IMPLEMENTASI *DIGITAL SIGNATURE ALGORITHM* (DSA) MENGGUNAKAN *SECURE HASH ALGORITHM*-256 (SHA-256) PADA MEDIA GAMBAR

Sahl Fawzy Sutopo*, Rini Marwati, dan Cece Kustiawan

Program Studi Matematika
Fakultas Pendidikan Matematika dan Ilmu Pengetahuan Alam
Universitas Pendidikan Indonesia
*surel : sahlfawzys@gmail.com

ABSTRAK. Di era digitalisasi ini, dibutuhkan suatu teknik keamanan dokumen digital agar dokumen tersebut terjaga keasliannya yaitu dengan memberikan tanda tangan digital. Tanda tangan digital bukanlah tanda tangan manual yang didigitalkan melainkan sebuah pengkodean yang didapat dari proses *Digital Signature Algorithm* (*DSA*). Penelitian ini bertujuan untuk menjelaskan proses implementasi *Digital Signature Algorithm* dengan menggunakan fungsi *hash*, yaitu *Secure Hash Algorithm*-256 pada media gambar. Melalui penelitian ini, diperoleh suatu program untuk membangkitkan kunci, membangkitkan tanda tangan, dan autentikasi tanda tangan digital.

Kata Kunci : Kriptografi, Fungsi *Hash*, Kunci Asimetris, *Digital Signature Algorithm*, *Secure Hash Algorithm-256.*

IMPLEMENTATION OF DIGITAL SIGNATURE ALGORITHM (DSA) USING SECURE HASH ALGORITHM-256 (SHA-256) IN IMAGE FILE

ABSTRACT. *In this digitalization era, a digital document security technique is needed so that the authenticity of the document is maintained, namely by added a digital signature. Digital signature is not a signature that is digitized but rather an encoding obtained from the Digital Signature Algorithm process. This study aims to explain the implementation process of the Digital Signature Algorithm using the Secure Hash Algorithm-256 as a hash function on an image file. Through this research, it has been successfully created a program to generate keys, generate signatures, and authenticate digital signatures.*

Keywords*: Cryptography, Hash Function, Asymmetric Key, Digital Signature Algorithm, Secure Hash Algorithm-256.*

1. PENDAHULUAN

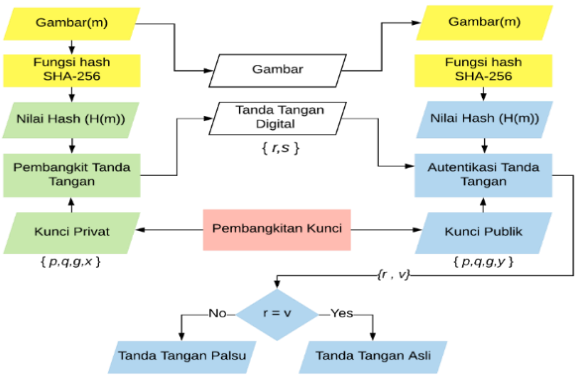
Dengan semakin berkembangnya teknologi, kini dokumen tidak hanya diproduksi dalam bentuk cetak tetapi diproduksi juga dalam bentuk digital. Untuk menjaga keaslian pada dokumen tersebut, dibutuhkanlah tanda tangan. Terdapat dua jenis tanda tangan, yaitu tanda tangan konvensional dan tanda tangan digital. Tanda tangan konvensional bersifat statis (tetap) yang berarti tanda tangan yang sama dapat digunakan pada dokumen yang berbeda sedangkan tanda tangan digital bersifat dinamis yang berarti tanda tangan yang sama tidak dapat digunakan pada dokumen yang berbeda. Maka dari itu, tanda tangan digital lebih aman dibandingkan tanda tangan konvensional. Tanda tangan digital bukanlah tanda tangan konvensional yang didigitalkan melainkan sebuah kode yang diproses menggunakan proses kriptografi, dimana dalam penelitian ini menggunakan *Digital Signature Algorithm* (DSA) [1].

