Jurnal Informatika Terpadu

https://journal.nurulfikri.ac.id/index.php/JIT
ISSN ONLINE : 2460-8998

VISUALISASI IMPLEMENTASI AUTENTIFIKASI *DIGITAL SIGNATURE* MENGGUNAKAN FUNGSI HASH PADA FILE DIGITAL

Rufaidah Taqiyyah[1], Ahmad Rio Adriansyah[2]

[1, 2]Teknik Informatika, Sekolah Tinggi Teknologi Terpadu Nurul Fikri
Jakarta Selatan, DKI Jakarta, Indonesia 12640
rufaidah.chan@gmail.com , ahmad.rio.adriansyah@gmail.com

Abstract

Information security is fundamental to maintain the integrity and authenticity of a document. Cryptography is the study of mathematical techniques related to aspects of information security. Cryptographic data integrity algorithms use to protect data blocks, such as protecting documents from changes, one of which is the digital signature algorithm. With a digital signature, we can verify the authenticity of the document and can find out whether the contents of the document have changed or not. However, no system is one hundred percent safe; one way to minimize losing security is to understand the security system. In this study, the author will create a system that visualizes the process of the digital signature algorithm to make it easier for someone to learn it.

Keywords: *Information System Security, Cryptography, Digital Signature Algorithm*

Abstrak

Keamanan informasi sangatlah penting untuk menjaga integritas dan keaslian dari suatu dokumen. Kriptografi adalah studi teknik matematika yang berkaitan dengan aspek keamanan informasi. Algoritma integritas data kriptografi digunakan untuk melindungi blok data seperti melindungi dokumen dari perubahan, salah satunya adalah algoritma *digital siganture*. Dengan *digital signature* kita dapat memverifikasi keaslian dari dokumen dan dapat mengetahui isi dokumen tersebut sudah diubah atau belum. Namun tidaklah ada sistem yang seratus persen aman, salah satu cara untuk meminimalisir resiko dari kehilangan keamanan adalah dengan memahami sistem keamanan tersebut. Pada penelitian kali ini, penulis akan membuat sistem yang memvisualisasikan proses dari algoritma *digital signature* untuk memudahkan seseorang dalam mempelajarinya.

Kata kunci: Keamanan Sistem Informasi, Kriptografi, Algoritma Digital Signature

1. PENDAHULUAN

Keamanan sistem informasi sangatlah penting untuk menjaga integritas dan keaslian dari suatu dokumen. Salah satu cara untuk menjaga integritas dan keaslian dari suatu dokumen adalah dengan menggunakan *digital signature*. Dengan *digital signature* kita dapat memverifikasi keaslian dari dokumen dan dapat mengetahui isi dokumen tersebut sudah diubah atau belum.

Namun tidaklah ada sistem yang seratus persen aman, selalu ada seseorang yang mencari celah dari suatusistem, entah untuk membuat sistem tersebut lebih baik ataupun untuk menghancurkan sistem tersebut. Ketika seseorang merasa nyaman dengan suatu sistem, maka kewaspadaan terhadap keamanannya akan melemah. Banyak resiko yang didapatkan ketika keamanan itu hilang, mulai dari yang kecil sampai besar.

Pemahaman terhadap keamanan sangatlah penting sebagai salah satu cara untuk meminimalisir resiko dari kehilangan keamanan. Oleh sebab itu penulis akan memvisualisasikan proses dari algoritma *digital signature* untuk memudahkan seseorang dalam mempelajari salah satu sistem keamanan dokumen.

2. KONSEP KEAMANAN KOMPUTER, ALGORITMA INTEGRITAS DATA KRIPTOGRAFI, *DIGITAL SIGNATURE ALGORITHM*

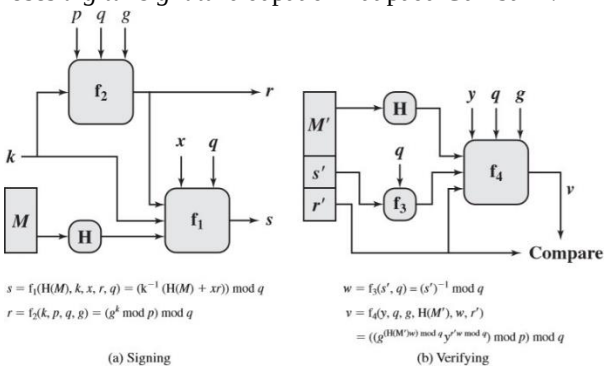
Menurut *National Institute of Standards and Technology (NIST) Computer Security Handbook*, keamanan komputer adalah perlindungan yang diberikan pada sistem informasi otomatis untuk mencapai tujuan yang berlaku untuk menjaga integritas, ketersediaan, dan kerahasiaan sumber daya sistem informasi (temasuk *hardware*, *software*,

