingga *h(M)=y*
2. *Second preimage resistant*
   Jika diberikan sebuah masukan *M*, akan sulit mencari *M'* sedemikian sehingga *h(M)=h(M')*
3. *Collision resistant*

Akan sulit untuk mencari *M* dan *M'* sedemikian sehingga *h(M)=h(M')*

2.6 Secure Hash Algorithm-256 (SHA-256)

Secure Hash Algorithm-256 adalah salah satu jenis *hash* yang masih umum digunakan. Fungsi ini adalah varian dari SHA-1, SHA-256 dibuat karena telah ditemukan bentrokan dari SHA-1, SHA-1 sendiri adalah pengganti dari SHA-0.

Sampai saat ini belum ada yang dapat memecahkan algoritma untuk SHA-256. SHA-256 umumnya digunakan sebagai fungsi antara untuk fungsi lain, termasuk fungsi *hash* MAC, HMAC, dan beberapa fungsi penghasil *digital signature* [3]. Fungsi utama SHA-256 dapat dilihat pada Gambar 5.



Gambar 5. Fungsi Utama SHA-256

2.7 RSA

RSA ditemukan oleh tiga orang yang kemudian disingkat menjadi RSA. Ketiga penemu itu adalah Ron Rivest, Adi Shamir, dan Leonard Adleman. RSA termasuk algoritma asimetri, yang berarti memiliki dua kunci, yaitu kunci publik dan kunci privat [7]. RSA menjadi sistem kriptografi kunci-publik yang terpopuler karena merupakan sistem pertama yang sekaligus dapat digunakan untuk *key distribution*, *confidentiality* dan *digital signature*. Parameter pembangkit kunci RSA dapat dilihat pada Tabel 3.

Tabel 3. Parameter Pembangkit Kunci RSA

Parameter	Keterangan
p	Bilangan prima
q	Bilangan prima
n	Merupakan hasil dari $p \times q$
ø (n)	Merupakan hasil dari $(p-1) \times (q-1)$
e	Dengan ketentuan, gcd (ø (n), e) = 1 gcd = *greatest common divisor*
d	e^{-1} *(mod ø (n))* Menggunakan algoritma *extended euclid*
Kpublik	*(e,n)*
Kprivat	*d*

Setelah kunci publik *Kpublik* dibangkitkan oleh pendekripsi, maka sembarang orang dapat menggunakan kunci publik tersebut. Algoritma enkripsi RSA menggunakan fungsi eksponensial dalam modular *n*, seperti yang dijelaskan pada Tabel 4.

Tabel 4. Algoritma Enkripsi RSA

Algortima Enkripsi RSA	
Input	*Kpublik = (e,n)*
Output	$C = P^e \bmod n$ Menggunakan algoritma *square and multipy*

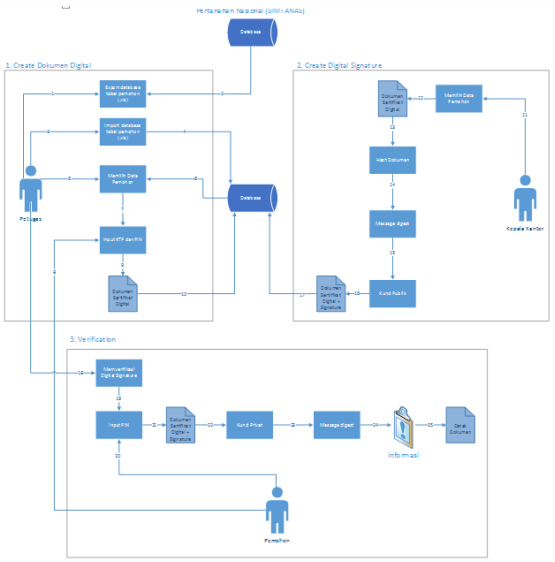
Seperti halnya enkripsi, algoritma dekripsi RSA merupakan fungsi eksponensial modular *n* dengan menggunakan kunci privat, seperti yang dijelaskan pada Tabel 5.