DSA merupakan kriptografi yang digunakan dalam autentikasi pesan. Dengan pengaplikasian DSA, penerima dapat langsung mengetahui bahwa pesan yang diterimanya terjadi perubahan atau tidak dari dokumen asli. Dalam prosesnya, DSA memerlukan fungsi *hash*. Fungsi *hash* adalah suatu fungsi yang dapat mengkonversi pesan teks dengan panjang karakternya sembarang menjadi pesan teks acak dengan panjang karakter yang tetap [2]. Fungsi *hash* dapat memungkinkan terjadi *collision* (penumbukan) yang mengakibatkan dua dokumen berbeda dapat memiliki nilai *hash* yang sama. DSA umumnya menggunakan fungsi *hash* SHA-1 pada prosesnya namun pada 23 februari 2017 google security mengumumkan bahwa CWI Institute in Amsterdam beserta Google berhasil menemukan kecacatan berupa penumbukan pada SHA-1 [3]. Maka dari itu digunakanlah SHA-256 sebagai penggantinya.

Pada makalah ini, dijabarkan pengkonstruksian sebuah program DSA menggunakan Bahasa pemrograman python untuk mempermudah pengaplikasian DSA pada gambar. Program terdiri atas tiga bagian, yaitu pembangkit kunci untuk mendapatkan kunci privat, pembangkit tanda tangan digital untuk mendapatkan tanda tangan digital dari pesan gambar, dan autentikasi tanda tangan untuk mengecek keaslian pesan gambar.

2. METODE

Umumnya DSA diimplementasikan pada media teks menggunakan SHA-1, namun SHA-1 telah ditemukan penumbukan dan juga saat ini dokumen digital tidak hanya dibuat menjadi media teks namun juga media gambar, contohnya ialah sertifikat. Adapun *flowchart* dari model DSA ini dapat dilihat dalam Gambar 1.



Gambar 1. *Flowchart DSA*

Dapat dilihat dalam Gambar 1, terdapat 4 tahapan yang direpresentasikan dengan warna dalam proses DSA menggunakan SHA-256 pada gambar. Warna kuning merepresentasikan proses *hashing* pada gambar, warna merah merepresentasikan proses pembangkitan kunci, warna hijau merepresentasikan proses pembangkitan tanda tangan, dan warna biru merepresentasikan proses autentikasi tanda tangan. Adapun detail tahapan-tahapan dalam algoritma ini adalah sebagai berikut :

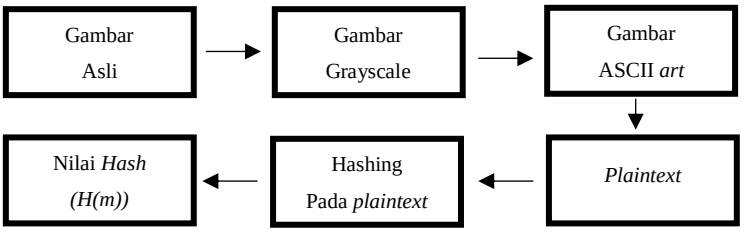
2.1. Pembangkitan Kunci

Kunci yang dibangkitkan merupakan kunci privat dan kunci publik. Kunci privat digunakan untuk membangkitkan tanda tangan digital sedangkan kunci publik digunakan untuk autentikasi tanda tangan digital. Adapun kunci yang dibangkitkan ialah:

- p, adalah bilangan prima dengan $p \in \mathbb{Z}^+$
- q, bilangan prima yang merupakan pembagi dari $p - 1$ dengan $q \in \mathbb{Z}^+$. Dengan kata lain, $(p - 1)\ mod\ q = 0$.
- $g = h^{\frac{p-1}{q}}\ mod\ p$, dengan $1 < h < p - 1$ dan $h^{\frac{p-1}{q}}\ mod\ p > 1$ dengan $g, h \in \mathbb{Z}$
- x, dengan $x < q,\ x \in \mathbb{Z}$. Parameter x adalah kunci privat.
- $y = g^x\ mod\ p$, Parameter y adalah kunci publik

2.2. Hashing Pada Media Gambar

Sebelum memasuki proses algoritma DSA, pesan terlebih dahulu dienkripsi menggunakan fungsi *hash* SHA-256. Karena *hashing* hanya dapat dilakukan pada plaintext maka pesan gambar akan diubah menjadi *plaintext* lalu dilakukan proses *hashing*. Setelah mendapatkan nilai *hash* (*message digest*) dari file gambar, maka dapat dilanjut ke proses DSA.