Diterima 05 Januari 2020, Direvisi 26 Februari 2020, Diterima untuk publikasi 06 Maret 2020

firmware, informasi atau data, dan telekomunikasi). Dari pengertian tersebut dapat diambil tiga konsep untuk memenuhi tujuan utama dari keamanan komputer, yaitu integritas, ketersediaan, dan kerahasiaan yang membentuk apa yang sering disebut sebagai *CIA triad*. Namun di beberapa bidang keamanan menyatakan perlu adanya konsep tambahan, diantaranya yang sering disebutkan adalah keaslian dan akuntabilitas [1]. Pada penelitian ini akan difokuskan kepada dua konsep keamanan yaitu integritas untuk menjamin keaslian dari dokumen dan keaslian untuk memastikan dokumen bersifat asli, dapat diverifikasi, dan dapat dipercaya.

Kriptografi adalah studi teknik matematika yang berkaitan dengan aspek keamanan informasi. Layanan dasar yang disedikan kriptografi adalah kemampuan mengirim informasi antar pengguna tanpa ada orang lain yang dapat membacanya [2]. Algoritma integritas data digunakan untuk melindungi blok data, seperti melindungi suatu dokumen dari perubahan. *Digital signature* adalah salah satu algoritma untuk menjaga integritas suatu dokumen.

Teknologi tanda tangan digital memanfaatkan sepasang kunci privat-pubik yang dibuat untuk keperluan seseorang. Kunci privat disimpan oleh pemiliknya dan digunakan untuk membuat tanda tangan digital, sedangkan kunci publik dapat digunakan oleh siapa saja yang ingin memeriksa tanda tangan digital yang bersangkutan pada suatu dokumen [3]. DSA adalah salah satu mekanisme otentikasi, karna algoritma ini memungkinkan pembuat pesan untuk melampirkan kode yang bertindak sebagai tanda tangan digital. Dengan algoritma ini, pembuat dan penerima pesan dapat terlindungi satu sama lain yang memungkinkan terjadi perselisihan antar keduanya [5]. Proses *digital signature* dapat dilihat pada Gambar 1.

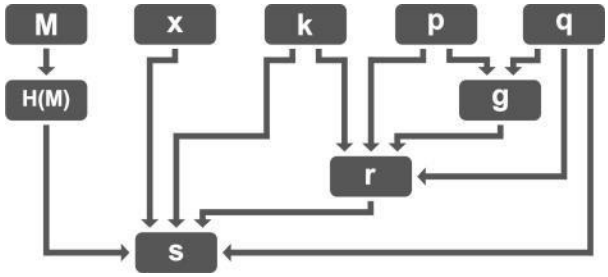


Gambar 1. Proses *Digital Signature* [4]

3. ANALISIS ALGORITMA *DIGITAL SIGNATURE*

Analisis ini dilakukan berdasarkan studi pustaka yang sudah dilakukan sebelumnya. Berdasarkan hasil analisis tersebut, penulis membuat daftar apa saja yang dibutuhkan dalam pembuatan algoritma sistem keamanan yaitu *digital signature* yang terdiri dari dua proses yaitu *signing* dan *verifying*.

