Original Article

e-ISSN: 2774-2016 - https://journal.itera.ac.id/index.php/indojam/
p-ISSN: 2774-2067

Received 14th July 2026
Accepted 31st October 2026
Published 31st October 2026

Open Access

DOI
https://doi.org/10.35472/indojam.v6i2.2992

Elliptic Curve Digital Signature Algorithm dan *Secure Hash Algorithm-256* untuk Autentikasi Transaksi Berbasis PIN pada Prototipe Aplikasi *Mobile Banking*

Alvina Hardiyanti[a], Siti Fatimah[a], Rini Marwati*[a]

[a,b,c,d] *Program Studi Matematika, Universitas Pendidikan Indonesia*

* *Corresponding E-mail: rinimarwati@upi.edu*

Abstract: *The implementation of security models such as device-bound keys and Fast Identity Online 2 (FIDO2) in mobile banking applications remains vulnerable to local attacks, while Transport Layer Security (TLS) protection often leaves vulnerabilities regarding the misuse of certificate pinning, exposing the transmission of Personal Identification Numbers (PIN) to Man-in-the-Middle (MitM) eavesdropping. As a solution to these physical and transmission vulnerabilities, this research integrates the Elliptic Curve Digital Signature Algorithm (ECDSA) and Secure Hash Algorithm-256 (SHA-256) for more secure transaction authentication in a mobile banking application prototype. The PIN is no longer sent to the server, but rather combined with the username and device ID to generate a private key using the Password-Based Key Derivation Function 2 (PBKDF2). The application was built using the Python programming language with a client-server architecture. The client performs hashing on the transaction data using SHA-256 and signs it with ECDSA, while the server verifies the signature. Functional testing of the application was conducted through three testing scenarios, namely normal transactions, access from unregistered devices, and transaction nominal manipulation. The test results indicate that the application is capable of executing signing and verification accurately across all scenarios and effectively supports authentication, data integrity, and non-repudiation in mobile banking transactions. However, this prototype has an inherent limitation in the event of a server database leak, where the low entropy of a 6-digit PIN enables offline brute-force attacks on the private key, thereby invalidating the non-repudiation aspect.*

Keywords: *Authentication, Digital Signature, ECDSA, Mobile Banking, PBKDF2, PIN, SHA-256*.

Abstrak: Penerapan model keamanan seperti *device-bound key* dan *Fast Identity Online 2* (FIDO2) pada aplikasi *mobile banking* masih rentan terhadap serangan lokal, sementara proteksi *Transport Layer Security* (TLS) kerap menyisakan celah penyalahgunaan *certificate pinning* yang membuat transmisi *Personal Identification Number* (PIN) terancam penyadapan *Man-in-the-Middle* (MitM). Sebagai solusi atas kerentanan fisik dan transmisi tersebut, penelitian ini mengintegrasikan *Elliptic Curve Digital Signature Algorithm* (ECDSA) dan *Secure Hash Algorithm-256* (SHA-256) untuk autentikasi transaksi yang lebih aman pada prototipe aplikasi *mobile banking*. PIN tidak lagi dikirimkan ke server, melainkan digabungkan bersama *username* dan *device ID* untuk membangkitkan kunci privat menggunakan *Password-Based Key Derivation Function 2* (PBKDF2). Prototipe aplikasi dibangun menggunakan bahasa pemrograman Python dengan arsitektur klien-server. Klien melakukan *hashing* data transaksi menggunakan SHA-256 dan menandatanganinya dengan ECDSA, sementara server memverifikasi tanda tangan tersebut. Pengujian fungsional aplikasi dilakukan melalui tiga skenario pengujian, yaitu transaksi normal, akses dari perangkat tidak terdaftar, dan manipulasi nominal transaksi. Hasil pengujian menunjukkan bahwa aplikasi mampu mengeksekusi penandatanganan dan verifikasi secara akurat pada seluruh skenario serta secara efektif mendukung autentikasi, integritas data, dan nirsangkal pada transaksi *mobile banking*. Namun, prototipe ini memiliki keterbatasan apabila terdapat kebocoran basis data server, di mana rendahnya entropi PIN 6 digit memungkinkan serangan *offline brute-force* kunci privat sekaligus menggugurkan aspek nirsangkal.

Kata Kunci: Autentikasi, ECDSA, *Mobile Banking*, PBKDF2, PIN, SHA-256, Tanda Tangan Digital.

Copyright © 2026 – Indonesian Journal of Applied Mathematics.
Content from this work may be used under the terms of the Creative Commons Attribution-NonCommercial 4.0 International Licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI. Published under licence by Indonesian Journal of Applied Mathematics.

