

IMPLEMENTASI TANDA TANGAN DIGITAL MENGGUNAKAN RSA-PSS DAN SHA-256 UNTUK KEAMANAN DOKUMEN ELEKTRONIK

Sahrani Indah Sari¹, Mahardika Abdi Prawira Tanjung²

^{1,2}Universitas Muhammadiyah Sumatera Utara, Indonesia

Email: sahraniindahh26@gmail.com¹, mahardikaabdiprawira@umsu.ac.id²

ABSTRAK

Semakin banyak instansi memindahkan pengelolaan dokumen ke bentuk elektronik, dan pergeseran ini menuntut cara baru untuk membuktikan bahwa sebuah dokumen memang asli dan belum dimodifikasi. Penelitian ini membangun sistem tanda tangan digital berbasis RSA-PSS dan SHA-256 untuk dokumen PDF, dengan QR Code sebagai penunjuk ke metadata verifikasi. SHA-256 dipakai untuk menghitung message digest dari seluruh byte stream PDF final, RSA-PSS dipakai untuk membentuk dan memverifikasi tanda tangan lewat pasangan kunci RSA 2048-bit. React.js menangani frontend, FastAPI (Python) menangani backend, dan PostgreSQL menyimpan datanya. Pengujian menggunakan metode Black Box Testing terhadap 20 dokumen PDF, yang terdiri atas 10 dokumen asli dan 10 dokumen yang telah dimodifikasi. Semua dokumen asli lolos verifikasi sebagai VALID. Semua dokumen yang dimodifikasi terdeteksi INVALID, termasuk dokumen dengan perubahan isi dan QR Code. Pengujian RSA-PSS juga menunjukkan bahwa tanda tangan ditolak ketika diverifikasi menggunakan kunci publik yang tidak sesuai. Temuan ini menunjukkan bahwa kombinasi SHA-256, RSA-PSS, dan QR Code berbasis referensi mampu memeriksa integritas dokumen PDF dan memverifikasi tanda tangan berdasarkan kunci publik penandatanganan yang terdaftar pada sistem. Kata Kunci: Tanda Tangan Digital, RSA-PSS, SHA-256, QR Code, Dokumen Elektronik.

ABSTRACT

As more institutions transition document management to an electronic format, this change requires new ways to demonstrate that a document is authentic and has not been altered. This research develops a digital signature system based on RSA-PSS and SHA-256 for PDF documents, with a QR code serving as an indicator of verification metadata. SHA-256 is used to compute the message hash of the entire byte sequence of the final PDF, and RSA-PSS is used to create and verify the signature through a pair of 2048-bit RSA keys. React.js handles the frontend, FastAPI (Python) manages the backend, and PostgreSQL stores the data. Testing was conducted using the Black Box Testing method on 20 PDF documents, consisting of 10 original documents and 10 documents that have been modified. All original documents passed verification as VALID. All modified documents were detected as INVALID, including those with changes in content and the QR code. RSA-PSS testing also showed that the signature was rejected when verified using a public key that did not correspond. These findings indicate that the combination of SHA-256, RSA-PSS, and a reference-based QR code is capable of verifying the integrity of PDF documents and validating the signature based on the public key of the signer registered in the system. Keywords: Digital Signature, RSA-PSS, SHA-256, QR Code, Electronic Document

PENDAHULUAN

Perkembangan teknologi informasi mendorong peralihan pengelolaan dokumen dari bentuk fisik menuju dokumen elektronik. Penyimpanan dan penyebaran dokumen digital menjadi lebih mudah, tetapi kondisi tersebut juga meningkatkan kebutuhan terhadap mekanisme yang dapat memastikan keaslian dan integritas dokumen. Dokumen digital dapat disalin dan didistribusikan dengan mudah, sehingga tanda tangan berupa gambar atau hasil pemindaian belum dapat membuktikan bahwa dokumen tetap sama seperti saat ditandatangani (Kazmierczyk & Turner, 2022).

Istilah "tanda tangan elektronik" memiliki cakupan yang luas. Berbeda dari tanda tangan elektronik pada umumnya, tanda tangan digital secara khusus mengandalkan pasangan kunci asimetris. Kunci privat dipakai penandatangan untuk membentuk signature, sementara pihak penerima memakai kunci publik pasangannya untuk memastikan keabsahan signature tersebut. Fungsi hash berperan menghasilkan representasi tetap dari data yang ditandatangani, sehingga perubahan sekecil apa pun pada dokumen akan terdeteksi saat proses verifikasi berjalan (Melina et al., 2022).

RSA sudah digunakan beberapa peneliti dengan cara yang berbeda-beda. Aryasanti et al. membangun tanda tangan digital berbasis web memakai RSA dan SHA-512 (Aryasanti et al., 2022). Sementara Sarasvananda & Iswara memilih kombinasi RSA dan *QR Code* untuk sistem surat-menyurat (Sarasvananda & Iswara, 2022). Untuk lembar pengesahan skripsi, Akbar et al. menerapkan RSA pada proses verifikasi tanda tangan digital (Akbar et al., 2021). Hutagalung et al. mengamankan Surat Keterangan Lulus lewat kombinasi SHA-256 dan RSA, dengan *QR Code* masuk sebagai bagian dari verifikasi (Hutagalung et al., 2023). Pendekatan serupa dipakai Barus et al. pada dokumen Surat Perintah Perjalanan Dinas (Barus et al., 2025). Nurjaman dan Rizal mengambil jalur berbeda yaitu SHA3-512 dipadukan dengan RSA pada sistem penyimpanan berkas digital yang juga didukung *QR Code* (Nurjaman & Rizal, 2024). Di sisi lain, Natho et al. membandingkan efektivitas algoritma RSA dengan HMAC dalam pembentukan tanda tangan digital, dan menemukan perbedaan karakteristik keamanan antara kedua pendekatan tersebut (Natho et al., 2025). Penelitian-penelitian tersebut menunjukkan penerapan kriptografi untuk pengamanan dokumen, tetapi belum secara khusus menempatkan RSA-PSS sebagai skema padding serta mengikat proses tanda tangan pada byte stream PDF final setelah *QR Code* ditempatkan.