Tabel 5. Algoritma Dekripsi RSA

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

2.8 Pembahasan

Sistem yang akan dibagun dapat diakses oleh petugas dan kepala kantor BPN Kota Cimahi, sedangkan pemohon hanya memasukan PIN lewat petugas dimana arsitektur sistem yang akan dibangun diilustrasikan pada Gambar 6.



Gambar 6. Arsitektur Sistem

1. *Create* Dokumen Digital

Pertama petugas akan men-*export* data pemohon dengan format (*.xls*) yang berasal dari tabel pemohon pada *database*, kemudian petugas akan men-*import file* (*.xls*) tersebut ke sistem, petugas akan men-*create* dokumen digital milik pemohon ketika pemohon datang untuk yang kedua kalinya, namun sebelumnya petugas meminta pemohon untuk memasukan KTP dan PIN ketika akan men-

create dokumen digital. KTP dan PIN ini digunakan untuk men-*generate* kunci publik dan kunci privat.

2. *Create Digital Signature*

Kepala Kantor akan men-*hash* dokumen sertifikat digital yang telah di-*create* oleh petugas dengan menggunakan algoritma SHA-256, hasil dari dokumen yang telah di-*hash* akan menghasilkan *message digest*. Proses selanjutnya yaitu memproses *message digest* dengan kunci publik yang telah di-*generate* sebelumnya dengan menggunakan algoritma RSA. Setelah proses ini selesai maka akan menghasilkan *signature* dari dokumen sertifikat digital tersebut.

3. *Verification*

Verification dilakukan ketika pemohon datang untuk yang ketiga kalinya, ketika dokumen milik pemohon tersebut hilang ataupun rusak, petugas akan memverifikasi dokumen sertifikat digital yang telah diberi *digital signature* oleh kepala kantor dengan meminta pemohon untuk memasukan PIN yang sama dengan PIN yang telah dibuat sebelumnya, proses *verification* ini dilakukan untuk memastikan keabsahan bahwa dokumen sertifikat tanah digital ini benar-benar dari kepala kantor. *Verification* akan menggunakan kunci privat untuk memproses dokumen sertifikat digital yang telah diberi *digital signature* dengan menggunakan algoritma RSA, keluaran dari proses ini akan menghasilkan *message digest*.

Jika hasil *message digest* dari proses *hash* dokumen sertifikat digital yang dilakukan oleh kepala kantor sama dengan *message digest* yang dihasilkan dari proses *verification*, maka sistem akan memberikan informasi bahwa dokumen tersebut berhasil diverifikasi dan dokumen sertifikat tanah digital bisa langsung dicetak untuk keperluan pemohon, namun jika hasil *massage digest* yang dihasilkan dari proses *hash* tidak sama dengan *massage digest* yang dihasilkan dari proses *verification*, maka sistem akan memberikan informasi bahwa dokumen tersebut gagal diverifikasi.

2.9 Proses Hash

Pada proses *hash* menggunakan algoritma SHA-256 untuk menghasilkan *message digest* dari sebuah pesan, data dokumen serifikat digital yang sudah di *hash* dapat dilihat seperti Tabel 6.

Tabel 6. Hasil Message Digest Dokumen

Input dokumen	*Massage digest* (*Hex Format*)
Sertifikat_4802002.pdf	ec660b8304c12dfabf0b6df01ec352ee26d5249ff5f7e273dca39e0d6cec9884

2.10 Proses Pembangkitan Kunci

Proses pembangkitan kunci menggunakan algoritma RSA, dimana nantinya akan menghasilkan *p, q, n, ø (n), e, d, Kpublik*, dan *Kprivat*. Berikut ini akan dijelaskan lebih lanjut tentang proses pembangkitan kunci pada algoritma RSA:

1. Nomor KTP digunakan sebagai *p*, penjelasan mengenai nomor KTP dapat dilihat pada Gambar 7.

Gambar 7. Nomor KTP

Nomor KTP hanya diambil 6 digit terakhir saja, karena jika mengambil digit awal ada kemungkinan untuk nomornya sama dengan nomor KTP milik orang lain, dimana dua digit awal itu kode provinsi, lalu dikuti dengan dua digit kode kabupaten atau kota, dan dua digit selanjutnya adalah kode kecamatan. Maka dari itu dipilih 6 digit terakhir dimana dua digit awalnya terdiri dari tahun lahir dan empat digit selanjutnya adalah nomor komputerisasi yang angkanya pasti unik. Sedangkan untuk *q* menggunakan PIN 6 digit. Berhubung *p* dan *q* harus bilangan prima, maka bila *p* dan *q* bukan bilangan prima akan digunakan fungsi *next prime* untuk mendapatkan bilangan prima dari bilangan sebelumnya.

Misalnya p = 3273020312500032 dan q = 210195, *p* diambil 6 digit terakhir menjadi 500032, berhubung *p* dan *q* bukan bilangan prima maka dicari bilangan prima selanjutnya. Didapat p = 500041 dan q = 210209

2. Cari *n*, dimana *n* adalah modulus yang digunakan.

$n = p \times q$ = 500041 × 210209
= 105113118569

3. Hitung ø(*n*) = $(p - 1) \times (q - 1)$
= (500041 − 1) × (210209 − 1)
= 105112408320

4. Kunci publik (*e*), dimana *e* adalah eksponen publik (sering juga disebut eksponen enkripsi). sehingga nilai *e* relatif prima terhadap ø(*n*), misalnya ditentukan e = 7
Nilai 7 dipilih karena memenuhi syarat gdc(105112408320,7) = 1

5. Kunci privat (*d*) dengan menggunakan persamaan $d.e = 1 \pmod{ø(n)}$. Perhatikan bahwa $d.e = 1 \pmod{ø(n)}$ ekivalen dengan $e.d = 1 + k\ ø(n)$ sehingga secara sederhana *d* dapat dihitung dengan $d = (1+(k\ ø(n)) / e$. Dengan rumus tersebut maka di dapat nilai:
d = {1+(k × 105112408320)} / 7 dengan k = 1,2,3,4,....*n*. Dengan mencoba nilai-nilai k =

1,2,3,4.....*n* sehingga diperoleh *d* yang bulat, dipilih *k* = 2 menghasilkan:
d = {1+(2 × 105112408320)} / 7
= 30032116663

6. Maka *Kpublik* = (*e*,*n*) = (7, 105113118569) dan
7. *Kprivat* = (*d*,*n*) = (30032116663, 105113118569)

2.11 Proses Enkripsi

Proses enkripsi menggunakan algoritma RSA membutuhkan nilai *e* = 7 dan nilai *n* =105113118569 yang telah dibangkitkan sebelumnya. Selanjutnya memecah *message digest* yang dihasilkan dari proses *hash* dokumen sertifikat tanah menjadi blok-blok P_1, P_2,... P_n. Sebelum memecah *message digest* menjadi blok yang lebih kecil, *message digest* dirubah ke dalam kode ASCII terlebih dahulu seperti pada Tabel 7.

Tabel 7. Konversi Message Digest ke Kode ASCII

***Massage digest* dalam format Hex**	**ec6**0b8304c12dfabf0b6df01e c352ee26d5249ff5f7e273dca3 9e0d6cec9884
***Massage digest* dalam kode ASCII (P_i)**	**101 99 54** 54 48 98 56 51 48 52 99 49 50 100 102 97 98 102 48 98 54 100 102 48 49 101 99 51 53 50 101 101 50 54 100 53 50 52 57 102 102 53 102 55 101 50 55 51 100 99 97 51 57 101 48 100 54 99 101 99 57 56 56 52

Setiap karakter *plainteks (Pi)* dienkripsi menjadi 1 blok *cipherteks* (*Ci*) dengan rumus $Ci = P^e \bmod n$ dan diberi pemisah titik (.) seperti pada Tabel 8 dengan *e* = 7 dan *n* = 105113118569. Misal proses *message digest* yang akan dienkripsi adalah **ec6** dengan kode ASCII **101 99 54**.