Pendahuluan

Perkembangan teknologi digital serta maraknya penggunaan perangkat *mobile* telah mendorong inovasi di sektor perbankan. Hal ini menjadikan aplikasi *mobile banking (m-banking)* sebagai layanan vital yang mempermudah berbagai transaksi finansial konsumen [1, 2]. Namun, di balik kemudahan tersebut, celah keamanan bagi serangan siber seperti *phishing*, *malware*, dan *man-in-the-middle attack* semakin terbuka lebar. Salah satu sasaran utama dari serangan tersebut adalah eksploitasi dan pencurian kredensial pengguna, seperti *Personal Identification Number* (PIN).

Meskipun sejumlah sektor perbankan telah menerapkan *device bound-key* ataupun *Fast Identity Online 2* (FIDO2) serta pengamanan transmisi jaringan menggunakan *Transport Layer Security* (TLS), pendekatan tersebut belum sepenuhnya kedap terhadap eksploitasi. Penelitian menunjukkan bahwa FIDO2 memiliki titik lemah pada serangan lokal, khususnya ketika penyerang berhasil mendapatkan akses fisik terhadap perangkat keras pengguna atau mengeksploitasi celah pada tingkat sistem [3]. Sementara itu, perlindungan TLS pada tingkat aplikasi perbankan kerap menyisakan celah penyalahgunaan *certificate pinning*, yang membuka ruang bagi penyerang untuk melakukan penyadapan data sensitif dan mengeksekusi serangan *phishing* atau MitM [4].

Untuk memitigasi kerentanan pada akses lokal dan transmisi jaringan tersebut, kriptografi digunakan sebagai teknik utama untuk melindungi informasi sensitif dari akses tidak sah [5]. Kriptografi modern berbasis algoritma kunci publik memungkinkan proses autentikasi dilakukan tanpa perlu mengirimkan informasi rahasia seperti PIN secara langsung. Penerapan arsitektur ini mengubah peran PIN, di mana PIN digunakan untuk membangkitkan kunci privat melalui *Password-Based Key Derivation Function 2* (PBKDF2). Kunci privat tersebut kemudian dimanfaatkan oleh *Elliptic Curre Digital Signature Algorithm* (ECDSA), yaitu algoritma kriptografi berbasis matematika kurva eliptik yang berfungsi menghasilkan tanda tangan digital unik pada setiap transaksi. ECDSA dipilih karena keunggulannya dalam memberikan tingkat keamanan yang tinggi meskipun dengan panjang kunci yang minimal [6].

Sebelum tanda tangan digital dibentuk oleh ECDSA, data transaksi terlebih dahulu diproses melalui tahapan *hashing* menggunakan SHA-256. *Hashing* merupakan proses kriptografis satu arah yang mengubah data input menjadi serangkaian nilai unik dengan panjang tetap. SHA-256 secara khusus diimplementasikan karena memiliki reputasi keamanan yang kuat serta sifatnya yang tahan terhadap kolisi [7]. Nilai *hash* ringkas inilah yang kemudian ditandatangani oleh ECDSA.

Penerapan alur kriptografis ini sejalan dengan penelitian terdahulu. [8] mengimplementasikan SHA-256, FIDO, dan enkripsi Fernet untuk autentikasi multifaktor pada tahapan *login* aplikasi keuangan *mobile*. Meskipun menawarkan tingkat keamanan yang memadai, proses validasi berlapis tersebut mengurangi efisiensi *user experience* karena harus melewati tiga tahapan autentikasi, yaitu input PIN, verifikasi OTP, dan pemindaian biometrik. Sementara itu, [9] mengimplementasikan ECDSA dan SHA-256 pada skema *Two-Factor Authentication* (2FA) untuk *login* akun, yang terbukti mampu meningkatkan keamanan autentikasi sekaligus mencegah akses tidak sah. Mengacu pada keberhasilan tersebut, penelitian ini mengembangkan pendekatan serupa untuk mengamankan proses autentikasi transaksi berbasis PIN pada prototipe aplikasi *m-banking* menggunakan ECDSA dan SHA-256, dengan memanfaatkan PBKDF2 sebagai algoritma penurunan kunci.

Copyright © 2026 – Indonesian Journal of Applied Mathematics

Published by: Lembaga Penelitian dan Pengabdian Masyarakat (LPPM)
Institut Teknologi Sumatera, Lampung Selatan, Indonesia

Original Article

Metode

Password-Based Key Derivation Function 2 (PBKDF2)

PIN umumnya memiliki entropi rendah, sehingga rentan terhadap serangan *offline* seperti *brute-force*, *dictionary attack*, maupun *rainbow table attack*. Untuk memitigasi kerentanan tersebut, algoritma PBKDF2 diimplementasikan berdasarkan standar [10]. Kekuatan PBKDF2 terletak pada PRF dan jumlah iterasi, sehingga meminimalisir serangan tebakan dengan meningkatkan beban komputasi.