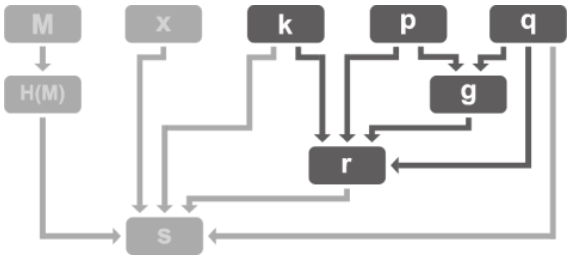
3.1 Proses *Signing*



Gambar 2. Kebutuhan Proses *Signing*

Proses *signing* adalah proses untuk membuat *digital signature* terhadap dokumen yang menandakan bahwa dokumen tersebut adalah benar milik dari seseorang yang membuat file tersebut. *Output* dari proses *signing* adalah nilai s dan nilai r yang nantinya akan dipakai untuk proses *verifying*. Keseluruhan *item* yang dibutuhkan dalam proses signing dapat dilihat pada Gambar 2.

1. Nilai r
   Nilai r adalah nilai yang didapatkan dari komputasi nilai rahasia per-dokumen (k) dan komponen G*lobal public-key* yaitu bilangan prima (p), pembagi utama (q), dan nilai g.



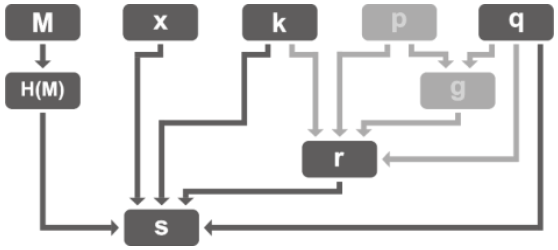
Gambar 3. Proses Mendapatkan Nilai r

Tabel 1. Komputasi untuk Mendapatkan Nilai r

No	Yang Dibutuhkan	Komputasi	
1	Nilai rahasia per-dokumen (k)	Random atau pseudorandom integer **$0 < k < q$**	**$(g^k \bmod p) \bmod q$**
2	Nilai p (komponen *Global public-key*)	Bilangan prima **$2^{(L-1)} < p < 2^L$** L = kelipatan 64512 ≤ L ≤ 1024	
3	Nilai q (komponen *Global public-key*)	Pembagi utamadari (p-1) **$2^{159} < q < 2^{160}$**	
4	Nilai g (komponen *Global public-key*)	**$h^{(p-1)/q} \bmod p$** $1 < h < (p-1)$ $g > 1$	

2. Nilai s

Nilai s adalah nilai yang didapatkan dari komputasi nilai hash (H(M)), *private-key* (x), nilai rahasia per-dokumen (k), nilai r, dan pembagi utama (q).



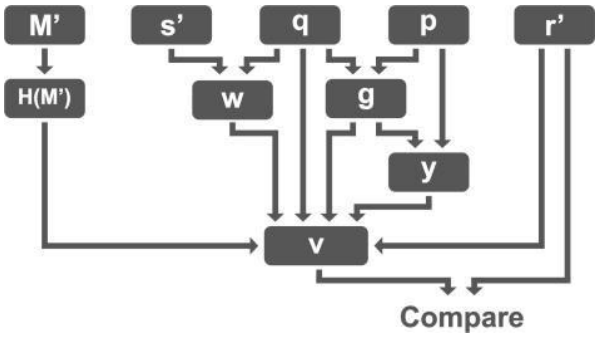
Gambar 4. Proses Mendapatkan Nilai s

Tabel 2. Komputasi untuk Mendapatkan Nilai s

No	Yang Dibutuhkan	Komputasi	
1	Nilai hash (H(M))	**M → f(H) → H(M)**	
2	*Private-key* (x)	Random atau pseudorandom integer **0 < x < q**	
3	Nilai rahasia per-dokumen (k)	Random atau pseudorandom integer **0 < k < q**	**$(k^{-1}$ (H(M) + xr)) mod q**
4	Nilai r	**$(g^k$ mod p) mod q**	
5	Nilai q (komponen *Global public-key*)	Pembagi utamadari (p-1) **$2^{159} < q < 2^{160}$**	

3.2 Proses *Verifying*

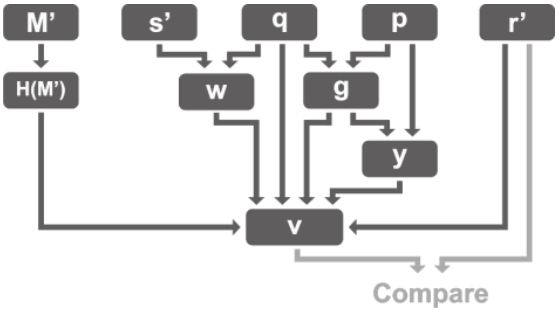
Proses *verifying* adalah proses untuk membutikan bahwa dokumen tersebut adalah benar milik dari seseorang yang membuat file tersebut atau sudah diubah oleh orang lain. *Output* dari *verifying* adalah nilai v yang nantinya akan di bandingkan dengan nilai r. Keseluruhan *item* yang dibutuhkan dalam proses *verifying* dapat dilihat pada Gambar 5.