Gambar 2. *Hashing* pada media gambar.

Gambar 2 menampilkan langkah-langkah dalam proses *hashing* pesan gambar, yaitu:

1) Gambar asli diubah menjadi gambar grayscale dengan bantuan *packages* pillow yang telah disediakan di bahasa pemrograman Python.
2) Gambar grayscale diubah menjadi gambar ASCII *art* menggunakan teknik *mapping*. Setiap warna pada gambar dapat dikonversi menjadi angka menggunakan packages pillow yang tersedia di bahasa pemrograman Python. Rumus untuk memetakan interval warna ke huruf adalah sebagai berikut :

$$C_i = \left\lfloor \frac{W_i}{255} \times interval \right\rfloor$$

$$W_i \xrightarrow{dipetakan} K_{C_i}$$

Dengan :

K_i = Karakter ke-i
C_i = indeks interval ke-i
W_i = Nilai warna ke-i
Interval = Banyaknya interval yang digunakan (dalam penelitian ini digunakan 60 interval)

3) Gabungkan setiap karakter hingga membuat sebuah *plaintext*.
4) Hashing *plaintext* menggunakan *Secure Hash Algorithm-256* (SHA-256). Setelah *plaintext* telah diproses menggunakan SHA-256, maka didapatlah H(m) atau nilai *hash*.

2.3. Pembentukan dan Autentikasi Tanda Tangan

Pembentukan dan autentikasi tanda tangan dalam penelitian ini dilakukan menggunakan algoritma DSA. Proses pembentukan tanda tangan memerlukan input berupa kunci privat dan nilai *hash* gambar dan menghasilkan *output* berupa tanda tangan digital. Pada proses autentikasi tanda tangan memerlukan input berupa kunci publik dan nilai *hash* pesan gambar. Pada proses ini akan dicek apakah tanda tangannya sesuai dengan pesan gambar ataukah tidak. Adapun algoritmanya adalah sebagai berikut:

1) Algoritma proses pembangkitan tanda tangan digital.
   a. Ubah pesan *m* menjadi *message digest* dengan fungsi *hash* SHA-256.

b. Tentukan bilangan acak $k < q$.

c. Tanda-tangan dari pesan *m* adalah bilangan *r* dan *s*. Hitung *r* dan *s* sebagai berikut:

$$r = \left(g^k \bmod p\right) \bmod q$$

$$s = \left(k^{-1}(H(m) + x * r)\right) \bmod q$$

d. Kirim pesan *m* beserta tanda-tangan *r* dan *s*.

2) Algoritma autentikasi tanda tangan digital.

Hitung :

a. $w = s^{-1} \bmod q$

b. $u_1 = (H(m) * w) \bmod q$

c. $u_2 = (r * w) \bmod q$

d. $v = ((g^{u_1} * y^{u_2}) \bmod p) \bmod q$

e. Jika *v* = *r*, maka tanda-tangan sah, yang berarti bahwa pesan masih asli dan dikirim oleh pengirim yang benar.