Namun, perlu dicatat bahwa PBKDF2 tidak meningkatkan entropi asli PIN, sehingga kekuatan keamanan pada akhirnya tetap dibatasi oleh ruang kombinasi PIN yang digunakan. Meskipun demikian, penggunaan sandi yang lebih panjang dan rumit akan memaksimalkan hambatan komputasi dari algoritma ini. Hal ini sejalan dengan temuan [11] yang menunjukkan bahwa eskalasi biaya sumber daya dan waktu komputasi akan membuat serangan semacam ini menjadi tidak layak secara ekonomi bagi penyerang. Secara matematis, perumusan PBKDF2 adalah sebagai berikut:

$$DK = \text{PBKDF2}(\text{PRF}, P, S, c, dkLen) \quad (1)$$

di mana:

- PRF : *pseudorandom function* (menggunakan HMAC-SHA256)
- P: PIN
- S: *salt*
- c: jumlah iterasi
- $dkLen$: panjang kunci yang ingin dihasilkan
- DK: kunci turunan yang dihasilkan

Secure Hash Algorithm-256 (SHA-256)

SHA-256 merupakan fungsi *hash* iteratif yang termasuk dalam SHA-2. Algoritma ini memproses data masukan berukuran sembarang menjadi *message digest* berukuran tetap 256-bit. Keunggulan dari algoritma ini adalah sifat antikolisinya (*collision resistance*). Secara garis besar, proses komputasi SHA-256 disederhanakan ke dalam tahapan operasional berikut [12]:

1. **Pra-pemrosesan:** Penambahan bit pengganjal (*padding*) dan informasi panjang pesan asli agar total ukuran blok menjadi kelipatan 512-bit.
2. **Inisialisasi & Parsing:** Pesan panjang dipecah menjadi blok-blok 512-bit. Sistem menyiapkan 8 variabel nilai *hash* awal (diambil dari akar bilangan prima).
3. **Kompresi (64 Putaran):** Setiap blok 512-bit diekspansi menjadi 64 kata, lalu diproses melalui 64 putaran operasi logika matematika rumit (*sigma*, *choice*, *majority*) menggunakan konstanta unik.
4. **Output Akhir:** Nilai *hash* akhir dibentuk melalui operasi konkatenasi (penggabungan) dari kedelapan variabel *hash* pada blok terakhir.

$$H = H_0^{(N)} \parallel H_1^{(N)} \parallel H_2^{(N)} \parallel H_3^{(N)} \parallel H_4^{(N)} \parallel H_5^{(N)} \parallel H_7^{(N)} \quad (2)$$

Elliptic Curve Digital Signature Algorithm (ECDSA)

ECDSA merupakan salah satu algoritma tanda tangan digital berbasis kriptografi kurva eliptik yang menawarkan tingkat keamanan setara dengan RSA, namun dengan ukuran kunci yang lebih kecil [13]. Dalam penelitian ini, kurva eliptik yang digunakan adalah kurva standar NIST P-256.

Adapun parameter domain global yang harus disepakati oleh pihak yang terlibat meliputi bilangan prima (p), koefisien kurva (a, b) yang mendefinisikan persamaan kurva $y^2 = x^3 + ax + b$, titik basis (G), dan orde (n). Selanjutnya, tahapan proses ECDSA dibagi ke dalam tiga langkah utama, yaitu:

1. **Pembangkitan Kunci**

Signer harus membangkitkan sepasang kunci dengan langkah-langkah berikut:

a. Bangkitkan Kunci Privat (d) secara deterministik menggunakan PBKDF2 dalam rentang $[1, n-1]$.
b. Hitung Kunci Publik (Q) melalui operasi perkalian skalar terhadap titik basis (G):

$$Q = dG \quad (3)$$

2. Pembangkitan Tanda Tangan

Tanda tangan digital untuk pesan m dibentuk melalui langkah-langkah berikut:

a. Pilih bilangan acak k dalam rentang $[1, n-1]$ menggunakan *Cryptographically Secure Pseudorandom Number Generator* (CSPRNG) berbasis entropi sistem operasi [14].
b. Hitung titik kurva R:

$$R = kG \quad (4)$$

c. Ambil nilai koordinat-x dari titik R, kemudian hitung nilai r:

$$r = x_R \pmod n \quad (5)$$

jika $r = 0$, pilih k yang baru.
d. Hitung nilai *hash* dari pesan m menggunakan SHA-256, sehingga:

$$e = H(m) \quad (6)$$

e. Hitung nilai s dengan rumus:

$$s = k^{-1}(e + dr) \pmod n \quad (7)$$

jika $s = 0$, pilih k yang baru.
f. Tanda tangan digital untuk pesan m adalah pasangan (r, s).