Berdasarkan kondisi tersebut, penelitian ini mengimplementasikan SHA-256 dan RSA-PSS 2048-bit pada PDF final, yaitu dokumen setelah seluruh elemen yang menjadi bagian dari dokumen, termasuk *QR Code*, ditempatkan. Seluruh byte stream PDF final digunakan sebagai masukan SHA-256 sebelum proses pembentukan signature RSA-PSS. *QR Code* diterapkan secara reference-based sebagai penghubung antara dokumen dan metadata penandatanganan yang tersimpan pada basis data, bukan sebagai dasar utama penentuan validitas tanda tangan. Pendekatan ini

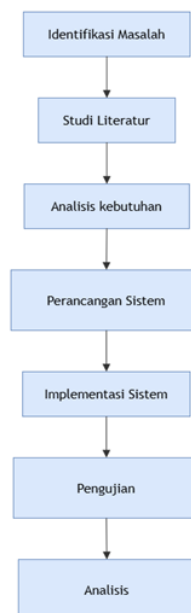
memungkinkan perubahan terhadap isi maupun QR Code mengubah byte stream dokumen sehingga dapat terdeteksi pada proses verifikasi.

Tujuan penelitian ini yaitu membangun dan menguji penerapan SHA-256 serta RSA-PSS 2048-bit untuk penandatanganan dan verifikasi dokumen PDF, dengan QR Code sebagai referensi menuju metadata. Penelitian ini memberikan tiga kontribusi utama. Pertama, menerapkan RSA-PSS dan SHA-256 pada dokumen PDF final setelah seluruh elemen dokumen, termasuk QR Code, ditempatkan, sehingga perubahan terhadap byte stream dokumen dapat terdeteksi melalui proses verifikasi. Kedua, menerapkan QR Code berbasis referensi sebagai penghubung antara dokumen dan metadata penandatanganan tanpa menjadikan QR Code sebagai dasar utama penentuan validitas tanda tangan. Ketiga, mengevaluasi kemampuan mekanisme tersebut dalam mendeteksi berbagai perubahan pada dokumen PDF setelah proses signing, termasuk perubahan konten, elemen dokumen, dan QR Code.

METODE PENELITIAN

1 Tahapan Penelitian

Penelitian ini dilaksanakan melalui tahapan identifikasi masalah, studi literatur, perancangan, implementasi, pengujian, dan analisis hasil. Alur penelitian ditunjukkan pada Gambar 1.



Gambar 1. Tahapan Penelitian

Gambar 1 menunjukkan tujuh tahap penelitian, dari identifikasi masalah dan studi literatur hingga implementasi kriptografi (message hash, encoding EMSA-PSS, MGF1, salt acak, operasi RSA 2048-bit) pada aplikasi web, pengujian dengan dokumen PDF asli dan dimodifikasi, serta analisis hasil untuk memastikan sistem mendukung verifikasi integritas dan keaslian dokumen.

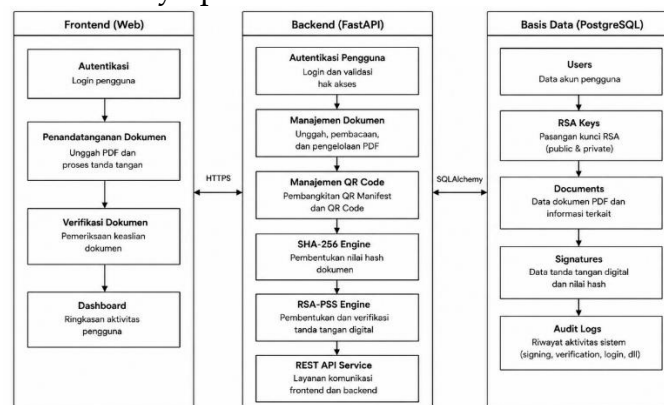
2 Arsitektur Sistem

Sistem menggunakan arsitektur client-server dengan React.js sebagai frontend, FastAPI sebagai backend, dan PostgreSQL sebagai basis data. Frontend menangani interaksi pengguna, sedangkan backend menjalankan proses hashing, pembentukan dan verifikasi tanda tangan RSA-PSS, serta pengelolaan metadata dokumen. Arsitektur sistem ditunjukkan pada Gambar 2, sedangkan komponen dan fungsinya dirangkum pada Tabel 1.

Tabel 1. Komponen implementasi sistem

Komponen	Teknologi	Fungsi
Frontend	React.js	Antarmuka pengguna untuk <i>signing</i> dan <i>verification</i>
Backend	FastAPI / Python	Logika aplikasi dan proses kriptografi
Kriptografi	Implementasi kriptografi pada source code	SHA-256, RSA 2048-bit, RSA-PSS, MGF1
PDF	PyMuPDF	Pembacaan, rendering, dan pengolahan PDF
QR Code	pyzbar dan OpenCV	Pembacaan QR Code
Database	PostgreSQL + SQLAlchemy	Penyimpanan pengguna, dokumen, kunci, signature, dan audit log

Setiap pengguna memiliki pasangan kunci RSA 2048-bit. Private key digunakan oleh backend untuk menghasilkan signature pada proses *signing*, sedangkan public key digunakan untuk memverifikasi signature pada proses *verification*. Sistem membatasi objek pemrosesannya pada dokumen berformat PDF.