Agar proses penggunaan *digital signature* dapat berjalan dengan baik, maka diperlukan sebuah protokol kriptografi. Misalkan Alice berperan sebagai orang pertama dan Bob berperan sebagai orang kedua, maka protokol kriptografi model *Digital Signature Algorithm* dengan *Secure Hash Algorithm-256* pada media gambar adalah sebagai berikut:

1) Alice meringkas media gambarnya menjadi *message digest* dengan menggunakan fungsi *hash* SHA-256.
2) Alice membangkitkan kunci publik dan kunci privat.
3) Alice membangkitkan tanda tangan *message digest* menggunakan kunci privatnya. Hasil enkripsinya digunakan sebagai tanda tangan digital yang disertakan dengan file gambarnya.
4) Alice mengirim gambar beserta tanda tangan digitalnya kepada Bob.
5) Bob meringkas media gambar dari Alice menjadi *message digest* dengan fungsi *hash* SHA-256. Bob melakukan autentikasi tanda tangan digital yang disertakan pada file gambar Alice. Jika hasil output autentikasi tanda tangannya bernilai sama, maka tanda tangan digital tersebut sah. [4]

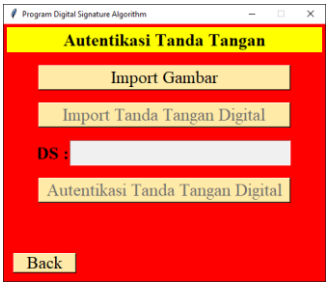
Agar proses DSA dapat diimplementasikan dengan mudah pada pesan gambar, maka dibuatlah sebuah program komputer. Program dibuat menggunakan bahasa pemrograman Python 3.8 dengan bantuan software Notepad++ dan Spyder dengan antarmuka program ditampilkan pada Gambar 3.

Gambar 3. Menu DSA.

Gambar 4. Pembangkit Kunci.

Program DSA yang dikonstruksi memiliki tiga proses yaitu proses pembangkitan kunci, pembangkitan tanda tangan, dan autentikasi tanda tangan dengan antarmuka program dapat dilihat pada Gambar 3. Dalam proses pembangkitan kunci, bilangan prima akan dibuat otomatis oleh sistem sedangkan nilai *x* merupakan kunci privat yang ditentukan oleh pengguna, adapun antarmuka program dapat dilihat pada Gambar 4. Dalam proses pembangkitan tanda tangan digital diperlukan kunci privat dan file gambar yang akan diberi tanda tangan, adapun antarmuka program dapat dilihat pada Gambar 5. Pada proses pembangkit tanda tangan akan menghasilkan tanda tangan digital yang telah digabung dengan kunci publik. Dalam proses autentikasi tanda tangan digital yang ditampilkan pada Gambar 6, pengguna memerlukan gambar beserta tanda tangan digitalnya yang akan dicek keasliannya. Pada proses ini, program akan menampilkan keterangan "Gambar ini asli" apabila gambar cocok dengan tanda tangan digitalnya sebaliknya apabila menampilkan keterangan "Gambar ini palsu" maka gambar tidak cocok dengan tanda tangan digitalnya.

Gambar 5. Pembangkit Tanda Tangan



Gambar 6. Autentikasi Tanda Tangan

3. HASIL DAN PEMBAHASAN

3.1. Proses Pembangkitan Kunci

Proses pembangkitan kunci akan menghasilkan output berupa kunci publik dan kunci privat. Hitung nilai parameter { p, q, h, g, x, y }, diperoleh :

$p =$ 126079601550846295330440265251462547902668625323616708003593936470760676484732102505855022457750610298577653282606143722688989483249729244752923004082666031188886944507336986943397034245626647509997225638194084988908232115412753017224484879800981511103137460091545691874765269666965030526746298760835839015883.

$q =$ 900568582506044966646001894653303913590490180882976485739956689076861974890943589327535874698218644989840380590043883733492782023212351748235164314876185937063478175052407049595693101754476053642837325987100607063630229395805378694460606284292725079308124714939612084819751926192607360905330705434541707256 3.

$h =$ 653321883111799394129489095291442849479179793498505735786957419849792686999813701470173377615438168083250859090707420855305117563340520983656195171649572997198071902660761301323276777398951720119582659271206801818506003264004501131825903569011853406508985911710906085801102775490320939117110627556900270421 5.