Tabel 8. Proses Enkripsi

Kunci publik	(*e*, *n*)
	e = 7
	n = *105113118569*
Rumus Enkripsi	$C_i = P^e \bmod n$
Proses pemecahan dan enkripsi blok-blok P_i	$C_1 = 101^7 \bmod 105113118569$ = 103267388890
	$C_2 = 99^7 \bmod 105113118569$ = 40817281514
	$C_3 = 54^7 \bmod 105113118569$ = 77567787156
Hasil Enkripsi	**103267388890.40817281514.775 67787156**

2.12 Proses Dekripsi

Proses dekripsi dilakukan menggunakan algoritma RSA, setiap blok *cipherteks* (*Ci*) dirubah kembali menjadi blok *plainteks* (P_i) dengan rumus $P_i = C_i^d \bmod n$. Dekripsi dilakukan dengan menggunakan kunci privat *d* = 30032116663 dan *n* = 105113118569, kemudian blok–blok *cipherteks* didekripsikan seperti pada Tabel 9.

Tabel 9. Proses Dekripsi

Kunci privat	(*d*, *n*)
	d = 30032116663
	n = *105113118569*
Rumus Dekripsi	$P_i = C_i^d \bmod n$
Proses pemecahan dan dekripsi blok-blok P_i	$P_1 = 103267388890^{30032116663} \bmod 105113118569 = 101$
	$P_2 = 40817281514^{30032116663} \bmod 105113118569 = 99$
	$P_3 = 77567787156^{30032116663} \bmod 105113118569 = 54$
Hasil Dekripsi	**1019954**

Blok *message digest* dikembalikan dengan cara yang serupa, dari hasil dekripsi diperoleh kembali *message digest* semula seperti pada Tabel 8. Jika telah didekripsi menghasilkan *message digest* (*M*') yang sama dengan *message digest* (*M*) saat proses *hash*, berarti dokumen yang diterima merupakan dokumen yang sama dengan aslinya.

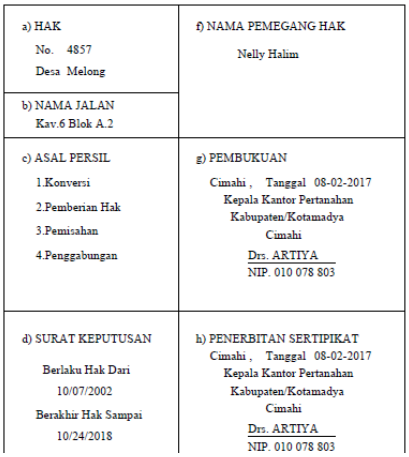
Tabel 10. Hasil Dekripsi

***Input* (*C*)**	103267388890.40817281 514.77567787156
***Output* (*P*) *Message digest* dalam kode ASCII**	1019954
Hasil *Message digest* dalam format Hex (*M'*)	ec6

Dari hasil dekripsi seperti pada Tabel 10 dapat disimpulkan bahwa dokumen yang diterima terbukti keasliannya, karena hasil *message digest* dari proses dekripsi sama dengan *message digest* dari dokumen asli ketika akan dienkripsi.

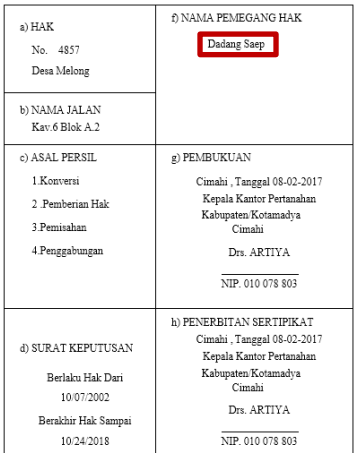
2.13 Hasil Pengujian

Autentifikasi dokumen dilakukan ketika pemohon memerlukan kembali dokumen sertifikat tanah yang telah hilang atau rusak. Pemohon harus memasukan PIN yang telah dibuat ketika pembuatan dokumen digital dilakukan. Pengujian autentifikasi dokumen dilakukan untuk memeriksa bahwa dokumen yang akan dicetak adalah dokumen yang benar-benar masih asli ketika pertama kali dokumen sertifikat tersebut dibuat, data sertifikat asli dapat dilihat pada Gambar 8.



