ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

PEMANFAATAN EKSTENSI PRF WEBAUTHN UNTUK PEMBUNGKUSAN KUNCI PRIVAT SISI KLIEN PADA PENANDATANGANAN TRANSAKSI *OFFLINE HYPERLEDGER FABRIC*
Purwarupa dan Evaluasi Kelayakan

I Made Prajna Paramahansa Satriagama✉, I Nyoman Gede Arya Astawa, Agus Adi Putrawan
Jurusan Teknologi Informasi, Politeknik Negeri Bali, Badung, Indonesia
Email: 2215354047@pnb.ac.id

DOI: https://doi.org/10.46880/jmika.Vol10No2.pp607-621

ABSTRACT

Private key management remains a practical barrier to adopting consortium blockchains such as Hyperledger Fabric, where transactions must be signed with the ECDSA private key associated with an X.509 certificate. Server custody simplifies implementation but creates a single point of compromise, whereas delegating key files to users reduces usability and raises key-loss risk. This study proposes and evaluates a non-custodial prototype that uses the WebAuthn Pseudo-Random Function (PRF) extension to derive a client-side key-wrapping key. Unlike existing Fabric key management (file-based wallets, server custody, or HSM/PKCS#11) and prior WebAuthn PRF uses that mainly encrypt user data, this work combines PRF-derived key wrapping, envelope encryption, and Fabric offline signing to protect rather than replace the ECDSA signing key. The ECDSA P-256 key is generated in the browser and protected by envelope encryption: it is encrypted with a random Data Encryption Key (DEK, AES-256-GCM), while the DEK is wrapped with a Key Encryption Key derived from PRF output via HKDF-SHA256. The server stores only ciphertexts and the user certificate. During submission, the key is briefly decrypted in browser memory for client-side offline signing before being sent to the Fabric network. Evaluation with the asset-transfer-basic chaincode shows that browser-signed transactions are accepted and validated without exposing the plaintext private key to the server. Average cryptographic overhead is 0.248 ms (SD 0.022 ms). It remains feasible for interactive use, although PRF support remains uneven across authenticators. This study is a proof of concept for technical feasibility, not a comprehensive security analysis.

Keyword: WebAuthn PRF, Hyperledger Fabric, Offline Signing, Envelope Encryption, Non-Custodial Key Management.

ABSTRAK

Manajemen kunci privat masih menjadi hambatan praktis dalam adopsi blockchain konsorsium seperti Hyperledger Fabric, karena transaksi harus ditandatangani menggunakan kunci privat ECDSA yang terkait dengan sertifikat X.509. Penyimpanan kunci pada server menyederhanakan implementasi, tetapi menciptakan titik kompromi tunggal, sedangkan penyerahan berkas kunci kepada pengguna menurunkan kegunaan dan meningkatkan risiko kehilangan kunci. Penelitian ini mengusulkan dan mengevaluasi purwarupa non-kustodian yang menggunakan ekstensi Pseudo-Random Function (PRF) pada WebAuthn untuk menurunkan kunci pembungkus di sisi klien. Berbeda dari mekanisme manajemen kunci Fabric yang ada (wallet berbasis berkas, kustodian server, atau HSM/PKCS#11) maupun pemanfaatan WebAuthn PRF terdahulu yang umumnya hanya mengenkripsi data pengguna, penelitian ini memadukan pembungkusan kunci berbasis PRF, enkripsi envelope, dan offline signing Fabric untuk melindungi, bukan menggantikan, kunci penandatangan ECDSA. Kunci ECDSA P-256 dibangkitkan di peramban dan dilindungi melalui enkripsi envelope: kunci privat dienkripsi dengan Data Encryption Key (DEK) acak menggunakan AES-256-GCM, sedangkan DEK dibungkus dengan Key Encryption Key yang diturunkan dari keluaran PRF menggunakan HKDF-SHA256. Server hanya menyimpan ciphertext dan sertifikat pengguna. Saat pengiriman, kunci didekripsi sementara di memori peramban untuk client-side offline signing sebelum dikirim ke jaringan Fabric. Evaluasi dengan chaincode asset-transfer-basic menunjukkan transaksi yang ditandatangani dari peramban dapat diterima dan divalidasi tanpa mengekspos kunci privat plaintext. Rerata overhead kriptografi sebesar 0,248 ms (SB 0,022 ms), masih layak untuk penggunaan interaktif,

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

meskipun dukungan PRF belum merata antar-autentikator. Penelitian ini merupakan pembuktian konsep kelayakan teknis, bukan analisis keamanan menyeluruh.

Kata Kunci: WebAuthn PRF, Hyperledger Fabric, Offline Signing, Enkripsi Envelope, Manajemen Kunci Non-Kustodian.

PENDAHULUAN

Teknologi *blockchain* konsorsium banyak dipertimbangkan untuk sistem informasi yang menuntut integritas catatan, akuntabilitas, dan tata kelola identitas antarpihak (Chen et al., 2024; Krichen et al., 2022). Hyperledger Fabric merupakan salah satu *platform* utama pada kelas *permissioned blockchain* karena arsitekturnya *modular* dan mendukung pemisahan peran dalam jaringan (Androulaki et al., 2018). Dalam Fabric, identitas peserta direpresentasikan melalui sertifikat digital X.509 yang dikelola oleh *Membership Service Provider* (MSP). Setiap proposal dan transaksi harus ditandatangani secara kriptografis oleh kunci privat yang berpasangan dengan sertifikat tersebut. Dengan demikian, keamanan transaksi tidak hanya ditentukan oleh mekanisme konsensus dan validasi jaringan, tetapi juga oleh cara kunci privat pengguna disimpan, dibuka, dan digunakan.