Gambar 2. Arsitektur Sistem

Arsitektur sistem terdiri atas tiga lapisan yaitu Frontend, Backend, dan Basis Data. Frontend menangani interaksi pengguna. Backend menjalankan logika aplikasi, termasuk proses SHA-256 dan RSA-PSS. Basis Data menyimpan data pengguna, dokumen, kunci, dan metadata tanda tangan. Ketiga lapisan berkomunikasi lewat REST API.

3 Proses Penandatanganan

Proses penandatanganan dilakukan setelah pengguna mengunggah berkas PDF untuk memulai proses penandatanganan. Sistem menyusun metadata dokumen

dan QR Manifest, kemudian menyisipkan *QR Code* ke halaman terakhir dokumen. Dokumen dengan *QR Code* ini menjadi versi final dan masuk ke endpoint *signing*. Sistem menghitung seluruh byte stream dokumen final dengan SHA-256 untuk menghasilkan message hash. Hash ini menjadi masukan proses encoding EMSA-PSS. Sistem memproses encoded message dengan kunci privat RSA 2048-bit dan menghasilkan tanda tangan digital.

Secara konseptual, nilai hash dinyatakan sebagai:

$$mHash = SHA - 256(M) \quad (1)$$

Proses pembangkitan nilai hash dokumen (M) dilakukan dengan memanfaatkan fungsi hash kriptografi SHA-256 untuk menghasilkan ikhtisar pesan (mHash) berukuran 256 bit (National Institute of Standards and Technology, 2015). Perhitungan internal fungsi kompresi SHA-256 dijalankan sesuai dengan standar FIPS PUB 180-4 (Secure Hash Standard).

Dalam proses EMSA-PSS, Sistem menggabungkan message hash dengan salt acak, lalu memprosesnya untuk membentuk data block dan mask memakai MGF1, mengikuti spesifikasi RFC 8017 (K. Moriarty et al., 2016). Salt acak membuat setiap tanda tangan berbeda meski dokumen dan kunci sama. *hLen* menunjukkan panjang keluaran SHA-256, yaitu 32 byte. PS berfungsi sebagai padding nol. MGF1 menghasilkan mask. Byte 0xbc menutup encoded message sebagai trailer field EMSA-PSS. Secara ringkas:

$$M' = 0x00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ ||\ mHash\ ||\ salt \quad (2)$$

$$H = SHA - 256(M') \quad (3)$$

$$DB = PS\ ||\ 0x01\ ||\ salt \quad (4)$$

$$dbMask = MGF1(H, emLen - hLen - 1) \quad (5)$$

$$maskedDB = DB \oplus dbMask \quad (6)$$

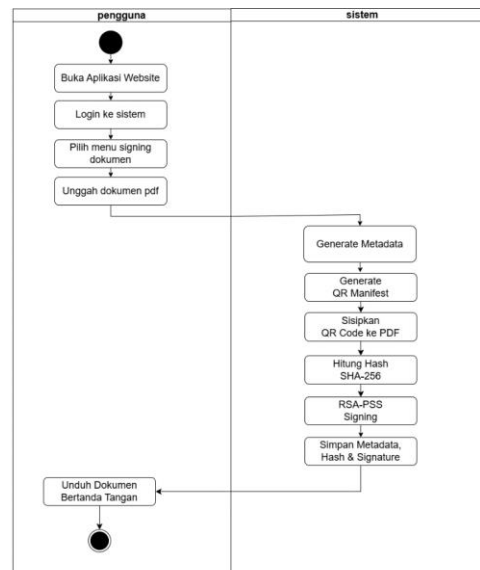
$$EM = maskedDB\ ||\ H\ ||\ 0xbc \quad (7)$$

Pada implementasi sistem ini, nilai salt dibangkitkan secara acak untuk setiap proses signing, sedangkan panjang keluaran SHA-256 yang digunakan sistem tetap konsisten pada 32 byte. Bagian padding nol disisipkan sistem untuk menyesuaikan panjang data block, sementara fungsi MGF1 dipanggil backend untuk membentuk mask yang menutupi data block tersebut. Byte penutup 0xbc ditambahkan sistem di akhir proses sebagai penanda bahwa encoded message telah selesai dibentuk sesuai struktur EMSA-PSS.

$$S = EM^d \bmod n \quad (8)$$

dengan S sebagai tanda tangan digital, d sebagai eksponen privat, dan n sebagai modulus RSA.

Tanda tangan beserta metadatanya disimpan ke basis data. QR Code hanya membawa rujukan menuju data tersebut, bukan signature RSA-PSS itu sendiri.



Gambar 3. Alur Penandatanganan Dokumen

Gambar 3 menggambarkan urutan interaksi pengguna dan sistem pada proses penandatanganan. Pengguna membuka aplikasi, login, memilih menu penandatanganan, dan mengunggah berkas PDF. Sistem membangkitkan metadata dan QR Manifest, menyisipkan *QR Code* ke dalam PDF, menghitung hash SHA-256, lalu membentuk tanda tangan digital lewat RSA-PSS. Data penandatanganan tersimpan di basis data, dan berkas PDF bertanda tangan siap diunduh pengguna.