Gambar 5. Kebutuhan Proses *Verifying*

1. Nilai v

Nilai v adalah nilai yang didapatkan dari komputasi nilai hash (H(M)); nilai hasil *signing* yaitu nilai s dan nilai r; komponen *Global public-key* yaitu bilangan prima (p), pembagi utama (q), dan nilai g; *public-key* (y); dan nilai w.



Gambar 6. Proses Mendapatkan Nilai v

Tabel 3. Komputasi untuk Mendapatkan Nilai v

No	Yang Dibutuhkan	Komputasi	
1	Nilai hash (H(M))	**M → f(H) → H(M)**	
2	Nilai r (hasil *signing*)	**$(g^k$ mod p) mod q**	
3	Nilai s (hasil *signing*)	**$(k^{-1}$ (H(M) + xr)) mod q**	**$((g^{u1} y^{u2})$ mod p) mod q** u1 = (H(M') w) mod q u2 = (r') w mod q
4	Nilai p (komponen *Global public-key*)	Bilangan prima **$2^{(L-1)} < p < 2^L$** L = kelipatan 64 512 ≤ L ≤ 1024	
5	Nilai q (komponen *Global public-key*)	Pembagi utama dari (p-1) **$2^{159} < q < 2^{160}$**	
6	Nilai g (komponen *Global public-key*)	**$h^{(p-1)/q}$ mod p** 1 < h < (p-1) g > 1	
7	*Public-key* (y)	**g^x mod p**	
8	Nilai w	**$(s')^{-1}$ mod q**	

2. *Compare* Nilai v dan Nilai r

Proses ini adalah proses terakhir dari *verifying*. Pada proses ini nilai v akan dibandingkan dengan nilai r. Jika nilai v sama dengan nilai r, maka dokumen tersebut adalah benar milik dari seseorang yang membuat file tersebut. Namun jika nilai v tidak sama dengan nilai r, berarti menunjukkan bahwa dokumen tersebut telah diubah oleh orang lain.

Tabel 4. Komputasi dari Nilai v dan Nilai r

No	Yang Dibutuhkan	Komputasi
1	Nilai r	**$(g^k$ mod p) mod q**

No	Yang Dibutuhkan	Komputasi
2	Nilai v	$((g^{u1} y^{u2}) \bmod p) \bmod q$ u1 = (H(M') w) mod q u2 = (r') w mod q

4. VISUALISASI ALGORITMA *DIGITAL SIGNATURE*

Dari analasisis algoritma yang sudah dilakukan, penulis akan memvisualisasikan algoritma tersebut agar nantinya mudah di pelajari oleh pengguna.

4.1 Proses *Signing*

Pada proses *signing* ini akan divisualisasikan tahap demi tahap dari memasukkan file yang akan di-*signing* sampai mendapatkan hasil *signing* yaitu nilai r dan nilai s. Tahapan dari proses *signing* adalah sebagai berikut:

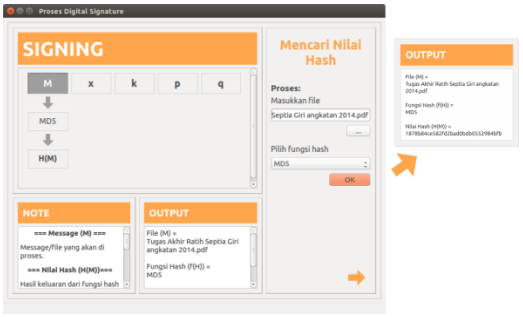
1. Mencari Nilai Hash.
2. Mencari Bilangan Prima.
3. Mencari Pembagi Utama.
4. Mencari Random atau Pseudorandom Integer.
5. Mencari Nilai g.
6. Mencari Nilai r.
7. Mencari Nilai s.