3. Verifikasi Tanda Tangan

Verifier memverifikasi keabsahan tanda tangan (r, s) dengan langkah-langkah berikut:

a. Verifikasi (r, s) berada dalam rentang $[1, n-1]$.
b. Hitung nilai *hash* dari pesan yang diterima untuk mendapatkan nilai e.
c. Hitung nilai w, yaitu invers multiplikatif dari s modulo n:

$$w = s^{-1} \pmod n \quad (8)$$

d. Hitung dua variabel bantu u_1 dan u_2:

$$u_1 = ew \pmod n \quad (9)$$

$$u_2 = rw \pmod n \quad (10)$$

e. Hitung titik baru X pada kurva menggunakan kunci publik (Q):

$$X = u_1 G + u_2 Q \quad (11)$$

jika $X = \mathcal{O}$, maka tanda tangan ditolak. Jika tidak, hitung nilai v.
f. Ambil koordinat- x dari titik X untuk menghitung nilai v:

$$v = x_X \pmod n \quad (12)$$

g. Tanda tangan dinyatakan valid jika dan hanya jika nilai

$$v = r \quad (13)$$

Pustaka Python

Aplikasi dibangun menggunakan bahasa pemrograman Python v3.14.0 dengan pustaka pendukung sebagai berikut:

- tkinter dan customtkinter: Pustaka antarmuka grafis standar serta tingkat lanjut untuk membangun tampilan aplikasi.
- ecdsa: Pustaka kriptografi eksternal untuk pembangkitan kunci, penandatanganan (*signing*), dan verifikasi tanda tangan digital.
- hashlib: Pustaka standar untuk fungsi *hashing* SHA-256 dan derivasi kunci PBKDF2 dalam pembentukan kunci privat.
- time dan uuid: Pustaka standar untuk menghasilkan *timestamp* unik serta mengambil identitas perangkat (*device ID*).

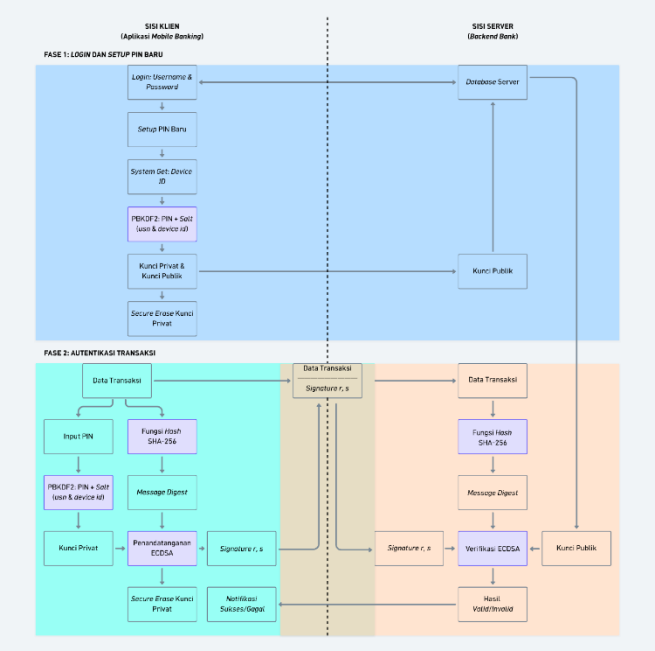
Pengembangan Model

Model aplikasi yang dikembangkan pada penelitian ini terbagi menjadi dua fase utama yang mengintegrasikan algoritma PBKDF2, SHA-256, dan ECDSA untuk autentikasi transaksi berbasis PIN, seperti diilustrasikan pada Gambar 1.

Copyright © 2026 – Indonesian Journal of Applied Mathematics

Published by: Lembaga Penelitian dan Pengabdian Masyarakat (LPPM)
Institut Teknologi Sumatera, Lampung Selatan, Indonesia

Original Article



Gambar 1. Pengembangan Model Prototipe Aplikasi *Mobile Banking*

Fase 1: *Login* dan *Setup* PIN Baru

Pada tahap ini, aplikasi dirancang untuk menghubungkan (*binding*) identitas pengguna dengan perangkat. Proses diawali dengan validasi *login* oleh server. Setelah proses validasi berhasil, pengguna diwajibkan untuk membuat PIN baru. Selanjutnya, aplikasi menggunakan PBKDF2 untuk mengolah PIN dan *salt* (*username* dan *device ID*) menjadi kunci privat. Kunci privat yang dihasilkan kemudian digunakan untuk membentuk kunci publik, yang selanjutnya dikirimkan ke server untuk disimpan dalam *database*. Kunci privat kemudian dihapus dari memori. Karena pembentukan pasangan kunci bergantung pada *device ID*, pengguna yang ingin beralih ke perangkat lain harus melakukan pendaftaran ulang perangkat di kantor bank terdekat.