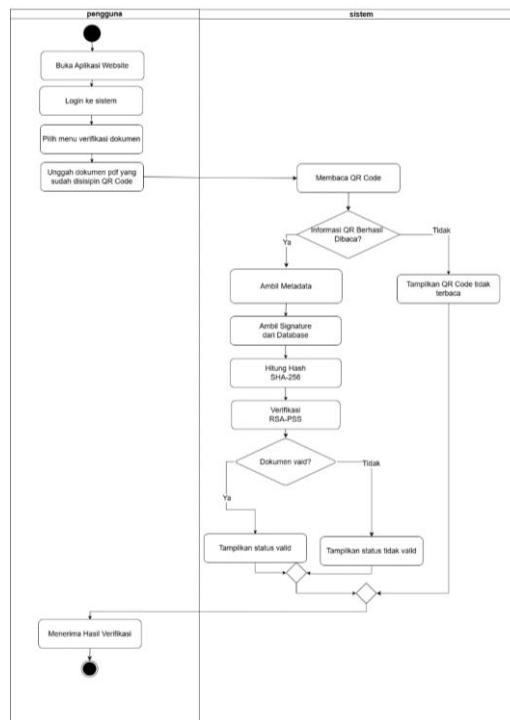
4 Proses Verifikasi

Sistem memulai verifikasi dengan membaca *QR Code* pada halaman terakhir dokumen. Metadata yang terbaca menjadi kunci rujukan untuk mengambil data penandatanganan dari basis data. Sistem membaca ulang seluruh byte stream dokumen yang diunggah dan menghitung SHA-256-nya. Sistem memakai nilai hash ini dalam proses verifikasi RSA-PSS bersama kunci publik penandatanganan. Secara konseptual, proses verifikasi RSA dapat dituliskan:

$$EM' = S^e \bmod n \quad (9)$$

dengan EM' sebagai encoded message yang diperoleh dari tanda tangan S menggunakan eksponen publik e .

Sistem menghitung EM' dari tanda tangan S menggunakan eksponen publik e , lalu memeriksa struktur EMSA-PSS dan mencocokkan hash hasil perhitungan ulang dengan hash saat penandatanganan. Dokumen dinyatakan VALID jika metadata sesuai, hash dokumen cocok, dan tanda tangan RSA-PSS terverifikasi dengan kunci publik pasangan yang benar. Jika salah satu syarat gagal, dokumen dinyatakan INVALID atau proses dihentikan sesuai jenis kegagalannya (K. Moriarty et al., 2016).



Gambar 4. Alur Verifikasi Dokumen

Gambar 4 memperlihatkan urutan interaksi pada proses verifikasi. Pengguna membuka aplikasi, login, memilih menu verifikasi, dan mengunggah PDF yang sudah memuat *QR Code*. Sistem membaca *QR Code* untuk memperoleh metadata, mengambil data penandatanganan dari basis data, menghitung ulang hash SHA-256, dan menjalankan verifikasi RSA-PSS. Sistem menampilkan hasil akhirnya kepada pengguna sebagai status valid atau tidak valid.

5 Mekanisme *QR Code Reference-based*

QR Code tidak menyimpan tanda tangan RSA-PSS, tetapi metadata berupa *document_id*, *signature_id*, nama berkas, waktu penandatanganan, dan identitas penandatanganan sebagai referensi ke data pada basis data. Pendekatan *reference-based* digunakan karena tanda tangan RSA 2048-bit berukuran 256 byte sehingga kurang efisien disimpan dalam *QR Code*. Pada verifikasi, metadata hanya digunakan untuk mengambil data dari server, sedangkan keabsahan dokumen ditentukan oleh hash SHA-256 dan verifikasi RSA-PSS. Karena *QR Code* merupakan bagian dari PDF final yang diproses secara kriptografis, perubahan atau pengantiannya mengubah byte stream dan menyebabkan hash tidak lagi cocok (Mainka et al., 2021).

Tabel 2. Fungsi metadata *QR Code*

Data QR	Fungsi
<i>document_id</i>	Mengidentifikasi dokumen pada basis data
<i>signature_id</i>	Mengidentifikasi record tanda tangan
Informasi metadata	Mendukung penelusuran dan identifikasi dokumen