Dari tahapan-tahapan di atas, akan dijelaskan secara singkat penjelasan beserta visualisasi dari proses *signing* yang telah dibuat oleh penulis.

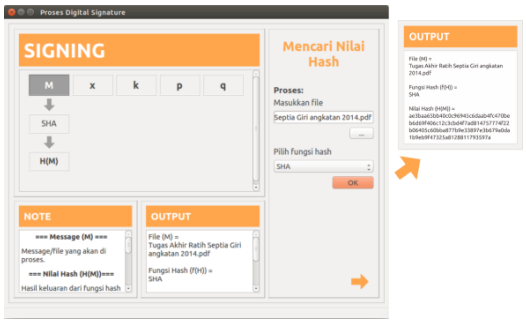
1. Mencari Nilai Hash

   Pada tahapan ini pengguna dapat memasukkan file berformat pdf dan dapat memilih fungsi hash yang diinginkan untuk memproses file tersebut. Ketika tombol "OK" ditekan, maka file tersebut akan di proses menjadi nilai hash dengan panjang tetap sesuai dengan fungsi hash yang pengguna pilih. Jika fungsi hash yangdipilih adalah MD5, maka nilai hash yang dihasilkan berupa string dengan panjang 128 bit. Jika fungsi hash yang dipilih adalah SHA, maka nilai hash yang dihasilkan berupa string dengan panjang 160 bit.

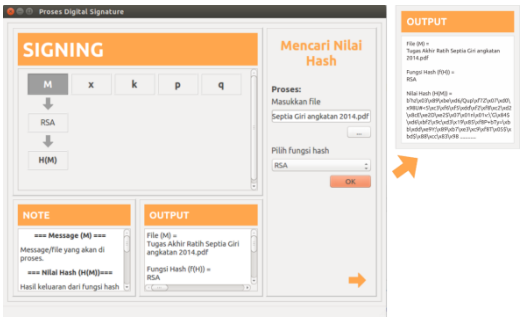
Gambar 8. Tampilan Pertama Sebelum Diproses



Gambar 9. Tampilan Setelah Diproses menggunakan MDS



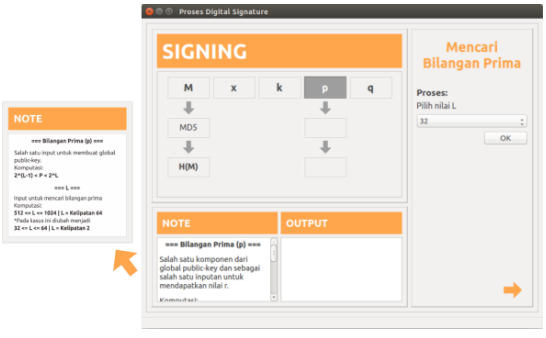
Gambar 10. Tampilan Setelah Diproses menggunakan SHA



Gambar 11. Tampilan Setelah Diproses menggunaakn SHA

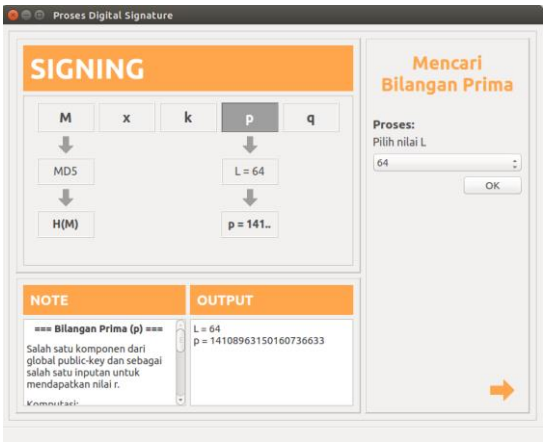
2. Mencari Bilangan Prima

   Pada tahap ini akan mencari salah satu komponen *global public-key* yaitu nilai p. Nilai p akan dipilih secara *random*, dimana nilai p adalah bilangan prima dan $2(L-1) < p < 2L$. Pada algoritma yang semestinya, nilai L adalah kelipatan 64 dimana $512 \leqslant L \leqslant 1024$. Namun, pada sistem ini nilai L diperkecil agar komputasi dapat dilakukan secara cepat. Pengguna dapat memilih nilai L dengan 32 ataupun 64.