Pada tingkat pengguna akhir, pengelolaan kunci privat menjadi persoalan yang sulit diseimbangkan antara keamanan dan kegunaan. Pendekatan kustodian, yaitu menyimpan kunci privat di *server* aplikasi, mudah diterapkan dan nyaman, tetapi menjadikan server sebagai titik kegagalan tunggal: jika basis data atau server terkompromi, kunci pengguna berpotensi ikut terekspos. Pendekatan sebaliknya, yaitu menyerahkan berkas kunci sepenuhnya kepada pengguna, menghilangkan ketergantungan pada *server* tetapi memindahkan beban keamanan kepada pihak yang sering kali tidak memiliki literasi kriptografi yang memadai. Studi tentang pengelolaan kunci pada sistem *blockchain* menunjukkan bahwa pengguna awam kerap mengalami kesulitan dalam menyimpan, mencadangkan, dan memulihkan kunci atau frasa pemulihan secara aman (Eskandari et al., 2015; Sharma et al., 2024).

Perkembangan WebAuthn dalam ekosistem FIDO2 menawarkan jalan tengah yang relevan. WebAuthn memungkinkan autentikasi berbasis kunci publik yang tahan terhadap phishing melalui autentikator pada perangkat pengguna (W3C, 2021). Pada WebAuthn Level 3, ekstensi *Pseudo-Random Function* (PRF) menyediakan kemampuan untuk memperoleh nilai rahasia deterministik yang terikat pada kredensial tertentu tanpa mengekspos rahasia internal autentikator kepada aplikasi web (W3C, 2025). Dalam penelitian ini, kredensial WebAuthn tidak digunakan sebagai kunci penandatangan transaksi Fabric, melainkan hanya untuk menghasilkan bahan kunci yang diturunkan menjadi kunci pembungkus; penandatanganan transaksi tetap dilakukan oleh kunci ECDSA Fabric yang berpasangan dengan sertifikat X.509 pengguna.

Hyperledger Fabric mendukung mekanisme *offline signing*, yaitu pembentukan dan penandatanganan struktur proposal atau transaksi di luar SDK *server*. Mekanisme ini relevan untuk skenario ketika pengguna akhir ingin menjaga kerahasiaan kunci privatnya sendiri, sehingga aplikasi tidak perlu memegang kunci tersebut dalam bentuk *plaintext* (Hyperledger Foundation, 2026). Dengan menggabungkan *offline signing* dan ekstensi PRF WebAuthn, kunci privat Fabric dapat disimpan di *server* hanya dalam bentuk terenkripsi, dibuka sementara di memori peramban ketika pengguna melakukan autentikasi, lalu digunakan untuk menghasilkan tanda tangan transaksi di sisi klien.

Berdasarkan peluang tersebut, penelitian ini bertujuan membangun dan mengevaluasi purwarupa yang mengintegrasikan ekstensi PRF WebAuthn dengan mekanisme *offline signing* Hyperledger Fabric. Istilah *non-kustodian* dalam penelitian ini digunakan secara terbatas, yaitu kondisi ketika *server* aplikasi tidak pernah memegang kunci privat penandatangan dalam bentuk *plaintext*. Evaluasi difokuskan pada tiga pertanyaan:

- Apakah arsitektur yang diusulkan dapat menghasilkan transaksi yang diterima jaringan Fabric?
- Sejauh mana dukungan ekstensi PRF pada beberapa autentikator yang diuji?
- Berapa besar *overhead* komputasi kriptografi di sisi klien ketika kunci privat dibuka dan digunakan untuk penandatanganan?

Ruang lingkup penelitian dibatasi sebagai pembuktian konsep kelayakan teknis dengan *chaincode* asset-transfer-basic sebagai transaksi uji, dan belum mencakup studi kasus bisnis spesifik maupun analisis keamanan formal.

KAJIAN LITERATUR

Kajian literatur ini disusun untuk membangun dasar konseptual bagi rancangan perlindungan kunci privat pengguna pada Hyperledger Fabric melalui pemanfaatan ekstensi WebAuthn PRF. Pembahasan

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

difokuskan pada lima aspek yang saling berkaitan, yaitu mekanisme identitas dan *offline signing* pada Hyperledger Fabric, kemampuan WebAuthn PRF sebagai sumber bahan kunci, pola enkripsi envelope untuk perlindungan kunci privat, karakteristik autentikator, serta posisi penelitian ini dibandingkan penelitian terdahulu. Dengan struktur tersebut, kajian literatur tidak hanya menjelaskan istilah teknis, tetapi juga menunjukkan hubungan langsung antara literatur dan rancangan purwarupa yang dievaluasi dalam penelitian ini.

Hyperledger Fabric dan Offline Signing

Hyperledger Fabric adalah *platform permissioned blockchain* yang memisahkan peran node menjadi *peer* dan *orderer* serta menjalankan logika bisnis melalui *chaincode* (Androulaki et al., 2018). Identitas pada Fabric dibangun di atas *Public Key Infrastructure* (PKI): setiap anggota memiliki sertifikat X.509 yang diterbitkan oleh *Certificate Authority* (CA) organisasinya dan divalidasi oleh *Membership Service Provider* (MSP). Dalam alur transaksi Fabric, klien membentuk proposal transaksi, menandatanganinya dengan kunci privat yang berpasangan dengan sertifikat X.509, lalu mengirimkannya ke jaringan untuk melalui tahap *endorsement*, *ordering*, dan *validation*. Dengan demikian, kunci privat pengguna merupakan komponen penting dalam pembuktian identitas transaksi.

Dukungan *offline signing* pada Hyperledger Fabric memungkinkan proses pembentukan tanda tangan dilakukan di luar SDK *server*, termasuk pada perangkat klien yang memegang kunci privat (Hyperledger Foundation, 2026). Mekanisme ini relevan untuk skenario ketika aplikasi tidak ingin atau tidak boleh menyimpan kunci privat pengguna dalam bentuk *plaintext*. Dalam konteks penelitian ini, *offline signing* menjadi prasyarat teknis agar transaksi tetap sah menurut mekanisme identitas Fabric, sementara operasi penggunaan kunci dipindahkan ke sisi klien. Dengan kata lain, rancangan yang diusulkan tidak mengganti model identitas Fabric, tetapi mengubah cara kunci privat Fabric disimpan dan digunakan.