Gambar 8. Dokumen Sertifikat Asli

Ketika pemohon ingin mencetak dokumen sertifikat tanah dan ternyata dokumen tersebut telah dirubah seperti pada Gambar 9 nama pemegang hak yang asalnya Nelly Halim diubah menjadi Dadang Saep, maka ketika verifikasi walaupun PIN yang dimasukan benar, tetapi akan muncul pemberitahuan bahwa dokumen tersebut telah dirubah seperti pada Gambar 10.



Gambar 9. Dokumen Sertifikat yang Telah Diubah

Gambar 10. Notifikasi Dokumen Telah Dirubah

Berdasarkan hasil pengujian yang dilakukan dengan kasus uji, dapat ditarik kesimpulan bahwa jika data masukkan benar, maka sistem akan mengeluarkan keluaran sesuai harapan. Jika data yang dimasukkan salah, maka sistem akan menunjukkan pesan kesalahan sesuai dengan kesalahannya.

3. PENUTUP

Berdasarkan hasil dan pengujian yang telah dilakukan maka dapat diperoleh kesimpulan sebagai berikut:

1. Penerapan mekanisme pembuatan dokumen digital didalam sistem dapat memberikan layanan alternatif untuk menerbitkan dokumen digital.
2. Penerapan mekanisme keamanan *digital signature* dapat memberikan layanan keamanan berupa otentikasi dokumen yang diterapkan pada dokumen digital pemohon oleh kepala kantor sehingga pemohon dapat mengetahui validitas dokumen yang diterima.

Adapun beberapa saran untuk pengembangan sistem, diantaranya:

1. Perlunya integrasi database dari sistem yang dibangun ini ke Sistem Informasi yang ada, agar data pemohon menjadi lebih mudah di-*update.*
2. Perlunya pengembangan *digital signature* dengan melibatkan *certificate author* (CA) untuk menciptakan mekanisme *digital signature* yang lebih baik.

DAFTAR PUSTAKA

[1] Isvarahadi, Fansiskus Asisi Tri. "Implementasi Digital Signature Menggunakan LSB Embedding untuk Uji Keutuhan, Otentikasi dan Penyangkalan Dokumen Pertanahan Digital", Universitas Kristen Satya Wacana. 2015. Tersedia pada : http://repository.uksw.edu/bitstream/123456789/14992/2/T1_672008139_Full%20text.pdf

[2] Dwiperdana, Aditia, "Cryptographic Hash Function dan Penggunaannya Dalam Digital Signature", Institut Teknologi Bandung. 2007. Tersedia dalam pada http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis /2006-2007/Makalah/Makalah0607-42.pdf

[3] Azhar, Hanifah, "Perbandingan Algoritma Fungsi Hash MD5 dengan SHA-1", Institut Teknologi Bandung. 2013. Tersedia dalam pada : http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2012-2013/Makalah2 2013/Makalah2Kripto2013-045.pdf

[4] Pratama, Aditya, "Studi Perbandingan dan Implementasi Kombinasi Fungsi Hash dan Kriptografi Kunci-Publik", Institut Teknologi Bandung. 2011. Tersedia pada : http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2010-2011/Makalah2/Makalah2-IF3058-Sem2-2010-2011-016.pdf

[5] Basharat, I, et al., "Database Security and Encryption: A Survey Study", International Journal of Computer Applications. vol.44, pp. 888-975, June 2012.

[6] Bin, Ladjamudin, Al-Bahra, Analisis dan Desain Sistem Informasi, Yogyakarta : Graha Ilmu, 2005.

[7] Munir, Rinaldi, Kriptografi, Bandung : Informatika, 2006.

[8] Kromodimoeljo, Sentot, Teori dan Aplikasi Kriptografi, Jakarta : SPK IT Consulting, 2010.