Fase 2: Autentikasi Transaksi

Dalam autentikasi transaksi, aplikasi menggunakan mekanisme penandatanganan digital melalui dua tahap, yaitu penandatanganan dan verifikasi. Pada tahap penandatanganan (sisi klien), data transaksi yang mencakup rekening tujuan, nominal transaksi, dan *timestamp* (untuk mencegah serangan *replay attack*) diolah menggunakan SHA-256 untuk menghasilkan *message digest*. Selanjutnya, *message digest* tersebut ditandatangani dengan kunci privat yang diregenerasi menggunakan PBKDF2 setelah verifikasi PIN berhasil. Kunci privat kemudian kembali dihapus dari memori sesaat setelah *signature* terbentuk. Pada tahap verifikasi (sisi server), server melakukan hashing ulang dan memverifikasi *signature* menggunakan kunci publik pengguna sebagai penentu akhir keberhasilan atau penolakan transaksi.

Hasil dan Diskusi

Bagian ini menguraikan hasil implementasi dan pengujian dari prototipe aplikasi *mobile banking* untuk autentikasi transaksi berbasis PIN yang telah dimodelkan pada tahapan sebelumnya. Prototipe ini diimplementasikan dalam bentuk antarmuka desktop guna mengantisipasi keterbatasan perangkat *mobile*, khususnya terkait beban komputasi PBKDF2 dengan 100.000 iterasi per transaksi serta batasan akses pengelolaan pada *keystore* bawaan.

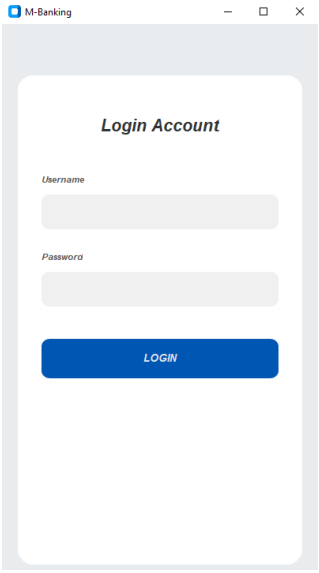
Tampilan Aplikasi dan Penggunaannya

Antarmuka aplikasi dibangun untuk memfasilitasi interaksi *user* sekaligus menjembatani rangkaian proses kriptografi di balik layar.

1. **Halaman *Login***

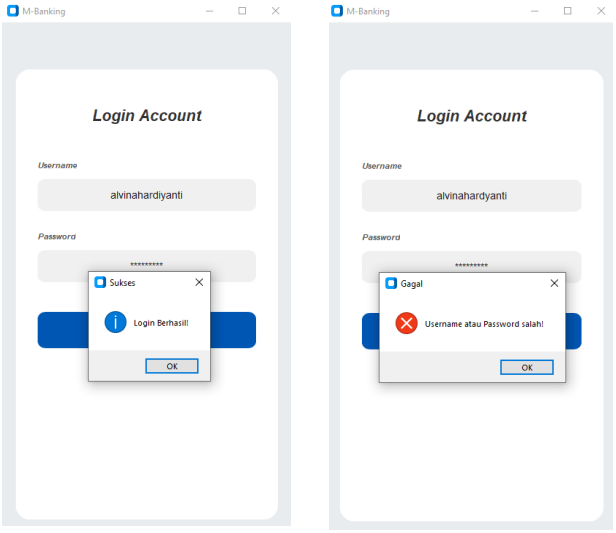
Login merupakan gerbang keamanan awal sebelum *user* dapat mengakses fitur utama aplikasi. Terdapat dua kolom masukan yang wajib diisi oleh *user*, yaitu *username* dan *password*. Tampilan halaman *login* dapat dilihat pada Gambar 2.

Setelah *user* menekan tombol "LOGIN", aplikasi akan mencocokkan kredensial yang dimasukkan dengan basis data pada server.



Gambar 2. Halaman *Login* Akun

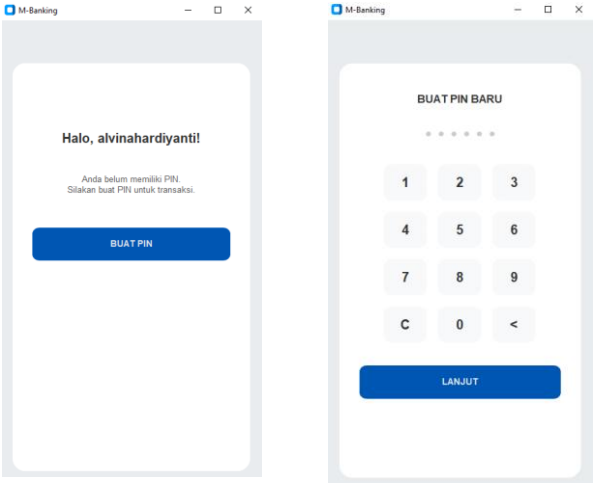
Apabila *username* dan *password* sesuai, aplikasi akan menampilkan informasi "Login Berhasil!" dan apabila sebaliknya, aplikasi akan menampilkan kotak dialog peringatan "Username atau Password salah!" seperti yang ditunjukkan pada Gambar 3.



