JIKUM
JURNAL ILMU KOMPUTER

E-ISSN: 9999-9999
Volume 1, **Number** 1, Mei 2025
https://doi.org/10.62671/jikum.vxix.xxxx

Implementasi Algoritma Asimetris DSA (Digital Signature Algorithm) dalam Aplikasi Sederhana untuk Tanda Tangan Digital

Aulia Fitri[1], Izza Abdullah[2], Tegar Ramada[3], Rosita Saputri[4], Juwita Saharani[5], M. Frizki Gultom[6]
[1,2,3,4,5,6]Universitas Malikussaleh, Indonesia
[1]aulia.220180016@mhs.unimal.ac.id, [2]izza.220180022@mhs.unimal.ac.id, [3]tegar.220180023@mhs.unimal.ac.id,
[4]rosita.220180025@mhs.unimal.ac.id, [5]juwita.220180027@mhs.unimal.ac.id, [6]frizky.220180157@mhs.unimal.ac.id

ABSTRACT

Seiring meningkatnya kebutuhan akan keamanan informasi di era digital, tanda tangan digital menjadi solusi penting dalam menjamin keaslian dan integritas data. Penelitian ini bertujuan untuk mengimplementasikan algoritma Digital Signature Algorithm (DSA) dalam bentuk aplikasi sederhana berbasis Python. Dengan pendekatan eksperimen terapan dan metode prototyping, aplikasi ini dikembangkan menggunakan pustaka kriptografi seperti PyCryptodome, hashlib, dan Tkinter. Aplikasi yang dihasilkan memungkinkan proses penandatanganan dan verifikasi pesan secara efisien. Hasil implementasi menunjukkan bahwa sistem mampu mendeteksi setiap perubahan pada pesan dan menjamin validitasnya, meskipun masih terdapat keterbatasan pada format pesan yang didukung dan ketergantungan terhadap file kunci eksternal. Penelitian ini memberikan pemahaman praktis tentang penerapan kriptografi asimetris dalam pengamanan data digital.

Kata Kunci: *Tanda Tangan Digital, Kriptografi Asimetris, DSA, Python, Keamanan Informasi*

PENDAHULUAN

Seiring dengan perkembangan teknologi informasi, pada tahun 1990 keamanan informasi menjadi bahan pembicaraan bagi banyak kalangan antaranya pemerintah, bisnis komersial, dan individu. Informasi disimpan dalam bentuk elektronik karena medium ini lebih sederhana, ukurannya kompak, dan melayani transfer data yang cepat. Namun dengan terjadinya revolusi elektronik maka informasi menghadapi masalah yang serius yaitu keamanan informasi pada proses komunikasi. Proses komunikasi sendiri melibatkan dua pihak yaitu pihak pengirim (sender) dan penerima (receiver). Tentunya, yang dikirim adalah informasi atau pesan yang hanya boleh diketahui oleh kedua belah pihak. Namun jika pihak ketiga menyadap dan memodifikasi pesan atau berpurapura sebagai pengirim asli tentunya akan sangat merugikan. Selain itu risiko terhadap manipulasi, pencurian, dan penyalahgunaan data juga semakin meningkat.Salah satu solusi penting dalam menjaga integritas dan keaslian data adalah melalui penggunaan Tanda Tangan Digital. Tanda tangan digital berfungsi sebagai bukti keaslian sebuah pesan atau dokumen elektronik serta menjamin bahwa pesan tersebut belum mengalami perubahan sejak ditandatangani. Teknologi ini bergantung pada konsep kriptografi asimetris, di mana sepasang kunci yaitu private key dan public key digunakan dalam proses penandatanganan dan verifikasi (Herawati, 2011).

Salah satu algoritma yang digunakan dalam penerapan tanda tangan digital adalah Digital Signature Algorithm (DSA). DSA merupakan algoritma kriptografi asimetris yang dirancang khusus untuk membuat dan memverifikasi tanda tangan digital(Alfani et al., 2024). Keamanan DSA didasarkan pada kesulitan dalam memecahkan masalah logaritma diskret, menjadikannya pilihan yang andal dalam pengamanan data.

Tujuan kami menulis ini ingin mengimplementasikan algoritma DSA dalam bentuk aplikasi sederhana yang mampu melakukan proses penandatanganan dan verifikasi pesan. Dengan implementasi ini, diharapkan dapat memberikan pemahaman praktis mengenai cara kerja tanda tangan digital dan pentingnya dalam menjaga keamanan komunikasi digital.