WebAuthn PRF dan Derivasi Kunci

WebAuthn merupakan standar W3C untuk autentikasi berbasis kriptografi kunci publik, sedangkan CTAP2 dari FIDO Alliance menyediakan protokol komunikasi antara klien dan autentikator; keduanya membentuk kerangka FIDO2 (W3C, 2021). Pada proses registrasi, autentikator membuat pasangan kunci dan menyimpan kunci privat kredensial secara aman. Pada proses autentikasi, autentikator menandatangani *challenge* dari *server*. Namun, dalam penelitian ini kredensial WebAuthn tidak diposisikan sebagai pengganti kunci penandatangan Fabric. Kredensial WebAuthn hanya digunakan untuk menghasilkan bahan kunci yang membantu membuka kunci privat Fabric yang telah dienkripsi.

Ekstensi *Pseudo-Random Function* (PRF) pada WebAuthn Level 3 memperluas kemampuan WebAuthn dengan memanfaatkan mekanisme *hmac-secret* pada CTAP2. Melalui mekanisme tersebut, autentikator menghitung HMAC-SHA-256 atas *salt* tertentu menggunakan rahasia yang terikat pada kredensial (W3C, 2025). Keluaran PRF bersifat deterministik untuk kombinasi kredensial dan *salt* yang sama, tetapi rahasia internal autentikator tidak diekspos kepada aplikasi web. Karena keluaran PRF tidak secara langsung dipakai sebagai kunci enkripsi, keluaran tersebut perlu diturunkan kembali melalui fungsi derivasi kunci seperti *HMAC-based Key Derivation Function* (HKDF) agar menghasilkan kunci dengan panjang, konteks, dan pemisahan penggunaan yang sesuai (Krawczyk & Eronen, 2010).

Enkripsi Envelope untuk Perlindungan Kunci Privat

Pola enkripsi envelope memberikan kerangka yang sesuai untuk memanfaatkan keluaran PRF dalam perlindungan kunci privat Fabric. Dalam enkripsi envelope, data dilindungi menggunakan *Data Encryption Key* (DEK), sedangkan DEK dilindungi kembali menggunakan *Key Encryption Key* (KEK) (Barker, 2020). Pada penelitian ini, data yang dilindungi adalah kunci privat Fabric dalam format PKCS#8. Kunci privat tersebut dienkripsi menggunakan DEK, sedangkan DEK dibungkus menggunakan KEK yang diturunkan dari keluaran PRF WebAuthn. Dengan pola ini, *server* hanya perlu menyimpan *ciphertext* kunci privat, *ciphertext* DEK, dan sertifikat pengguna, tanpa perlu memegang kunci privat *plaintext*.

Untuk enkripsi simetris terautentikasi, AES dalam mode *Galois/Counter Mode* (AES-GCM) umum digunakan karena menyediakan kerahasiaan dan integritas data sekaligus (Dworkin, 2007). Penggunaan enkripsi envelope juga membantu penelitian ini memisahkan operasi kriptografi yang perlu dievaluasi secara kuantitatif. Overhead yang relevan bukan performa Hyperledger Fabric secara keseluruhan, melainkan biaya tambahan dari mekanisme perlindungan kunci di sisi klien, yaitu derivasi KEK melalui HKDF, dekripsi DEK, dekripsi dan impor kunci privat, serta penandatanganan ECDSA di peramban. Pemisahan ini penting agar hasil

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

pengukuran dapat ditafsirkan sebagai *overhead* dari rancangan manajemen kunci, bukan sebagai *benchmark* umum jaringan Fabric.

Karakteristik Autentikator dan Batas Klaim Keamanan

Karakteristik autentikator menentukan batas klaim keamanan dari arsitektur yang diusulkan. Autentikator WebAuthn dapat berupa autentikator *platform* yang melekat pada perangkat, seperti Windows Hello atau Touch ID, maupun penyedia *passkey* berbasis perangkat lunak yang menyinkronkan kredensial antarperangkat melalui akun pengguna. Pada autentikator *platform*, kredensial berpotensi terikat pada perangkat keras aman seperti TPM atau *secure enclave*. Sebaliknya, pada *passkey* tersinkron, portabilitas dan pemulihan menjadi lebih mudah, tetapi keamanan kredensial juga bergantung pada keamanan akun penyedia dan mekanisme sinkronisasinya.

Implikasi dari perbedaan tersebut adalah bahwa kunci yang diturunkan dari PRF tidak selalu dapat diklaim sebagai kunci yang sepenuhnya terikat perangkat keras. Pada konfigurasi *passkey* tersinkron, kunci pembungkus lebih tepat dipahami sebagai kunci berbasis perangkat lunak yang dilindungi oleh ekosistem penyedia *passkey* (Stamatopoulou et al., 2025). Selain itu, dukungan ekstensi PRF belum seragam pada semua autentikator. Oleh karena itu, selain menguji keberhasilan transaksi, penelitian ini juga perlu mengevaluasi kompatibilitas autentikator dan menyatakan batasan keamanan secara eksplisit.

Penelitian Terdahulu dan Celah Penelitian

Penelitian terdahulu mengenai *blockchain* dan dompet digital menunjukkan bahwa persoalan manajemen kunci bergerak dari penyimpanan terpusat menuju model yang lebih menempatkan kendali pada perangkat atau akun pengguna, tetapi tetap menghadapi tantangan kegunaan, pemulihan, dan kepercayaan (Eskandari et al., 2015; Sharma et al., 2024). Beberapa kajian terbaru mulai membahas pemanfaatan WebAuthn atau *passkey* untuk melindungi kunci aplikasi, termasuk pada konteks dompet identitas digital *non-kustodian* (Stamatopoulou et al., 2025). Literatur tersebut menunjukkan bahwa penggunaan autentikator modern untuk membantu perlindungan kunci merupakan arah yang relevan, tetapi konteks dan mekanisme yang digunakan masih beragam.