Gambar 12. Tampilan Kedua Sebelum Diproses

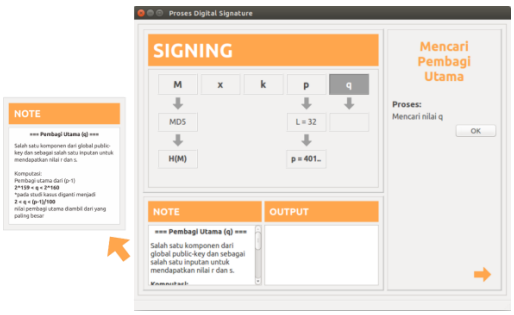
Gambar 13. Tampilan Setelah Diproses dengan L = 32



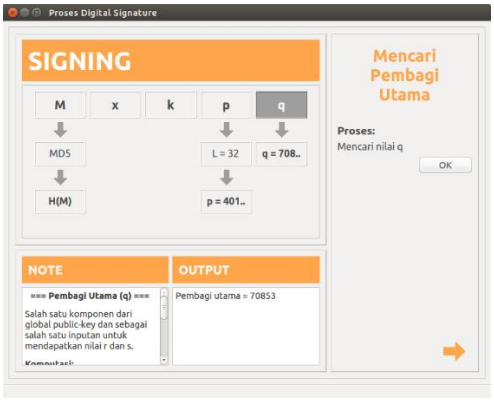
Gambar 14. Tampilan Setelah Diproses dengan L = 64

3. Mencari Pembagi Utama

Pada tahap ini akan mencari salah satu komponen *global public-key* yaitu nilai q. Nilai q adalah pembagi utama dari (p-1) dimana nilai q adalah bilangan prima dan 2159 < q < 2160. Namun pada sistem ini karna nilai L diperkecil maka nilai q tidak akan mencapai 2159 sehingga batas tersebut dihapuskan, pada sistem ini nilai q akan diambil dari pembagi yang paling besar.



Gambar 15. Tampilan Ketiga Sebelum Diproses

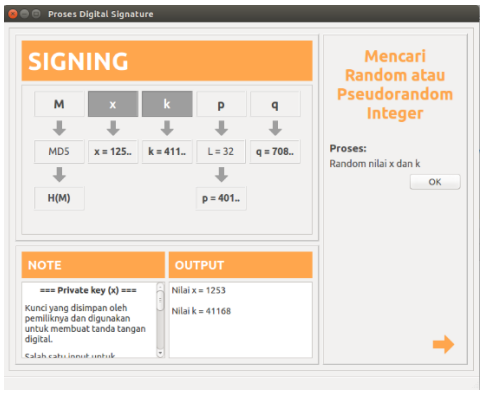


Gambar 16. Tampilan Ketiga Setelah Diproses

4. Mencari Random atau Pseudorandom Integer

Pada tahap ini akan mencari nilai x sebagai private-keydan nilai k sebagai nilai rahasia perdokumen, dimana nilai x dan nilai k adalah *random* atau *pseudorandominteger* antara 0 dan nilai q.

Gambar 17. Tampilan Keempat Sebelum Diproses

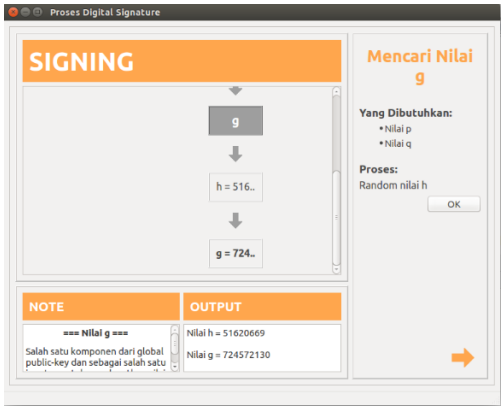


Gambar 18. Tampilan Keempat Setelah Diproses

5. Mencari Nilai g

Pada tahap ini akan mencari salah satu komponen *global public-key* yaitu nilai g. Nilai g didapatkan dari h(p-1)/q mod p dimana 1 < h < (p-1).