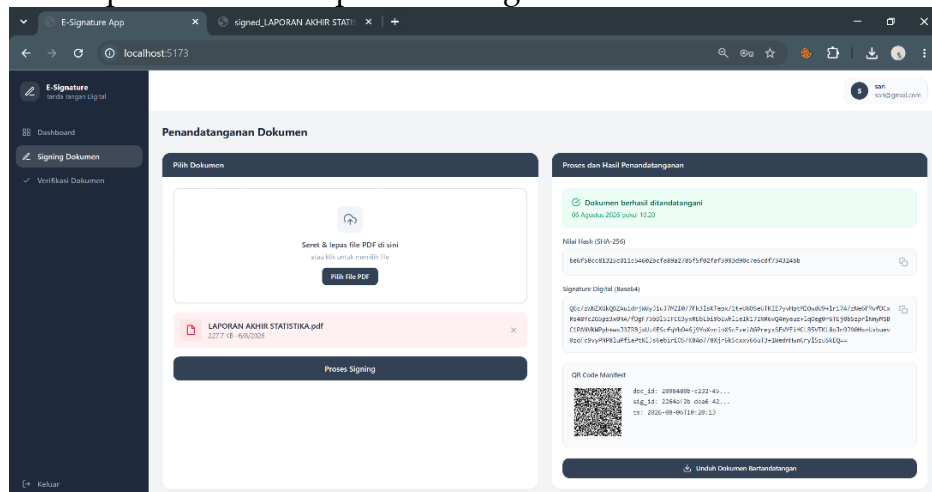
Reference ke server

Mengambil signature dan data terkait untuk verifikasi

HASIL DAN PEMBAHASAN

1. Implementasi Sistem

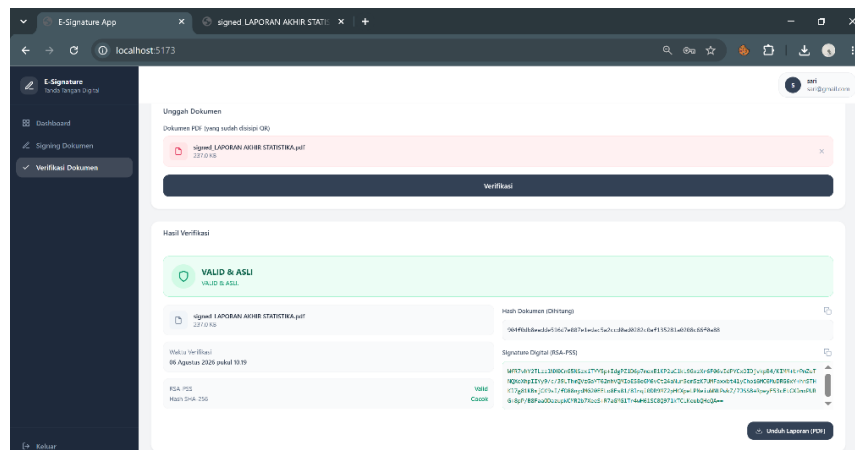
Hasil implementasi sistem berupa aplikasi web yang menyediakan fungsi autentikasi pengguna, pengelolaan pengguna, penandatanganan dokumen, dan verifikasi dokumen. Sistem menempatkan seluruh proses kriptografi di sisi backend, sehingga pembentukan hash dan tanda tangan digital tidak pernah tampak di antarmuka pengguna. Setiap kali pengguna menandatangani dokumen, sistem memproses berkas itu dengan SHA-256 dan RSA-PSS, menyimpan metadata tanda tangan, lalu membentuk dan menyisipkan *QR Code* pada halaman terakhir PDF. Gambar 5 menampilkan halaman penandatanganan.



Gambar 5. Halaman penandatanganan dokumen

Implementasi antarmuka menyediakan fungsi *signing* yang menghasilkan PDF final dengan *QR Code reference-based* sebelum proses hashing dan pembentukan signature.

Gambar 6 menampilkan halaman verifikasi, tempat pengguna mengunggah dokumen untuk diperiksa. Di sana, sistem membaca *QR Code* guna memperoleh identitas dokumen, mengambil data terkait dari basis data, menghitung ulang hash, lalu menjalankan verifikasi tanda tangan RSA-PSS sebelum menampilkan status akhirnya sebagai **VALID** atau **INVALID**.



Gambar 6. Halaman verifikasi dokumen

Proses di halaman verifikasi berjalan dalam empat langkah, yaitu membaca *QR Code* untuk mengidentifikasi dokumen, mengambil informasi terkait dari basis data, menghitung ulang hash, dan memverifikasi tanda tangan RSA-PSS. Pengguna menerima hasil akhirnya sebagai status **VALID** atau **INVALID**.

2. Hasil Pengujian Fungsional

Pengujian fungsional dilakukan terhadap fitur autentikasi dan manajemen pengguna, pembentukan tanda tangan digital, serta verifikasi dokumen. Pada pengujian autentikasi dan manajemen pengguna, terdapat tujuh skenario yang mencakup login dengan kredensial benar dan salah, pembatasan akses berdasarkan peran, penambahan pengguna, validasi data duplikat, serta pembatasan pembuatan akun Administrator. Seluruh skenario menghasilkan keluaran sesuai dengan kondisi yang diharapkan.

Pengujian pembentukan tanda tangan digital mencakup proses penandatanganan PDF secara end-to-end, konsistensi SHA-256, perubahan nilai hash setelah dokumen dimodifikasi, verifikasi dokumen asli dan dokumen yang dimodifikasi, pembentukan serta pembacaan *QR Code*, penyisipan *QR Code* pada halaman terakhir, dan pengunduhan dokumen hasil penandatanganan. Seluruh skenario tersebut menghasilkan keluaran sesuai dengan kondisi yang diharapkan.

Tabel 3. Ringkasan pengujian fungsional

Skenario	Objek	Hasil yang Diharapkan	Hasil
Autentikasi pengguna	Fungsional	Berhasil	Sesuai
Pembentukan hash SHA-256	Fungsional	Hash terbentuk	Sesuai
RSA-PSS <i>signing</i>	Fungsional	Signature terbentuk	Sesuai
RSA-PSS <i>verification</i>	Fungsional	Signature diverifikasi	Sesuai
<i>QR Code</i> generation/reading	Fungsional	QR berhasil dibuat/dibaca	Sesuai
Dokumen asli	10 PDF	VALID	Sesuai

Dokumen dimodifikasi	10 PDF	INVALID	Sesuai
Kunci publik tidak sesuai	Fungsional	Ditolak	Sesuai
QR Code dihapus	Fungsional	Verifikasi gagal	Sesuai
QR Code diganti/dimanipulasi	Fungsional	INVALID	Sesuai
Berkas bukan PDF	Fungsional	Ditolak	Sesuai

Hasil tersebut menunjukkan bahwa proses penandatanganan tidak hanya menghasilkan tanda tangan digital, tetapi juga mempertahankan keterkaitan antara dokumen PDF final, nilai hash, tanda tangan RSA-PSS, dan metadata yang direferensikan melalui QR Code. Hash SHA-256 menghasilkan nilai yang konsisten ketika dihitung terhadap dokumen yang identik, sedangkan perubahan pada dokumen menghasilkan nilai hash yang berbeda.