METODE PENELITIAN

Metode penelitian ini menggunakan pendekatan eksperimen terapan (applied experimental research), di mana fokus utamanya adalah pada penerapan langsung algoritma Digital Signature Algorithm (DSA) dalam sebuah aplikasi yang dikembangkan secara mandiri. Tujuannya adalah untuk membuktikan bagaimana konsep kriptografi asimetris dapat diimplementasikan secara nyata dalam sistem digital guna menjamin keaslian dan integritas data.

Metode ini dipilih karena sesuai dengan tujuan penelitian, yaitu untuk menguji keefektifan tanda tangan digital berbasis DSA melalui simulasi sistem yang mampu melakukan proses penandatanganan dan verifikasi terhadap pesan. Proses pengembangan dilakukan dengan pendekatan prototyping, yaitu membangun sistem secara bertahap, dimulai dari perancangan, pembuatan fitur utama, hingga pengujian fungsionalitas program.

JIKUM
JURNAL ILMU KOMPUTER
E-ISSN: 9999-9999
Volume 1, **Number** 1, Mei 2025
https://doi.org/10.62671/jikum.vxix.xxxx

Alat dan Bahan

Dalam pelaksanaan penelitian ini, digunakan beberapa komponen perangkat keras dan perangkat lunak yang mendukung proses pengembangan aplikasi tanda tangan digital. Adapun rincian alat dan bahan yang digunakan adalah sebagai berikut:

Perangkat Keras

Penelitian dijalankan menggunakan laptop dengan spesifikasi minimum prosesor Intel Core i3, memori RAM 4 GB, serta ruang penyimpanan yang mencukupi.

- Perangkat Lunak
- Sistem Operasi: Windows 10
- Bahasa Pemrograman: Python versi 3.10
- Lingkungan Pengembangan: Visual Studio Code Pustaka

Tambahan:

- PyCryptodome, untuk pembangkitan kunci, tanda tangan, dan proses verifikasi menggunakan algoritma DSA (Alfani et al., 2024).
- Tkinter, digunakan dalam pembuatan antarmuka pengguna grafis (GUI).
- hashlib, untuk melakukan proses hashing terhadap pesan menggunakan algoritma SHA-256.
- json dan base64, digunakan untuk menyimpan tanda tangan dalam format teks dan proses encoding/decoding data.
- Python dipilih karena bersifat open-source, fleksibel, dan menyediakan berbagai pustaka kriptografi yang mendukung proses pengembangan sistem keamanan digital secara efisien dan praktis (Munir, 2005).

HASIL DAN PEMBAHASAN

Implementasi dan Hasil Tanda Tangan Digital Menggunakan DSA

Setelah melalui tahap perancangan dan implementasi program tanda tangan digital menggunakan algoritma DSA (Digital Signature Algorithm), sistem berhasil dijalankan dengan antarmuka berbasis GUI yang sederhana dan interaktif. Program ini dibuat menggunakan bahasa pemrograman Python dengan bantuan *library tkinter*, *cryptography*, dan *hashlib*.

Berikut adalah tampilan antarmuka (GUI) saat program dijalankan:

Gamabar 1. Tampilan Antarmuka

Gambar di atas memperlihatkan bahwa pengguna dapat memasukkan pesan teks ke dalam field input, lalu menandatanganinya dengan tombol Sign Message, dan juga dapat memverifikasi tanda tangan tersebut dengan menekan tombol *Verify Signature.*

Penjelasan Kode Program

Kode program yang digunakan terdiri dari dua bagian utama: fungsi-fungsi backend (logika DSA) dan antarmuka pengguna (GUI) dengan *Tkinter*.

a. Pemrosesan Kunci dan Tanda Tangan Digital

```
def load_dsa_keys():
    with open("private_key.pem", "rb") as f:
        private_key = f.read()
    with open("public_key.pem", "rb") as f:
        public_key = f.read()
    return private_key, public_key
```

Gambar 2. Kode program pemrosesan kunci

JIKUM
JURNAL ILMU KOMPUTER
E-ISSN: 9999-9999
Volume 1, **Number** 1, Mei 2025
https://doi.org/10.62671/jikum.vxix.xxxx