Gambar 19. Tampilan Kelima Sebelum Diproses

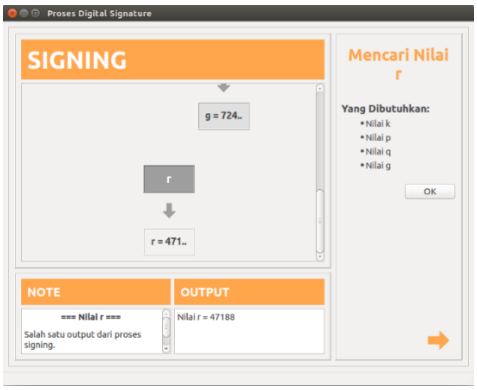


Gambar 20. Tampilan Kelima Setelah Diproses

6. Mencari Nilai r

Pada tahap ini akan mencari salah satu dari *output* proses *signing* yaitu nilai r, dimana nilai r didapatkan dari (gk mod p) mod q.

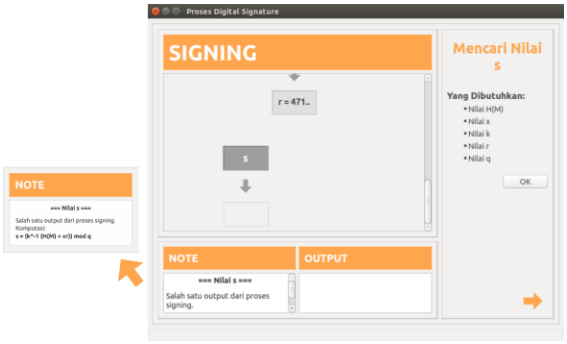
Gambar 21. Tampilan Keenam Sebelum Diproses



Gambar 22. Tampilan Keenam Setelah Diproses

7. Mencari Nilai s

Pada tahap ini akan mencari salah satu dari *output* proses *signing* yaitu nilai s, dimana nilai s didapatkan dari (k-1 (H(M) + xr)) mod q.

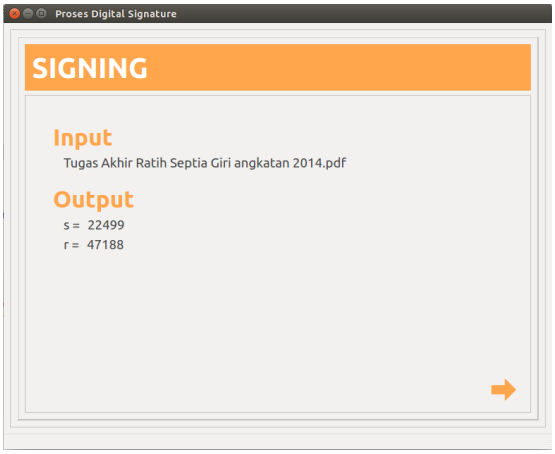


Gambar 23. Tampilan Ketujuh Sebelum Diproses

Gambar 24. Tampilan Ketujuh Setelah Diproses

8. Hasil Proses *Signing*

Pada tahap ini akan ditampilkan input file yang di proses dan hasil dari *signing* file tersebut yaitu nilai r dan nilai s.