Celah penelitian yang menjadi fokus paper ini terletak pada kombinasi spesifik antara WebAuthn PRF, enkripsi envelope, dan *offline signing* Hyperledger Fabric. Dibandingkan mekanisme manajemen kunci yang umum digunakan pada Hyperledger Fabric—penyimpanan kunci privat ECDSA pada wallet berbasis berkas di sisi klien, penitipan kunci pada server aplikasi (model kustodian), atau perlindungan melalui modul perangkat keras khusus (PKCS#11/HSM)—pendekatan-pendekatan tersebut tetap menyisakan kunci dalam bentuk plaintext pada suatu titik penyimpanan atau menuntut infrastruktur tambahan (Androulaki et al., 2018; Barker, 2020; Hyperledger Foundation, 2026). Sementara itu, pemanfaatan ekstensi WebAuthn PRF pada penelitian terdahulu umumnya diarahkan untuk menurunkan kunci simetris guna mengenkripsi data atau brankas kata sandi pengguna, serta pada konteks dompet identitas digital non-kustodian, dan belum menyasar pembungkusan kunci penandatangan blockchain pada jaringan permissioned (Stamatopoulou et al., 2025; W3C, 2025). Penelitian ini tidak menggunakan WebAuthn sebagai kunci penandatangan transaksi, tidak mengganti mekanisme MSP Fabric, dan tidak berfokus pada dompet blockchain publik. Sebaliknya, penelitian ini memanfaatkan keluaran PRF sebagai sumber bahan kunci untuk membungkus kunci privat ECDSA Fabric, kemudian menggunakan kunci tersebut untuk menandatangani transaksi melalui mekanisme *offline signing* di peramban. Dengan demikian, kebaruan penelitian ini terletak pada kombinasi spesifik tersebut, yaitu pemanfaatan keluaran PRF WebAuthn sebagai sumber bahan kunci untuk membungkus—bukan menggantikan—kunci privat ECDSA Fabric, yang belum dibahas baik pada penelitian manajemen kunci Hyperledger Fabric maupun pada pemanfaatan WebAuthn PRF sebelumnya. Kontribusi utama penelitian ini mencakup empat hal: (1) rancangan arsitektur manajemen kunci privat non-kustodian yang memadukan ekstensi WebAuthn PRF, enkripsi envelope, dan mekanisme offline signing Hyperledger Fabric—kombinasi yang membedakannya, baik dari mekanisme manajemen kunci Fabric yang ada maupun dari implementasi WebAuthn PRF sebelumnya; (2) pembuktian kelayakan fungsional bahwa transaksi yang ditandatangani sepenuhnya di sisi klien dapat diterima dan divalidasi oleh jaringan Fabric; (3) evaluasi kompatibilitas ekstensi PRF lintas autentikator; dan (4) pengukuran *overhead* kriptografi sisi klien yang timbul dari proses pembukaan serta penggunaan kunci privat.

METODE PENELITIAN

Penelitian ini menggunakan pendekatan eksperimental berbasis purwarupa untuk mengevaluasi kelayakan teknis perlindungan kunci privat pengguna pada Hyperledger Fabric melalui ekstensi WebAuthn

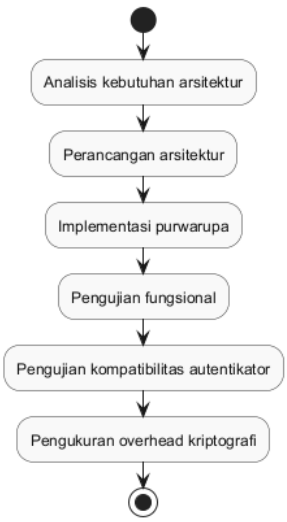
This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

PRF. Objek penelitian adalah arsitektur manajemen kunci privat dan penandatanganan transaksi dari sisi klien, bukan studi kasus bisnis tertentu. Model penelitian ini tidak menggunakan metode pengembangan perangkat lunak seperti Waterfall secara penuh, karena fokus utamanya adalah pembuktian kelayakan teknis dan pengukuran overhead kriptografi. Evaluasi difokuskan pada keberhasilan fungsional transaksi, kompatibilitas autentikator dalam menghasilkan keluaran PRF, serta overhead komputasi kriptografi ketika kunci privat dibuka dan digunakan untuk penandatanganan.

Tahapan Penelitian

Tahapan penelitian disusun untuk menggambarkan urutan kerja pengembangan dan evaluasi purwarupa, mulai dari penentuan kebutuhan arsitektur hingga analisis overhead kriptografi. Alur tahapan penelitian ditunjukkan pada Gambar 1.



Gambar 1. Diagram Tahap Penelitian

Secara umum, tahapan penelitian diawali dengan analisis kebutuhan arsitektur untuk mengidentifikasi komponen utama yang diperlukan, yaitu identitas Fabric berbasis sertifikat X.509, mekanisme *offline signing*, WebAuthn PRF, dan enkripsi envelope. Hasil analisis tersebut digunakan sebagai dasar perancangan arsitektur, terutama dalam memisahkan peran server sebagai penyimpan data terenkripsi dan klien web sebagai tempat pembangkitan, pembukaan, serta penggunaan kunci privat. Setelah rancangan ditetapkan, purwarupa diimplementasikan pada jaringan Hyperledger Fabric lokal dengan melibatkan server aplikasi, basis data, dan klien web. Purwarupa yang telah dibangun kemudian dievaluasi melalui pengujian fungsional untuk memastikan transaksi yang ditandatangani dari peramban dapat diterima oleh jaringan Fabric. Evaluasi dilanjutkan dengan pengujian kompatibilitas autentikator untuk melihat dukungan terhadap keluaran PRF. Tahap terakhir adalah pengukuran overhead kriptografi guna menilai biaya tambahan dari proses derivasi kunci, dekripsi, impor kunci, dan penandatanganan di sisi klien.