$g =$ 459455183409466845151580995630833922551271513583682173881704760753148373150175265378567619258341281552978338622430894735970211880426459854372386219232513801255128604980349655394028224799553916409871694538086112780386900026201805146787902103674651724245984964362280113150467176380883026930510770221427986640 85.

$x =$ 876123587547154781572547254751275471854871271274618641762472458648176247162745455451724687162476182764716247188842176471286124.

$y =$ 162858327562906391320200729967708760049965698621523674288783576466166699748624830024396909879453059327441356084231697570814740283756886090486925077324223114795571649659347128942769247920713164923180500659935383594193620738695235015593161062755235314960921389610280367901212027247110138316897728477967033793 12.

Didapatlah kunci publik $\{p, q, g, y\}$ dan kunci privat $\{p, q, g, x\}$.

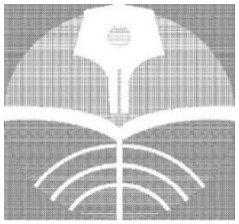
3.2. Pembangkitan Tanda Tangan Digital

Dalam proses tanda tangan digital ini akan dilakukan *hashing* pada file gambar dan pembentukan tanda tangan digital. Proses ini memerlukan input berupa file

gambar dan kunci privat. Adapun contoh input file gambar yang digunakan adalah logo UPI dengan ukuran file 252×260 pixels, yang terdapat pada Gambar 7.

Gambar 7. Logo UPI

Gambar 8. *Grayscale.*



Gambar 9. ASCII Logo

Sebelum dilakukan *hashing*, Gambar 7 diubah terlebih dahulu menjadi pesan teks. Pertama-tama Gambar 7 diubah menjadi gambar *grayscale*, didapatlah Gambar 8. Setelah itu tiap blok *pixel* warna pada Gambar 8 dilakukan pemetaan terhadap karakter sehingga menghasilkan gambar ASCII, didapatlah Gambar 9 dengan tiap blok *pixel* tidak lagi memuat warna namun berupa memuat karakter.

Setelah setiap karakter yang terdapat dalam Gambar 9 digabungkan menjadi pesan teks, maka hitung nilai *hash*. Diperoleh :

$H(m) =$ 49686519568737992458201025053675566434855544889651231820526195511351800108798.

Setelah itu bangkitkan tanda tangan pada gambar logo UPI dengan cara menghitung nilai $k, k_{inv}, r,$ dan s, diperoleh :

$k =$ 59977161535172193889841244998994832135415973642566916029170609428972618370961393992516002194799790589868176513879662357333617943075525304896358587265983709236140885116721508446525398579710758109587307688796446288815477442515075657774027016691573870881372521337064320425354089155867276767462401712775979872 88.

$k^{-} =$ 40972224661543326442497013065596849432028868460446448623183044909085991525121229526643416076937027239651974892304845626711548770692804098164819432712782979988685515607168490404471398227409218772035182596118251596164089643744502209510542219681039518199541394848802266262421376546763352072044142883043050121 04.

$r =$ 20243637044199598073255509545178510096704984086131654146194733916209234061287548137454009717319044436252342349048341328795625648653798184867276330142593055744306395100781126667240902272628771348972732775137374521909092387229030398518798672643806421439592228436742357129005660473184279378877945392727115812 82.

$s =$ 58369092956436925137472291658514923827391895249387147 13

60424134573912656263018882605376389905955084058125348878499
25266224102035973856600135403748770563863080640637289585820
53552120697530848057353447462044901709598644637465454160873
03175652799560752537364893924785413499106353482043851852006
4652323128082947.

Nilai r dan s digunakan sebagai tanda tangan digital dari gambar logo UPI. Setelah itu, kirimkan kunci publik, file gambar beserta tanda tangan digitalnya ke orang lain. Orang yang menerima kunci publik, file gambar dan tanda tangan digital dapat mengecek keaslian tanda tangan tersebut.