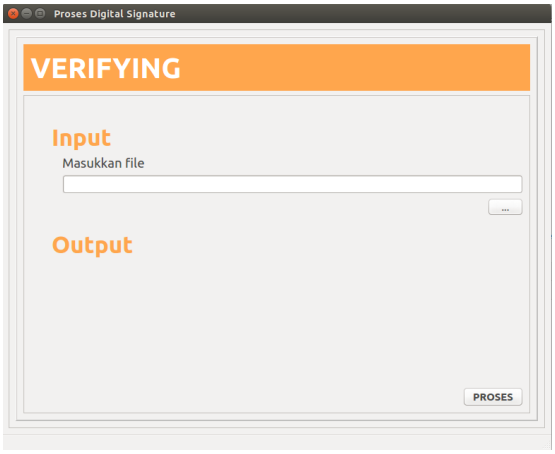


Gambar 25. Hasil dari Proses *Signing*

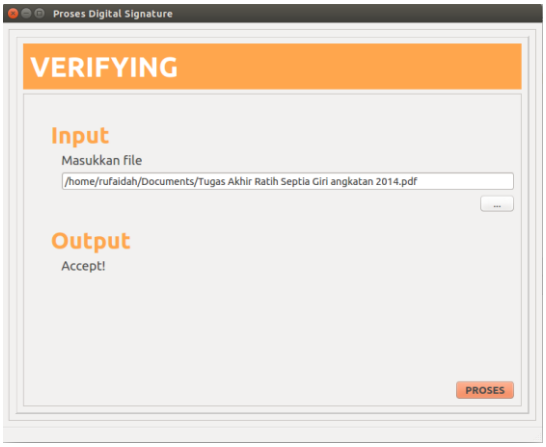
4.2 Proses *Verifying*

Pada proses *verifying* ini tidak divisualisasikan secara detail seperti proses *signing*. Pada tahap ini pengguna

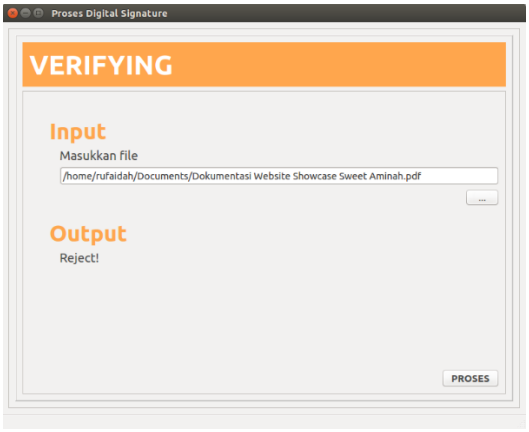
hanya dapat memasukkan file yang akan di *verify*, kemudian file tersebut akan diproses sehingga mendapatkan nilai v. Nilai v didapatkan dari ((gu1 yu2) mod p) mod q dimana u1 = (H(M') w) mod q dan u2 = (r') w mod q. Kemudian nilai v akan dibandingkan dengan nilai r, jika nilai v sama dengan nilai r maka akan tampil tulisan "*Accept!*" yang berarti file tersebut adalah benar file yang sudah di-*signing* dan tidak diubah isinya oleh orang lain, dan jika nilai v tidak sama dengan nilai r maka akan tampil tulisan "*Reject!*" yang berarti file tersebut berbeda dari file yang sudah di *signing* ataupun isi file tersebut sudah diubah oleh orang lain.



Gambar 26. Tampilan Awal Proses *Verifying*



Gambar 27. Tampilan Setelah Dimasukkan File yang Sama



Gambar 28. Tampilan Setelah Dimasukkan File yang Sama

5. KESIMPULAN

Sistem visualisasi dari algoritma *digital signature* ini telah dicoba oleh tiga narasumber dan kemudian narasumber menyatakan pendapatnya mengenai sistem ini. Mereka menyatakan bahwasanya alur sudah jelas namun tampilan terlalu kaku, dan juga kurang informatif.

Oleh karena itu dapat disimpulkan bahwasanya sistem ini tidak cocok untuk digunakan oleh orang awam, seseorang harus mempunyai ilmu mengenai *digital signature* terlebih dahulu untuk bisa memahami sistem ini.

Untuk penelitian kedepannya diharapkan dapat menambahkan informasi lebih banyak sehingga lebih banyak ilmu yang dapat di ambil oleh pengguna.

DAFTAR PUSTAKA

[1] W. Stallings, "*Cryptography and Network Security Principles and Practice*," pp. 9-11, United States of America, 2011.

[2] M. K. Islam, M. Hossain, M. Nashiry, "*Security of Cryptographic Algorithm SHA and MD5*," p. 28, 2012

[3] N. Herawati, R. R. Isnanto, A. Fatchurrohim, "Perancangan dan Implementasi DSA (*Digital Signature Algorithm*) menggunakan Bahasa Pemograman Java," pp. 1-2, 2008

[4] W. Stallings, "*Cryptography and Network Security Principles and Practice*," p. 406, United Stated of America, 2011.

[5] W. Stallings, "*Cryptography and Network Security Principles and Practice*," pp. 396-398, United Stated of America, 2011.