Arsitektur Sistem

Arsitektur purwarupa terdiri atas tiga komponen utama, yaitu perangkat klien, server aplikasi, dan jaringan Hyperledger Fabric lokal. Perangkat klien menjalankan aplikasi web melalui peramban. Pada sisi klien, WebAuthn API digunakan untuk berinteraksi dengan autentikator, sedangkan Web Crypto API digunakan untuk menjalankan operasi kriptografi lokal, seperti pembangkitan kunci, pembentukan Certificate Signing Request (CSR), derivasi kunci, dekripsi, dan penandatanganan. Dengan rancangan ini, operasi yang melibatkan kunci privat Fabric dilakukan di sisi klien, bukan pada server aplikasi.

Server aplikasi dikembangkan menggunakan Java Spring Boot dan berkomunikasi dengan klien web melalui HTTPS. Server berperan sebagai penghubung antara klien web, basis data MySQL, dan jaringan Hyperledger Fabric melalui Fabric Gateway SDK. Dalam arsitektur ini, server menerima dan mengelola data yang sudah berada dalam bentuk terenkripsi, serta meneruskan proses yang berkaitan dengan identitas dan transaksi ke jaringan Fabric. Server tidak menerima, menyimpan, atau memproses kunci privat Fabric dalam bentuk *plaintext*. Untuk memperjelas data yang dikelola oleh server, struktur penyimpanan utama pada basis data MySQL ditunjukkan pada
Tabel ***1***.

Tabel 1. Struktur Penyimpanan Data pada Basis Data MySQL

Komponen Data	Lokasi Penyimpanan	Fungsi
Sertifikat X.509	fabric_identities	Identitas Fabric pengguna
Kunci privat terenkripsi	fabric_identities	Material kunci Fabric dalam bentuk *ciphertext*
DEK terenkripsi	fabric_identities	Kunci data yang telah dibungkus menggunakan KEK
Credential ID WebAuthn	fabric_identities / user_credentials	Relasi dengan kredensial WebAuthn

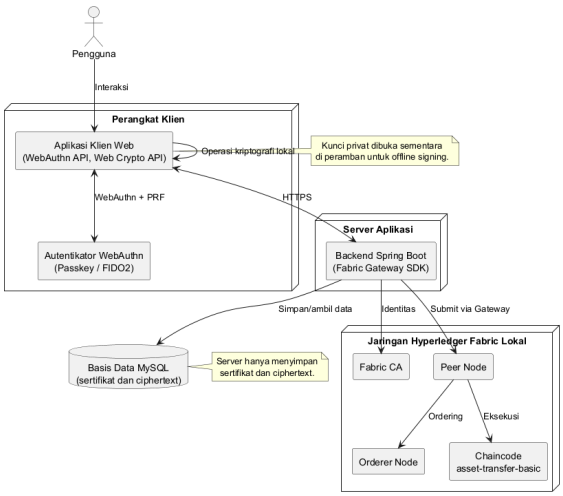
This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

Public key WebAuthn	user_credentials	Verifikasi autentikasi WebAuthn
Initialization Vector dan *Authentication Tag*	Kolom *ciphertext* JSON	Metadata untuk dekripsi AES-GCM

Basis data MySQL digunakan sebagai media penyimpanan identitas Fabric dan kredensial WebAuthn pengguna. Secara konseptual, data yang disimpan terbagi menjadi dua kelompok. Kelompok pertama adalah identitas Fabric, yang mencakup sertifikat X.509, kunci privat Fabric terenkripsi, DEK terenkripsi, dan metadata dekripsi seperti *Initialization Vector* (IV) serta *Authentication Tag*. Kelompok kedua adalah kredensial WebAuthn, yang mencakup credential ID dan public key WebAuthn. Pemisahan ini bertujuan untuk menjaga agar server hanya menyimpan data yang diperlukan untuk autentikasi, pemulihan *ciphertext*, dan pengiriman transaksi, tanpa menyimpan kunci privat Fabric dalam bentuk *plaintext*.

Jaringan Hyperledger Fabric lokal terdiri atas Fabric CA, peer, orderer, dan *chaincode* asset-transfer-basic sebagai objek transaksi uji. Fabric CA digunakan untuk menerbitkan sertifikat X.509 berdasarkan CSR yang dibuat dari sisi klien. Peer dan orderer digunakan untuk memproses transaksi yang telah ditandatangani melalui mekanisme *offline signing*. *Chaincode* asset-transfer-basic digunakan sebagai skenario transaksi sederhana untuk memverifikasi bahwa tanda tangan yang dibuat di peramban dapat diterima dan divalidasi oleh jaringan Fabric. Hubungan antar komponen pada arsitektur ini ditunjukkan pada Gambar 2.

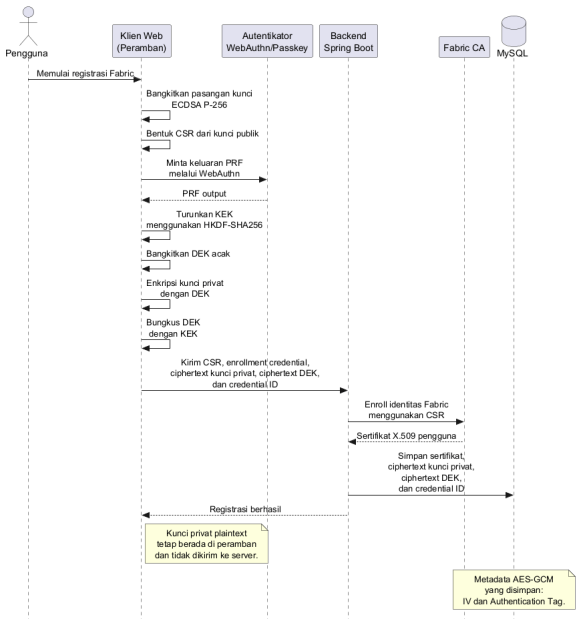


Gambar 2. Diagram Arsitektur Sistem yang Diusulkan

Berdasarkan Gambar 2, alur komunikasi utama dimulai dari klien web yang berinteraksi dengan autentikator dan API kriptografi peramban, kemudian berkomunikasi dengan server aplikasi melalui HTTPS. Server aplikasi berperan sebagai penghubung ke basis data dan jaringan Hyperledger Fabric, sedangkan operasi yang melibatkan kunci privat tetap dilakukan pada sisi klien. Pemisahan ini menjadi dasar rancangan non-kustodian yang digunakan dalam penelitian.