3. Hasil Pengujian 20 Dokumen PDF

Pengujian menggunakan 20 dokumen PDF dengan variasi jumlah halaman, ukuran file, dan jenis dokumen. Seluruh dokumen asli terlebih dahulu ditandatangani menggunakan pasangan kunci RSA terkait akun pengguna. Sepuluh dokumen dipertahankan tanpa perubahan, sedangkan sepuluh lainnya dimodifikasi setelah penandatanganan tanpa signing ulang. Seluruh dokumen kemudian diverifikasi menggunakan kunci publik yang sesuai.

Tabel 4. Hasil pengujian 20 dokumen PDF

No	Nama Dokumen	Jumlah Halaman	Ukuran File	Skenario Pengujian	Proses Penandatanganan	Proses Verifikasi
1	Surat Izin Tugas Kelompok	2	183 KB	Dokumen asli	Berhasil	VALID
2	Proposal Usaha	22	503 KB	Dokumen asli	Berhasil	VALID
3	Laporan Perancangan Smarthome	7	206 KB	Dokumen asli	Berhasil	VALID
4	Laporan Pengabdian Masyarakat	5	76 KB	Dokumen asli	Berhasil	VALID
5	Laporan KP	36	1,9 MB	Dokumen asli	Berhasil	VALID
6	Laporan KKN 2025	41	1,7 MB	Dokumen asli	Berhasil	VALID
7	Laporan Akhir Statistika	14	228 KB	Dokumen asli	Berhasil	VALID
8	Lampiran Surat	1	29 KB	Dokumen asli	Berhasil	VALID
9	Formulir Perbaikan Seminar Proposal	1	189 KB	Dokumen asli	Berhasil	VALID

10	Berita-Acara-Bimbingan-Proposal	1	256 KB	Dokumen asli	Berhasil	VALID
11	Surat Izin Tugas Kelompok	2	182 KB	Uji Perubahan tanggal Dokumen	Mengalami perubahan setelah <i>signing</i>	INVALID
12	Proposal Usaha	22	440 KB	Uji penambahan satu kalimat	Mengalami perubahan setelah <i>signing</i>	INVALID
13	Laporan Perancangan Smarthome	7	512 KB	Uji perubahan nama	Mengalami perubahan setelah <i>signing</i>	INVALID
14	Laporan Pengabdian Masyarakat	5	318 KB	Uji Penghapusan satu paragraf	Mengalami perubahan setelah <i>signing</i>	INVALID
15	Laporan KP	36	4,1 MB	Uji pengubahan gambar	Mengalami perubahan setelah <i>signing</i>	INVALID
16	Laporan KKN 2025	41	4,7 MB	Uji Penambahan satu karakter	Mengalami perubahan setelah <i>signing</i>	INVALID
17	Laporan Akhir Statistika (mengganti QR Code)	14	1,2 MB	Uji penggantian QR Code	Mengalami perubahan setelah <i>signing</i>	INVALID
18	Lampiran Surat (menimpa QR Code dengan gambar lain)	1	162 KB	Uji manipulasi QR Code	Mengalami perubahan setelah <i>signing</i>	INVALID
19	Formulir Perbaikan Seminar Proposal (menghapus tanggal)	1	253 KB	Uji penghapusan tanggal	Mengalami perubahan setelah <i>signing</i>	INVALID
20	Berita-Acara-Bimbingan-Proposal (menghapus logo)	1	324 KB	Uji penghapusan logo	Mengalami perubahan setelah <i>signing</i>	INVALID

Sepuluh dokumen asli berukuran 29 KB–1,9 MB dengan 1–41 halaman berhasil diverifikasi VALID, sedangkan sepuluh dokumen yang dimodifikasi setelah penandatanganan tanpa signing ulang memperoleh status INVALID. Hasil ini menunjukkan bahwa SHA-256 dan RSA-PSS mampu mendeteksi perubahan dokumen, sementara QR Code berfungsi sebagai referensi metadata dan validitas ditentukan melalui verifikasi kriptografis RSA-PSS.

4. Pengujian SHA-256

SHA-256 bersifat deterministic yaitu dokumen identik selalu menghasilkan nilai hash yang sama. Sifat ini memungkinkan sistem membandingkan hash yang tersimpan saat penandatanganan dengan hash saat verifikasi. Begitu dokumen berubah, byte stream-nya ikut berubah, dan nilai hash-nya otomatis berbeda. Tabel 5 merangkum pengujian perubahan satu karakter saja menghasilkan perbedaan signifikan pada *message digest* (National Institute of Standards and Technology, 2015).

Tabel 5. Pengujian konsistensi SHA-256

Kondisi	SHA-256
PDF asli	d4da3040ff8fb259059c90233edc69b42e9830f646e3c5d1d2d9f517c3412577
PDF identik	d4da3040ff8fb259059c90233edc69b42e9830f646e3c5d1d2d9f517c3412577
PDF + 1 karakter	a08a65d6554634da041961ed89932dc06fb85acbcd54b25148ff0af94b297eeb

Tabel 5 menegaskan bahwa proses verifikasi menangkap perubahan pada isi maupun struktur PDF, karena sistem menghasilkan nilai hash yang berbeda dari nilai hash saat penandatanganan. Dengan demikian, perbedaan nilai hash dapat digunakan sebagai indikator adanya perubahan pada dokumen setelah proses penandatanganan.