3.3. Autentikasi Tanda Tangan Digital

Proses autentikasi tanda tangan digital memerlukan input berupa kunci publik, pesan gambar, dan tanda tangan digital. Karena dalam tahap sebelumnya telah mendapatkan kunci publik $\{p, q, g, y\}$, pesan gambar, dan tanda tangan digital $\{r, s\}$ maka dapat dilakukan proses autentikasi tanda tangan digital.

Hitung nilai parameter $\{w, u_1, u_2, v\}$ lalu bandingkan nilai *r* dan *v*, diperoleh :

$w =$ 25084590397011690636042010722605604267782371679290481919415753297906174540459079635021691633112227219422834632242793259958184617698661107666332667328705166933649319635420894861946787829287392393689102257113275017173996675129018564776614631218469910553110984542528326256832589780573137575288500295712966444 71.

$u_1 =$ 76886214499546254925239566232210635261283212525924673725577957913277052906173825176947304325732426448152632796929784034723616982807039390122382204766314393370791450367843400975314694408111971193766376950296563982963660225248520602058313549353151947382356162812928844364203566164562305725454796813080301505 32.

$u_2 =$ 53361397464840719993993513720198559248878628359008512643952431612091758531075364269184309519434495248770447304525620397496898978973314975581734729299081736012015501656892767433712714205349238165445564081170628582148206001901299729304554325706616785192129171511148504589342164891745889444673084267926563849 56.

$v =$ 20243637044199598073255509545178510096704984086131654146194733916209234061287548137454009717319044436252342349048341328795625648653798184867276330142593055744306395100781126667240902272628771348972732775137374521909092387229030398518798672643806421439592228436742357129005660473184279378877945392727115812 82.

Karena $r = v$, maka dapat disimpulkan bahwa tanda tangan digital asli yang berarti file gambar tidak ada perubahan sedikitpun.

4. KESIMPULAN

Berdasarkan hasil dan pembahasan yang telah dipaparkan dalam bab sebelumnya maka didapatlah kesimpulan sebagai berikut:

1) *Digital Signature Algorithm* (DSA) memiliki tiga proses yaitu pembangkitan kunci, pembangkitan tanda tangan, dan autentikasi tanda tangan. DSA tidak hanya dapat diimplementasikan pada pesan teks namun dapat diimplementasikan pada pesan gambar yang telah dimodifikasi. .
2) Algoritma DSA dapat dikonstruksi menjadi program komputer sehingga dapat mempermudah pengguna (baik pengirim maupun penerima) dalam hal pembentukan tanda tangan dan autentikasi tanda tangan. Program dibuat menggunakan Bahasa Pemrograman Python 3.8 dengan spesifikasi komputer RAM 4 GB dan *processor* AMD A9. Program yang dibuat memiliki tiga menu utama yaitu menu pembangkit kunci, menu pembangkit tanda tangan digital, dan menu autentikasi tanda tangan digital. Menu pembangkit kunci digunakan untuk membangkitkan kunci publik dan kunci privat, menu pembangkit tanda tangan digital berfungsi untuk membangkitkan tanda tangan pada file gambar, dan menu autentikasi tanda tangan digital berfungsi untuk mengecek apakah tanda tangan digital yang digunakan asli atau palsu.

5. DAFTAR PUSTAKA

[1] R. A. Azdy, "Tanda tangan Digital Menggunakan Algoritme Keccak dan RSA," *J. Nas. Tek. Elektro dan Teknol. Inf.*, vol. 5, no. 3, pp. 184–191, 2016, doi: 10.22146/jnteti.v5i3.255.

[2] R. Munir, "Penggunaan Tanda-Tangan Digital Untuk Menjaga Integritas Berkas Perangkat Lunak," *Semin. Nas. Apl. Teknol. Inf. 2005*, vol. 2005, no. Snati, pp. 6–9, 2005.

[3] Stevens Marc *et al.*, "Google Online Security Blog: Announcing the first SHA1 collision," Jan. 23, 2017. https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html (accessed Jul. 07, 2020).

[4] R. Munir, "Protokol Kriptografi Bahan Kuliah Protokol," 2017.