Alur Registrasi dan Pembungkusan Kunci

Alur registrasi dan pembungkusan kunci menggambarkan proses pembentukan identitas Fabric pengguna sekaligus perlindungan kunci privat di sisi klien sebelum data dikirim ke server. Alur ini mengasumsikan bahwa kredensial WebAuthn pengguna telah dibuat pada proses terpisah dengan dukungan ekstensi PRF, serta pengguna telah memperoleh enrollment ID dan enrollment secret dari tahap pra-registrasi. Oleh karena itu, **Gambar 3** tidak menampilkan proses pembuatan *passkey* dari awal, melainkan merangkum alur registrasi identitas Fabric, pembungkusan kunci privat, enrollment ke Fabric CA, dan penyimpanan hasil enkripsi pada basis data.



Gambar 3. Alur Registrasi dan Pembungkusan Kunci

Berdasarkan Gambar 3, proses dimulai ketika pengguna memasukkan enrollment ID dan enrollment secret pada aplikasi klien web. Setelah itu, peramban membangkitkan pasangan kunci ECDSA kurva P-256 sebagai kunci penandatangan Fabric. Kunci publik digunakan untuk membentuk CSR, sedangkan kunci privat tetap berada di sisi klien dan tidak dikirim ke server dalam bentuk *plaintext*. Sebelum data dikirim ke

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

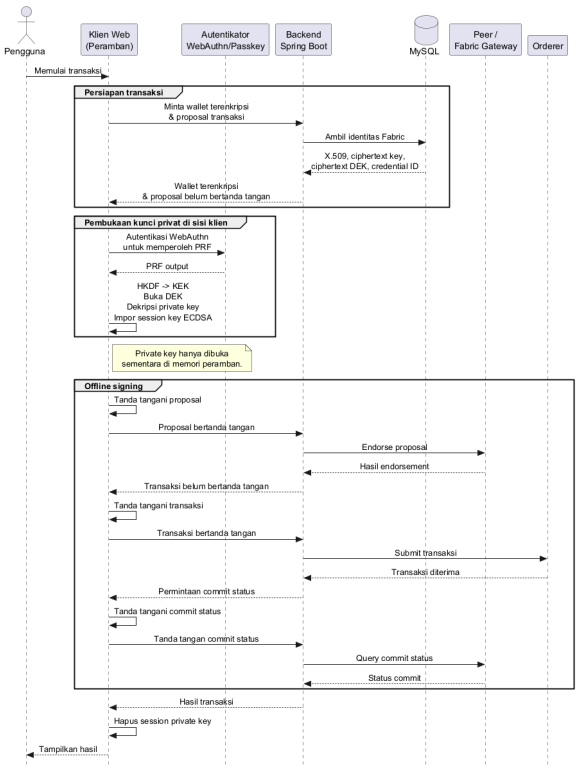
server, kunci privat Fabric dalam format PKCS#8 dienkripsi menggunakan AES-256-GCM dengan *Data Encryption Key* (DEK) acak sepanjang 32 byte. Pada tahap berikutnya, klien meminta evaluasi PRF melalui mekanisme WebAuthn terhadap kredensial yang sudah terdaftar. Keluaran PRF diturunkan melalui HKDF-SHA256 untuk menghasilkan *Key Encryption Key* (KEK), lalu KEK digunakan untuk membungkus DEK menggunakan AES-256-GCM.

Setelah kunci privat dan DEK berada dalam bentuk *ciphertext*, klien mengirim CSR, enrollment ID, enrollment secret, *ciphertext* kunci privat, *ciphertext* DEK, dan credential ID ke server. Server kemudian meneruskan CSR bersama enrollment credential ke Fabric CA melalui proses enrollment. Fabric CA menerbitkan sertifikat X.509 pengguna, kemudian server menyimpan sertifikat tersebut bersama *ciphertext* kunci privat, *ciphertext* DEK, dan relasi kredensial WebAuthn pada basis data. Dengan alur ini, server hanya menerima material kunci dalam bentuk terenkripsi dan tidak pernah menerima kunci privat Fabric dalam bentuk *plaintext*.

Salt untuk WebAuthn PRF tidak disimpan di server, tetapi didefinisikan secara statis pada sisi klien dengan nilai fabric-enrollment-salt-evaluation-key. Pada proses derivasi HKDF-SHA256, salt yang digunakan berupa array byte kosong sepanjang 32 byte. Dengan demikian, metadata yang disimpan bersama *ciphertext* pada basis data hanya berkaitan dengan proses AES-GCM, yaitu IV dan *Authentication Tag*.

Alur Transaksi dan Offline Signing

Alur transaksi dan *offline signing* menggambarkan proses penggunaan kunci privat Fabric yang telah dilindungi pada tahap registrasi untuk menandatangani transaksi dari sisi klien. Secara umum, alur ini mencakup pengambilan *wallet* terenkripsi dan proposal transaksi, pembukaan kunci privat melalui keluaran PRF WebAuthn, penandatanganan pada tiga titik (proposal, transaksi, dan permintaan *commit status*) yang seluruhnya dilakukan di peramban, pengiriman transaksi melalui server menuju jaringan Fabric, pemeriksaan *commit status*, serta pembersihan material kunci sementara dari memori klien. Rangkaian proses tersebut ditunjukkan pada Gambar 4, dimana proses dimulai ketika pengguna menjalankan transaksi melalui aplikasi klien web. Klien mengambil *wallet* terenkripsi dari server, yang berisi sertifikat X.509, *ciphertext* kunci privat, *ciphertext* DEK, dan *credential ID*. Setelah itu, klien meminta proposal transaksi yang belum bertanda tangan kepada server. Proposal ini disiapkan berdasarkan identitas pengguna, tetapi proses penandatanganannya tetap dilakukan di sisi klien.