5. Pengujian RSA-PSS

Sistem berhasil membentuk dan memverifikasi tanda tangan RSA-PSS dengan RSA 2048-bit menggunakan SHA-256, MGF1, salt acak, dan EMSA-PSS sesuai RFC 8017. Tanda tangan hanya dapat diverifikasi dengan public key pasangan private key yang digunakan, perubahan dokumen menyebabkan hash berubah sehingga verifikasi gagal. Signing ulang dengan kunci yang sama menghasilkan signature berbeda karena salt acak, tetapi tetap VALID dengan public key yang sesuai. Untuk penyajian hasil, digunakan identitas D1, Signature A1–A2, Private Key A, Public Key A, dan Public Key B.

Tabel 6. Pengujian RSA-PSS

No.	Kondisi Pengujian	Dokumen	Kunci Signing	Kunci Verifikasi	Hasil Signature	Hasil Verifikasi
1	Signing pertama	D1	Private Key A	Public Key A	Signature A1 terbentuk	VALID
2	Signing ulang dokumen yang sama	D1	Private Key A	Public Key A	Signature A2 terbentuk dan berbeda dari A1	VALID
3	Verifikasi Signature A1 dengan public key pasangannya	D1	—	Public Key A	Signature A1	VALID
4	Verifikasi Signature A2	D1	—	Public Key A	Signature A2	VALID

	dengan public key pasangannya					
5	Verifikasi Signature A1 menggunakan public key pengguna lain	D1	—	Public Key B	Signature A1	INVALID

Hasil pengujian menunjukkan bahwa signing ulang menggunakan private key yang sama menghasilkan signature berbeda karena penggunaan salt acak pada RSA-PSS. Kedua signature tetap VALID ketika diverifikasi menggunakan public key pasangannya. Sebaliknya, penggunaan public key yang tidak sesuai menghasilkan INVALID. Hasil tersebut menunjukkan bahwa validitas signature bergantung pada kesesuaian dokumen, signature, dan pasangan kunci yang digunakan.

6. Pengujian QR Code

QR Code berfungsi baik sebagai media rujukan metadata. Sistem memakai informasi di dalamnya untuk mengidentifikasi dokumen dan tanda tangan yang bersesuaian pada basis data, sebagaimana Tabel 7 tunjukkan. Menghapus QR Code menghentikan sistem dari memperoleh metadata yang dibutuhkan, sehingga proses verifikasi terhenti sebelum masuk ke pemeriksaan kriptografis. Mengganti QR Code dengan kode lain yang masih terbaca membuat metadata yang diperoleh tidak cocok dengan data pada basis data, sehingga sistem menyatakan dokumen INVALID. QR Code yang dimanipulasi menghasilkan status yang sama.

Tabel 7. Hasil pengujian QR Code

Skenario QR	Hasil	Makna
QR Code normal	Berhasil dibaca	Metadata dapat digunakan untuk verifikasi
QR Code dihapus	Verifikasi gagal	Referensi metadata tidak tersedia
QR Code diganti	INVALID	Perubahan PDF terdeteksi
QR Code dimanipulasi	INVALID	Perubahan PDF terdeteksi

Pada QR Code yang normal, metadata dapat dibaca dan digunakan untuk melanjutkan *verification*. Penggantian QR Code dengan kode lain dapat mengarahkan sistem pada metadata yang tidak sesuai. Saat QR dihapus maka sistem tidak memperoleh metadata sehingga *verification* tidak dapat dilanjutkan, sedangkan manipulasi terhadap bagian QR Code mengubah dokumen yang menjadi objek pemeriksaan. Dengan demikian, QR Code berfungsi sebagai penghubung metadata, bukan sebagai penentu tunggal validitas kriptografis.

7. Perbandingan dengan Penelitian Terdahulu

Tabel 8. Perbandingan Penelitian Terdahulu

No.	Penelitian	Algoritma	Objek & QR Code	Keterbatasan / Perbedaan
-----	------------	-----------	-----------------	--------------------------

1	Aryasanti et al. (2022)	RSA + SHA-512	Tanda tangan digital berbasis web	Belum menggunakan RSA-PSS dan fokus pada PDF final
2	Sarasvananda & Iswara (2022)	RSA	Surat-menyurat + <i>QR Code</i>	Skema padding RSA tidak menjadi fokus
3	Akbar et al. (2021)	RSA	Lembar pengesahan skripsi	Objek terbatas dan belum menggunakan RSA-PSS
4	Hutagalung et al. (2023)	RSA + SHA-256	Surat Keterangan Lulus + <i>QR Code</i>	Belum menerapkan RSA-PSS secara spesifik
5	Barus et al. (2025)	RSA + SHA-256	Surat Perintah Perjalanan Dinas + <i>QR Code</i>	Fokus pada jenis dokumen tertentu
6	Nurjaman & Rizal (2024)	RSA + SHA3-512	Penyimpanan file + <i>QR Code</i>	Belum menggunakan RSA-PSS dan SHA-256
7	Penelitian ini	RSA-PSS + SHA-256	PDF final + <i>QR Code</i> reference-based	20 PDF dengan berbagai modifikasi

Berdasarkan Tabel 8, penelitian terdahulu telah menerapkan RSA dan fungsi hash untuk tanda tangan digital pada berbagai jenis dokumen, serta sebagian memanfaatkan *QR Code* sebagai pendukung proses verifikasi. Namun, penelitian tersebut umumnya menggunakan RSA tanpa membahas RSA-PSS secara spesifik dan menggunakan objek dokumen tertentu. Penelitian ini memadukan SHA-256 dan RSA-PSS 2048-bit pada PDF final setelah *QR Code* disisipkan, dengan *QR Code* berfungsi sebagai referensi menuju metadata penandatanganan. Pengujian juga dilakukan pada 20 dokumen PDF dengan berbagai bentuk modifikasi untuk mengevaluasi kemampuan sistem dalam mendeteksi perubahan dokumen.