Gambar 3. Tampilan Ketika *Login* Berhasil dan Gagal

2. Halaman *Setup* PIN Baru

Setelah proses *login* berhasil dilakukan untuk pertama kalinya, aplikasi mendeteksi bahwa *user* belum memiliki PIN, sehingga akan diarahkan ke halaman Buat PIN Baru. *User* diwajibkan untuk membuat PIN baru berupa kombinasi enam digit angka. Tampilan awal untuk alur pendaftaran PIN baru ini ditunjukkan pada Gambar 4.



Gambar 4. Halaman Pendaftaran PIN Baru

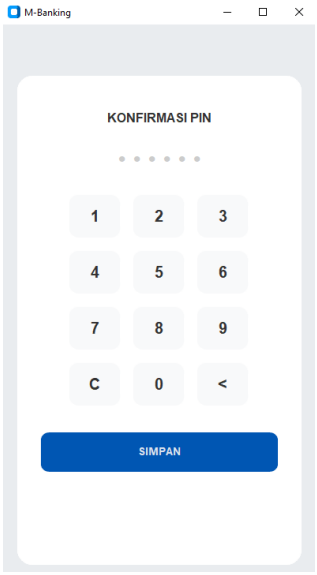
Setelah memasukkan PIN baru, *user* kemudian diarahkan ke halaman Konfirmasi PIN. Pada tahap ini, *user* harus memasukkan ulang kombinasi angka yang sama persis untuk memastikan tidak ada kesalahan pengetikan. Tampilan halaman Konfirmasi PIN dapat dilihat pada Gambar 5.

3. Halaman Transaksi Transfer Dana

Setelah PIN berhasil dibuat, aplikasi akan menampilkan kotak dialog pilihan apakah ingin melanjutkan transaksi transfer atau tidak. Apabila *user* memilih "YA, LANJUTKAN", aplikasi akan mengarahkan *user* ke halaman Menu Transfer. Sebaliknya, jika *user* memilih "TIDAK", sesi akan dialihkan kembali ke halaman *login* (seperti pada Gambar 1).

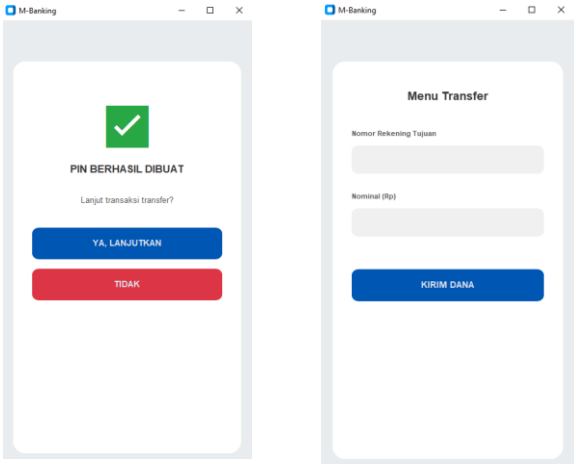
Copyright © 2026 – Indonesian Journal of Applied Mathematics
Published by: Lembaga Penelitian dan Pengabdian Masyarakat (LPPM)
Institut Teknologi Sumatera, Lampung Selatan, Indonesia

Original Article



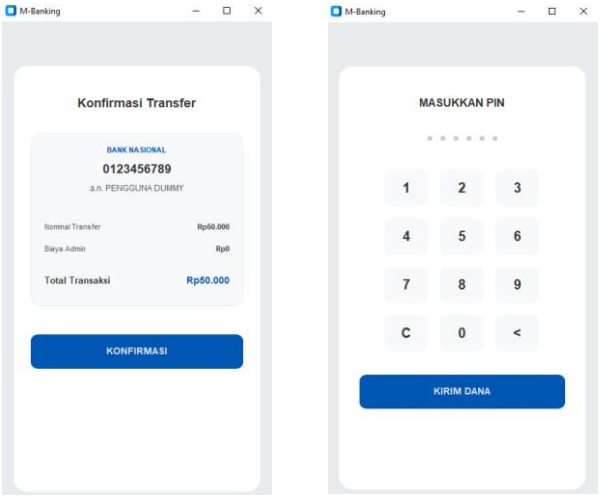
Gambar 5. Halaman Konfirmasi PIN Baru

Pada halaman Menu Transfer, *user* diminta memasukkan nomor rekening tujuan dan nominal transfer. Tampilan alur transaksi transfer dana ditunjukkan pada Gambar 6.