Gambar 4. Alur Transaksi dan Offline Signing

Untuk membuka kunci privat, pengguna melakukan autentikasi WebAuthn terhadap kredensial yang sama dengan tahap registrasi. Keluaran PRF yang diperoleh dari autentikator diturunkan kembali menggunakan HKDF-SHA256 untuk menghasilkan KEK. KEK digunakan untuk membuka DEK, lalu DEK digunakan untuk mendekripsi kunci privat Fabric. Kunci privat yang telah terbuka kemudian diimpor sebagai *session key* ECDSA dan hanya digunakan sementara di memori peramban untuk proses *offline signing*. Autentikasi WebAuthn hanya dilakukan satu kali; *session key* yang sama digunakan untuk ketiga penandatanganan berikutnya tanpa autentikasi ulang.

Setelah kunci privat tersedia di sisi klien, klien menandatangani proposal transaksi menggunakan ECDSA dengan SHA-256 dan mengirimkan proposal bertanda tangan ke server untuk proses *endorsement*. Setelah endorsement diterima, server mengirimkan transaksi yang belum bertanda tangan kepada klien. Klien kembali menandatangani transaksi tersebut dan mengirimkannya ke server untuk diteruskan ke *orderer*. Pada tahap akhir, klien menandatangani permintaan *commit status* untuk memastikan transaksi telah diproses dan divalidasi oleh jaringan Fabric. Setelah seluruh proses selesai, referensi terhadap *session private key* dan material kunci sementara dihapus dari variabel aplikasi sejauh yang dimungkinkan oleh lingkungan peramban.

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

Lingkungan dan Objek Uji

Lingkungan pengujian disusun untuk merepresentasikan eksekusi purwarupa secara lokal pada sisi klien, server aplikasi, basis data, dan jaringan Hyperledger Fabric. Klien web dikembangkan menggunakan Flutter Web dan dijalankan sebagai hasil kompilasi web pada peramban, dengan akses ke WebAuthn API dan Web Crypto API melalui Dart interop. Server aplikasi dikembangkan menggunakan Java Spring Boot dan berkomunikasi dengan jaringan Fabric melalui Fabric Gateway client. Objek transaksi uji menggunakan *chaincode* asset-transfer-basic dari repository fabric-samples, sedangkan autentikator yang diuji mencakup Proton Pass, Google Password Manager, iCloud Keychain, dan Windows Hello untuk merepresentasikan penyedia *passkey* tersinkron serta autentikator platform. Rincian lingkungan dan objek uji yang akan digunakan ditunjukkan pada
Tabel ***2***.

Tabel 2. Lingkungan dan Objek Uji

Komponen	Spesifikasi
Perangkat klien	Laptop (Intel® Core™ i7-10870H @ 2.20GHz, RAM 16.0 GB)
Sistem operasi	Windows 11 (64-bit)
Peramban	Edge, versi 149.0.4022.62 (Beta Channel, 64-bit)
Frontend	Dart 3.11.0, Flutter 3.41.0
Backend	Java 21, Spring Boot 4.0.5, Fabric Gateway client 1.11.0
Basis data	MySQL 8.0
Blockchain	Hyperledger Fabric 2.5.11, Fabric CA 1.5.15
Chaincode	asset-transfer-basic (dari repository fabric-samples)
Autentikator	Proton Pass, Google Password Manager, iCloud Keychain, Windows Hello

Berdasarkan
Tabel ***2***, pengujian dilakukan pada lingkungan lokal agar proses validasi fungsional dan pengukuran overhead kriptografi dapat dikendalikan tanpa dipengaruhi latensi jaringan eksternal. Pemilihan Proton Pass, Google Password Manager, iCloud Keychain, dan Windows Hello bertujuan untuk membandingkan perilaku penyedia *passkey* tersinkron dan autentikator platform dalam mendukung keluaran PRF WebAuthn.

Skenario Pengujian

Skenario pengujian disusun untuk mengevaluasi kelayakan fungsional, kompatibilitas autentikator, dan overhead kriptografi dari purwarupa. Ringkasan skenario, data yang dicatat, dan indikator keberhasilan ditunjukkan pada
Tabel ***3***.

Tabel 3. Ringkasan Skenario Pengujian

Skenario	Tujuan	Data yang Dicatat	Indikator Keberhasilan
Enrollment dan pembungkusan kunci	Memastikan identitas Fabric dapat dibuat dan kunci privat tersimpan terenkripsi	Sertifikat X.509, *ciphertext* kunci privat, *ciphertext* DEK, log registrasi	Sertifikat diterbitkan dan kunci privat tidak tersimpan *plaintext*
Transaksi dan *offline signing*	Memastikan transaksi dapat ditandatangani di peramban dan diterima Fabric	Transaction ID, status endorsement, commit status, block number	Transaksi valid dan tercatat pada ledger
Kompatibilitas PRF autentikator	Menilai dukungan PRF pada Proton Pass, Google Password Manager, iCloud Keychain, dan Windows Hello	Status PRF, pesan error, jenis autentikator	Autentikator mengembalikan PRF output yang dapat dipakai derivasi KEK
Overhead kriptografi sisi klien	Mengukur biaya tambahan proses kriptografi di *browser*	Waktu derivasi KEK, dekripsi DEK, dekripsi kunci privat, impor kunci, signing ECDSA	Operasi berhasil dan durasi tiap tahap tercatat

Berdasarkan
Tabel ***3***, pengujian fungsional dibagi menjadi dua bagian, yaitu registrasi dan pembungkusan kunci serta transaksi dan *offline signing*. Pemisahan ini dilakukan karena proses registrasi berfokus pada pembentukan identitas Fabric dan penyimpanan kunci privat dalam bentuk terenkripsi, sedangkan proses transaksi berfokus pada pembukaan kunci, penandatanganan dari sisi klien, dan validasi transaksi oleh jaringan Fabric. Pengujian kompatibilitas autentikator dilakukan untuk memastikan apakah autentikator yang digunakan mampu menghasilkan keluaran PRF yang dapat dipakai dalam proses derivasi KEK. Sementara itu, pengujian overhead kriptografi

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

dilakukan hanya pada operasi kriptografi sisi klien, sehingga waktu interaksi pengguna seperti PIN, biometrik, atau konfirmasi autentikator tidak dicampurkan dengan waktu komputasi kriptografi murni.