Fungsi ini memuat kunci dari file private_key.pem dan public_key.pem. File ini penting karena tanpa kunci tersebut, proses tanda tangan dan verifikasi tidak dapat dilakukan

```
def sign_dsa(message, private_key_bytes):
    key = DSA.import_key(private_key_bytes)
    hash_obj = SHA256.new(message.encode('utf-8'))
    signer = DSS.new(key, 'fips-186-3')
    signature = signer.sign(hash_obj)
    return base64.b64encode(signature).decode('utf-8')
```

Fungsi ini melakukan penandatanganan pesan:
- Mengubah pesan menjadi hash menggunakan SHA256.
- Menggunakan private key untuk menandatangani hash tersebut.

```
def verify_dsa(message, signature_b64, public_key_bytes):
    key = DSA.import_key(public_key_bytes)
    hash_obj = SHA256.new(message.encode('utf-8'))
    verifier = DSS.new(key, 'fips-186-3')
    try:
        verifier.verify(hash_obj, base64.b64decode(signature_b64))
        return True
    except ValueError:
        return False
```

Fungsi ini melakukan verifikasi tanda tangan:
- Pesan di-hash kembali.
- Tanda tangan yang dikodekan dalam Base64 di-decode.
- Kemudian diverifikasi menggunakan public key. Jika cocok, maka pesan valid.

b.Fungsi Penyimpanan dan Pengambilan Data

```
def save_signature(message, signature):
    data = {"message": message, "signature": signature}
    with open("signature.json", "w") as file:
        json.dump(data, file)
```

Menyimpan pesan dan tanda tangan ke dalam file signature.json.

```
def load_signature():
    try:
        with open("signature.json", "r") as file:
            data = json.load(file)
        return data["message"], data["signature"]
    except FileNotFoundError:
        messagebox.showerror("Error", "No signature found. Sign a message first!")
        return None, None
```

Membaca kembali pesan dan tanda tangan dari file signature.json. Jika file tidak ditemukan, akan muncul error.

JIKUM
JURNAL ILMU KOMPUTER

E-ISSN: 9999-9999
Volume 1, **Number** 1, Mei 2025
https://doi.org/10.62671/jikum.vxix.xxxx

c. Antarmuka Grafis dengan Tkinter

```
def sign_message():
    message = entry_message.get()
    if not message:
        messagebox.showerror("Error", "Please enter a message.")
        return
    private_key, _ = load_dsa_keys()
    signature = sign_dsa(message, private_key)
    save_signature(message, signature)
    messagebox.showinfo("Success", "Message signed successfully!")
```

Fungsi ini terhubung dengan tombol "**Sign Message**" di GUI. Ia mengambil input dari user, lalu menjalankan proses sign_dsa, menyimpan hasilnya, dan menampilkan pop-up notifikasi bahwa proses berhasil.

```
def (variable) input_message: str
    input_message = entry_message.get()
    if not input_message:
        messagebox.showerror("Error", "Please enter a message to verify.")
        return

    _, signature = load_signature()
    if signature is None:
        return

    _, public_key = load_dsa_keys()
    valid = verify_dsa(input_message, signature, public_key)

    if valid:
        messagebox.showinfo("Verification", "Signature is valid for the input message!")
    else:
        messagebox.showerror("Verification", "Signature is invalid for the input message!")
```

Fungsi ini terhubung dengan tombol "**Verify Signature**". Ia mengambil pesan yang diketik, memuat tanda tangan dari file JSON, dan melakukan proses verifikasi. Hasilnya ditampilkan dalam bentuk pop-up.

```
root = tk.Tk()
root.title("DSA Digital Signature")

tk.Label(root, text="Enter Message:").pack()
entry_message = tk.Entry(root, width=50)
entry_message.pack(pady=5)

tk.Button(root, text="Sign Message", command=sign_message).pack(pady=2)
tk.Button(root, text="Verify Signature", command=verify_message).pack(pady=2)


root.mainloop()
```

Kode ini adalah bagian utama GUI. Di sini antarmuka dirancang sederhana: label, input field, dan dua tombol. Seluruh sistem berjalan dalam jendela GUI yang dibuka dengan root.mainloop().