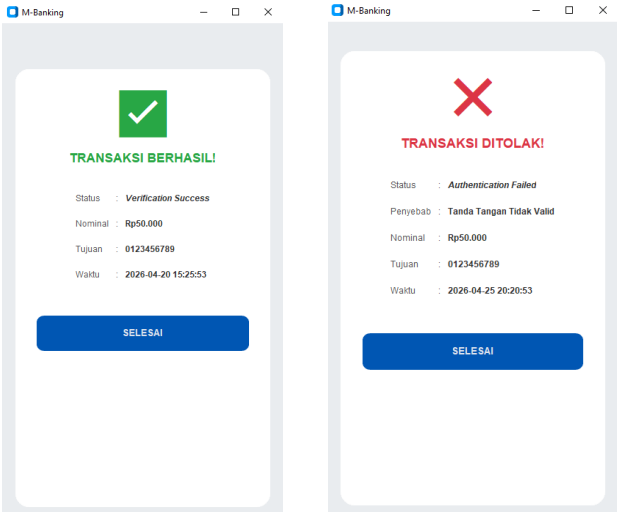
Gambar 6. Halaman Menu Transfer

Setelah *user* menekan tombol "KIRIM DANA", aplikasi mengarahkan *user* ke halaman Konfirmasi Transfer untuk memastikan tidak ada kesalahan *input*. Selanjutnya, *user* harus memasukkan PIN sebagai tanda persetujuan transaksi. Server menerapkan batas toleransi maksimal kesalahan input PIN sebanyak tiga kali sebelum akses transaksi diblokir guna menjaga keamanan transaksi. Tampilan proses konfirmasi dan *input* PIN ditunjukkan pada Gambar 7.



Gambar 7. Halaman Konfirmasi Transfer dan Input PIN

Setelah *user* memasukkan PIN, aplikasi menjalankan penandatanganan ECDSA dan mengirimkan paket data transaksi ke server. Apabila perhitungan matematis pada proses verifikasi ECDSA sesuai, aplikasi akan menampilkan halaman "TRANSAKSI BERHASIL!".



Gambar 8. Halaman Transaksi Berhasil dan Gagal

Sebaliknya, jika server mendeteksi adanya ketidaksesuaian, server akan menolak transaksi

tersebut dan aplikasi akan menampilkan halaman "TRANSAKSI DITOLAK!". Tampilan notifikasi hasil akhir transaksi ditunjukkan pada Gambar 8.

Skenario Pengujian

Pengujian keamanan aplikasi mencakup tiga skenario utama, yaitu transaksi valid (*normal case*), simulasi pencurian PIN (*device binding test*), dan manipulasi data transaksi (*integrity test*).

1. **Transaksi Valid (*Normal Case*)**

Pada skenario ini disimulasikan pengguna sah melakukan transaksi transfer dengan perangkat terdaftar dan PIN yang benar. Kunci privat dibangkitkan secara tepat, kemudian *signature* divalidasi dengan sukses oleh server sehingga transaksi disetujui, seperti yang ditunjukkan pada Gambar 8.

2. **Simulasi Pencurian PIN (*Device Binding Test*)**

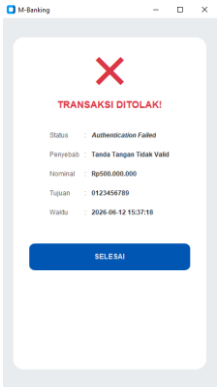
Pada skenario ini disimulasikan penyerang mengetahui *username* dan PIN yang benar, tetapi menggunakan perangkat yang berbeda. Ketidaksesuaian *device ID* menyebabkan algoritma PBKDF2 menghasilkan kunci privat yang salah, sehingga memicu *BadSignatureError*. Akibatnya, aplikasi secara otomatis menggagalkan transaksi, seperti yang ditunjukkan pada Gambar 8.

3. **Manipulasi Data Transaksi (*Integrity Test*)**

Pada skenario ini disimulasikan serangan *man-in-the-middle*, di mana data transaksi dicegat dan dimanipulasi sebelum mencapai server. Mengingat penandatanganan ECDSA terikat erat pada nilai *hash* data transaksi asli, perubahan sekecil apa pun akan mengubah total perhitungan matematis sehingga memicu *BadSignatureError*. Aplikasi pada akhirnya menolak transaksi dan menampilkan nominal yang telah dimanipulasi sebagai peringatan visual bagi pengguna sah, seperti pada Gambar 9.

Kesimpulan

Mekanisme autentikasi transaksi berbasis PIN pada prototipe aplikasi *mobile banking* dengan bahasa pemrograman Python telah berhasil diimplementasikan untuk mengatasi kerentanan pencurian PIN, di mana PIN tidak dikirimkan ke server, melainkan untuk membangkitkan kunci privat secara lokal melalui PBKDF2.



Gambar 9. Halaman Transaksi Ditolak Akibat Manipulasi Data

Data transaksi kemudian diubah menjadi nilai *hash* menggunakan SHA-256 kemudian ditandatangani dengan ECDSA. Arsitektur klien dan *server* beserta kinerja aplikasi ini telah dievaluasi melalui skenario pengujian dengan hasil akhir yang membuktikan bahwa aplikasi mampu mengeksekusi penandatanganan digital secara akurat pada seluruh kondisi, sehingga efektif dalam menjamin autentikasi, integritas, dan nirsangkal data.