KESIMPULAN

Penelitian ini berhasil mengimplementasikan tanda tangan digital pada dokumen PDF menggunakan SHA-256 dan RSA-PSS 2048-bit. Penandatanganan dilakukan pada PDF final setelah *QR Code* disisipkan sehingga seluruh byte stream dilindungi, sedangkan *QR Code* berfungsi sebagai referensi metadata dan data tanda tangan. Pengujian menunjukkan 10 dokumen asli berstatus VALID, sedangkan 10 dokumen yang dimodifikasi berstatus INVALID. Perubahan dokumen menghasilkan hash berbeda dan tanda tangan ditolak oleh kunci publik yang tidak sesuai. Hasil tersebut menunjukkan bahwa kombinasi SHA-256 dan RSA-PSS mampu mendukung verifikasi integritas dan mendeteksi perubahan dokumen. Penelitian selanjutnya dapat mengkaji perlindungan private key, penerapan PKI, verifikasi pada perangkat mobile, serta performa melalui load testing dan stress testing.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada kedua orang tua dan keluarga atas doa, dukungan, dan motivasi yang diberikan selama proses penelitian. Terima kasih juga disampaikan kepada Universitas Muhammadiyah Sumatera Utara, khususnya

Program Studi Sistem Informasi, atas dukungan yang diberikan dalam penyelesaian penelitian ini. Serta kepada pihak-pihak yang telah memberikan dukungan dalam proses penelitian dan penyusunan artikel ini.

DAFTAR PUSTAKA

- Akbar, M., L.B, P. L., & Fattah, F. (2021). Penerapan Rivest Shamir Adleman untuk Verifikasi Tanda Tangan Digital pada Lembar Pengesahan Skripsi. *Buletin Sistem Informasi Dan Teknologi Islam*, 2(4), 255–264. <https://doi.org/10.33096/busiti.v2i4.993>
- Aryasanti, A., Hardjianto, M., Brotosaputro, G., & Roeswidiyah, R. (2022). Implementasi Tanda Tangan Digital Menggunakan Algoritme RSA dan SHA-512 Berbasis Web. *Jurnal Ticom*, 10, 181–186. <https://doi.org/10.70309/ticom.v10i3.36>
- Barus, R. P., Suryanata, M. G., Pane, D. H., & Hafiz, A. Al. (2025). Penerapan Tanda Tangan Digital Menggunakan Algoritma SHA - 256 Dan RSA Untuk Pengamanan Surat Perintah Perjalanan Dinas Di Desa Namo Rambe. *JURNAL SISTEM INFORMASI TGD*, 4, 881–892. <https://doi.org/10.53513/jursi.v4i4.10313>
- Hutagalung, J., Ramadhan, P. S., & Sihombing, S. J. (2023). KEAMANAN DATA MENGGUNAKAN SECURE HASHING ALGORITHM (SHA)- 256 DAN RIVEST SHAMIR ADLEMAN (RSA) PADA DIGITAL SIGNATURE Juniar. *Jurnal Teknologi Informasi Dan Ilmu Komputer (JTIK)*, 10(6). <https://doi.org/10.25126/jtik.2023107319>
- K. Moriarty, E., Kaliski, B., Jonsson, J., & Rusch, A. (2016). *PKCS #1: RSA Cryptography Specifications Version 2.2 (RFC 8017)*. <https://www.rfc-editor.org/info/rfc8017>
- Kazmierczyk, Z., & Turner, I. J. (2022). Self-identification of electronically scanned signatures (ESS) and digitally constructed signatures (DCS). *Forensic Sciences Research*, 7(2), 261–264. <https://doi.org/10.1080/20961790.2021.1923167>
- Mainka, C., Mladenov, V., & Rohlmann, S. (2021). Shadow Attacks: Hiding and Replacing Content in Signed PDFs. *Network and Distributed Systems Security (NDSS) Symposium, February*. <https://doi.org/10.14722/ndss.2021.24117>
- Melina, M., Sukono, F., Napitupulu, H., & Kusumaningtyas, V. A. (2022). Verifikasi Tanda Tangan Elektronik dengan Teknik Otentikasi Berbasis Kriptografi Kunci Publik Sistem Menggunakan Algoritma Kriptografi Rivest-Shamir-Adleman. *Jurnal Matematika Integratif*, 18(1), 27. <https://doi.org/10.24198/jmi.v18.n1.38343.27-39>
- Natho, P., Boonmee, S., & Khongkraihoek, N. (2025). Evaluation of Digital Signatures Using RSA and HMAC Algorithms. *Engineering, Technology and Applied Science Research*, 15(6), 29509–29514. <https://doi.org/10.48084/etasr.14149>
- National Institute of Standards and Technology. (2015). Secure Hash Standard. In *National Institute of Standards and Technology* (Vol. 4, Issue August). <https://doi.org/10.6028/NIST.FIPS.180-4>
- Nurjaman, & Rizal, A. (2024). Penanda Tanganan Dokumen Digital Pada Sistem Penyimpanan File Menggunakan Kombinasi Algoritma SHA3-512 dan RSA

untuk. *Jurnal Ilmiah Teknologi Informasi Terapan*, 10(3), 119–129.
<https://doi.org/10.33197/jitter.vol10.iss3.2024.2200>

Sarasvananda, I. B. G., & Iswara, I. B. A. I. (2022). Tanda Tangan Elektronik Menggunakan Algoritma Rivest Shamir Adleman (RSA) pada Sistem Informasi Surat Menyurat LPIK INSTIKI. *Jurnal SISFOKOM (Sistem Informasi Dan Komputer)*, 11, 289–296. <https://doi.org/10.32736/sisfokom.v11i2.1403>