JIKUM
JURNAL ILMU KOMPUTER

E-ISSN: 9999-9999
Volume 1, **Number** 1, Mei 2025
https://doi.org/10.62671/jikum.vxix.xxxx

Kelebihan dan Kelemahan Sistem

Dari hasil pengujian dan pengamatan kode program, terdapat beberapa kelebihan yang dapat disimpulkan. Aplikasi ini mampu menjalankan proses tanda tangan digital dan verifikasi dengan akurasi tinggi dan waktu proses yang cepat. Setiap perubahan pada pesan langsung terdeteksi, menunjukkan bahwa sistem menjamin integritas data. Selain itu, antarmuka pengguna cukup sederhana dan mudah digunakan.

Namun, aplikasi ini juga memiliki beberapa kekurangan. Pertama, proses hanya terbatas pada pesan teks biasa, belum mendukung format dokumen lain seperti PDF atau Word. Kedua, pengguna harus memastikan file kunci (private_key.pem dan public_key.pem) tersedia di direktori yang sama, karena sistem tidak menyediakan fitur pembuatan kunci langsung dari GUI. Terakhir, file signature.json berisi data penting yang bisa diakses atau dimanipulasi jika tidak diberi proteksi tambahan.

KESIMPULAN

DSA merupakan algoritma kriptografi asimetris yang cukup efektif dalam menjaga keaslian dan integritas pesan digital. Melalui aplikasi sederhana yang dikembangkan, proses penandatanganan dan verifikasi pesan dapat berjalan dengan cepat dan akurat. Sistem ini juga mampu mendeteksi perubahan pesan secara langsung, sehingga sangat membantu dalam menjaga keamanan data.Namun, penerapan DSA ini masih memiliki beberapa keterbatasan, terutama pada jenis pesan yang didukung dan ketergantungan pada file kunci eksternal. Oleh karena itu, pengembangan lebih lanjut perlu dilakukan, seperti menambahkan dukungan untuk format dokumen lain dan integrasi fitur pembuatan kunci langsung dalam aplikasi. Dengan perbaikan ini, DSA dapat menjadi bagian penting dalam sistem keamanan digital yang praktis dan andal di masa mendatang.

REFERENSI

Alfani, M. R., Furqan, M., & Nasution, Y. R. (2024). Pengamanan Data Teks Menggunakan Metode Digital Signature Algorithm (Dsa) Dan Advanced Encryption Standard (Aes). *Journal of Science and Social Research*, *4307*(1), 301–306.

Amaliawati, Y. H., Pratama, A. A. R. P., & Saputro, I. A. (2024, December). Analisis Perbandingan Kinerja Algoritma Kriptografi Message Digest Algorithm 5 (MD5) dan Digital Signature Algorithm (DSA) Berdasarkan Ukuran File dalam Proses Enkripsi File. In *Prosiding Seminar Nasional Amikom Surakarta* (Vol. 2, pp. 1176-1186).

Ashari, W. M. (2020). Perbandingan Performa Kriptografi Asimetris Pada Proses Key Exchange. *Science Tech: Jurnal Ilmu Pengetahuan dan Teknologi*, *6*(1), 26-32.

Ashari, W. M. (2020). Perbandingan Performa Kriptografi Asimetris Pada Proses Key Exchange. *Science Tech: Jurnal Ilmu Pengetahuan dan Teknologi*, *6*(1), 26-32.

Haekal, M. M. (2022). *Tanda Tangan Digital: Keabsahan, Fungsi, dan Cara Kerjanya*. Mekarisign.Com. https://mekarisign.com/id/blog/tanda-tangan-digital- adalah/

Herawati, Nora, R. Rizal Isnanto, and Adian Fatchur Rochim. Perancangan dan implementasi dsa (digital signature algorithm) menggunakan bahasa pemrograman java. Diss. Jurusan Teknik Elektro Fakultas Teknik Undip, 2011.

Mauluddin, A., Setiawan, A., & Supriadi, I. (2022). IP Prototype Single IP Login dan Kriptografi Asimetris Digital Signature Algorithm pada User Auntentification Studi Kasus CV. Danofal's.

Munir, R. (2005). *Penggunaan Tanda-Tangan Digital untuk Menjaga Integritas Berkas Perangkat Lunak*. *2005*(Snati), 6–9.

Pardosi, I. A., & Purba, R. (2015). Pengembangan Web E-Voting Menggunakan Secure Election Protocol. *Jurnal SIFO Mikroskil*, *16*(1), 73–82.

Simetris, A. (2016). Kriptografi Simetris dan Asimetris dalam Perspektif Keamanan Data dan Kompleksitas Komputasi. *Jurnal Ilmiah Ilmu Komputer*, *2*(2).