Sebagai batasan penelitian, penggunaan PIN 6 digit secara teoritis menyisakan celah terhadap serangan *offline brute-force* apabila penyerang berhasil memperoleh akses terhadap kunci publik dan *salt* dari basis data server. Kerentanan ini berpotensi menyebabkan kebocoran kunci privat yang pada akhirnya dapat membatalkan jaminan aspek nirsangkal. Oleh karena itu, untuk pengembangan penelitian selanjutnya, disarankan menggunakan kombinasi sandi yang lebih panjang dan rumit, serta

Copyright © 2026 – Indonesian Journal of Applied Mathematics

Published by: Lembaga Penelitian dan Pengabdian Masyarakat (LPPM)
Institut Teknologi Sumatera, Lampung Selatan, Indonesia

penerapan arsitektur alternatif yang lebih tangguh seperti penggunaan perlindungan berbasis perangkat keras (*hardware-backed keys*) atau protokol *Password-Authenticated Key Exchange* (PAKE) seperti SRP maupun OPAQUE.

Konflik Kepentingan

Tidak ada konflik kepentingan yang dinyatakan.

Ucapan Terima Kasih

Ucapan terima kasih kami sampaikan kepada seluruh pihak yang telah berkontribusi dan memberikan dukungan dalam penyusunan artikel ini.

Referensi

[1] T. A. Berutu, D. Lorena, R. Sigalingging, and G. Kasih, "Pengaruh Teknologi Digital terhadap Perkembangan Bisnis Modern," *Neptunus: Jurnal Ilmu Komputer Dan Teknologi Informasi*, vol. 2, no. 3, pp. 358-370, 2024.

[2] H. Karjaluoto, A. A. Shaikh, H. Saarijärvi, and S. Saraniemi, "How Perceived Value Drives the use of Mobile Financial Services Apps," *International Journal of Information Management*, vol. 47, pp. 252-261, August 2019.

[3] T. K. Yadav dan K. Seamons, "A Security and Usability Analysis of Local Attacks Against FIDO2," dalam *Network and Distributed System Security (NDSS) Symposium 2024*, San Diego, CA, USA, 2024, doi: 10.14722/ndss.2024.24327.

[4] T. Chothia, F. Garcia, C. Heppel, dan C. McMahon Stone, "Why banker Bob (still) can't get TLS right: A Security Analysis of TLS in Leading UK Banking Apps," dalam *Financial Cryptography and Data Security: 21st International Conference (FC 2017)*, Sliema, Malta, 2017, pp. 579-597, doi: 10.1007/978-3-319-70972-7_33.

[5] V. Pongsitammu, A. R. Y. Simatupang, D. Annura, Y. Sari, Dachi, and D. R. Harries, "Keamanan Kriptosistem Modern Berdasarkan Algoritma Kriptografi Kunci Publik," *Siteba*, vol. 2, no. 1, 2023.

[6] E. Barker, "Recommendation for Key Management: Part 1 – General," National Institute of Standards and Technology (NIST), Special Publication (SP) 800-57 Part 1 Rev. 5, May 2020.

[7] V. Simangunsong, Y. R. Hutasoit, D. Siallagan, T. Rekayasa, P. Lunak, and I. T. Del, "Analisis Terhadap Keamanan Password Menggunakan Hash SHA-256," *Jurnal Quancom*, vol. 3, no. 1, pp. 13-17, 2025.

[8] G. Ali, M. A. Dida, and A. E. Sam, "A Secure and Efficient Multi-factor Authentication Algorithm for Mobile Money Applications," *Future Internet*, vol. 13, no. 12, pp. 1-31, 2021.

[9] H. Suseno, "Implementasi Elliptic Curve Digital Signature Algorithm dan Secure Hash Algorithm 256 pada Login Akun dengan Sistem Two-Factor Authentication," B.S. thesis, Universitas Pendidikan Indonesia, Bandung, Indonesia, 2025.

[10] K. Moriarty, B. Kaliski, and A. Rusch, "PKCS #5: Password-Based Cryptography Specification Version 2.1," RFC 8018, Internet Engineering Task Force (IETF), 2017.

[11] A. Visconti, O. Mosnáček, M. Brož, dan V. Matyáš, "Examining PBKDF2 security margin—Case study of LUKS," *Journal of Information Security and Applications*, vol. 46, pp. 296-306, 2019.

[12] National Institute of Standards and Technology, "Secure Hash Standard (SHS)," Federal Information Processing Standards Publication (FIPS PUB) 180-4, Aug. 2015.

[13] E. Barker, *Guideline for Using Cryptographic Standards in the Federal Government: Cryptographic Mechanisms*, NIST Special Publication 800-175B Revision 1, National Institute of Standards and Technology, Mar. 2020. doi: 10.6028/NIST.SP.800-175Br1.

[14] D. Eastlake 3rd, J. Schiller, dan S. Crocker, "Randomness Requirements for Security," RFC 4086, Jun. 2005.