Teknik Pengumpulan dan Analisis Data

Data penelitian dikumpulkan dari tiga sumber utama, yaitu respons aplikasi, log jaringan Hyperledger Fabric, dan hasil pengukuran waktu pada sisi klien. Data fungsional diperoleh dari respons aplikasi, transaction ID, status endorsement, commit status, block number, serta log peer Fabric untuk memastikan transaksi diterima dan tervalidasi dengan identitas pengguna yang tepat. Data kompatibilitas diperoleh dari hasil pengujian masing-masing autentikator, terutama status keberhasilan pengembalian keluaran PRF dan pesan kesalahan apabila proses tidak didukung atau gagal dijalankan. Data kinerja dikumpulkan pada sisi klien menggunakan performance.now() untuk mencatat waktu mulai dan selesai setiap operasi kriptografi yang diuji. Selain itu, data verifikasi penyimpanan dikumpulkan melalui inspeksi langsung terhadap isi tabel fabric_identities pada basis data serta payload permintaan HTTP yang dikirim peramban, untuk memastikan material kunci yang tersimpan maupun terkirim hanya berupa *ciphertext* beserta metadata AES-GCM.

Pengukuran overhead kriptografi difokuskan pada operasi sisi klien, yaitu derivasi KEK menggunakan HKDF-SHA256, dekripsi DEK, dekripsi kunci privat Fabric, impor kunci privat ke Web Crypto API, dan penandatanganan ECDSA. Waktu interaksi pengguna dengan autentikator, seperti PIN, biometrik, atau konfirmasi *passkey*, tidak dicampurkan ke dalam waktu komputasi kriptografi murni. Pengukuran dilakukan berulang sebanyak 30 (tiga puluh) kali pada setiap operasi yang berhasil dijalankan.

Analisis data fungsional dan kompatibilitas dilakukan secara kualitatif dengan membandingkan bukti yang terkumpul terhadap indikator keberhasilan pada

Tabel *3*, dan hasil kompatibilitas dirangkum sebagai matriks status dukungan PRF per autentikator. Analisis data kinerja dilakukan secara deskriptif menggunakan rerata, simpangan baku, nilai minimum, dan nilai maksimum. Statistik tersebut digunakan untuk melihat kecenderungan waktu eksekusi, variasi antarpercobaan, dan besaran overhead tambahan yang ditimbulkan oleh mekanisme perlindungan kunci. Hasil pengukuran overhead kriptografi disajikan dalam format tabel agar setiap operasi dapat dibandingkan secara langsung, beserta total overhead kriptografi sisi klien yang dihitung sebagai penjumlahan rerata seluruh operasi. Pemetaan menyeluruh antara aspek pengujian, data dan sumbernya, serta teknik analisis yang digunakan dirangkum pada **Tabel 4**. Format penyajian hasil pengukuran overhead kriptografi ditunjukkan pada Tabel 6.

Tabel 4. Pemetaan Pengumpulan dan Analisis Data Penelitian

Aspek Pengujian	Data dan Sumber	Teknik Analisis
Fungsional (registrasi, pembungkusan kunci, transaksi, *offline signing*)	Respons aplikasi, transaction ID, status endorsement, commit status, block number, dan log peer Fabric	Kualitatif terhadap indikator keberhasilan pada **Tabel *3***
Kompatibilitas autentikator	Status keberhasilan keluaran PRF dan pesan kesalahan pada masing-masing autentikator	Kualitatif; dirangkum sebagai matriks dukungan PRF per autentikator
Verifikasi penyimpanan non-kustodian	Inspeksi tabel fabric_identities pada basis data dan payload permintaan HTTP peramban	Kualitatif; konfirmasi material kunci hanya berupa *ciphertext* beserta metadata AES-GCM
Kinerja (overhead kriptografi sisi klien)	Waktu operasi kriptografi (derivasi KEK HKDF-SHA256, dekripsi DEK, dekripsi kunci privat Fabric, impor kunci ke Web Crypto API, penandatanganan ECDSA) menggunakan performance.now(), 30 pengulangan	Deskriptif: rerata, simpangan baku, minimum, maksimum; total overhead dihitung dari penjumlahan rerata setiap operasi

HASIL DAN PEMBAHASAN

Pembuktian Konsep Fungsional

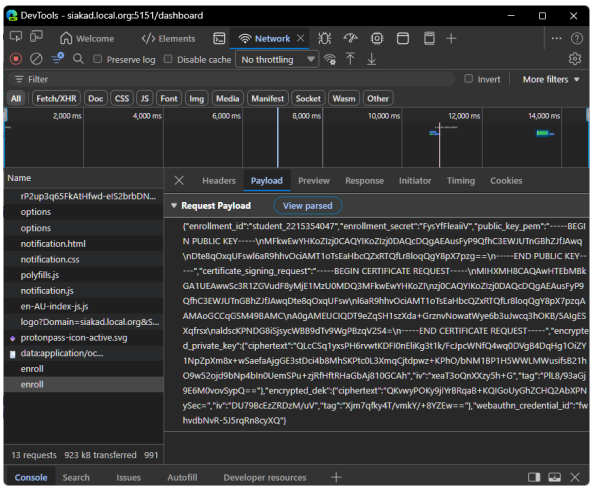
Purwarupa berhasil menjalankan alur lengkap dari enrollment identitas pengguna hingga pengiriman transaksi asset-transfer ke jaringan. Untuk membuktikan bahwa setiap tahap berjalan sebagaimana dirancang, alur fungsional diuji melalui serangkaian request dan response yang direkam langsung dari peramban, mulai dari pendaftaran identitas hingga konfirmasi bahwa transaksi telah masuk ke dalam ledger. Rangkaian bukti pengujian disajikan secara

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

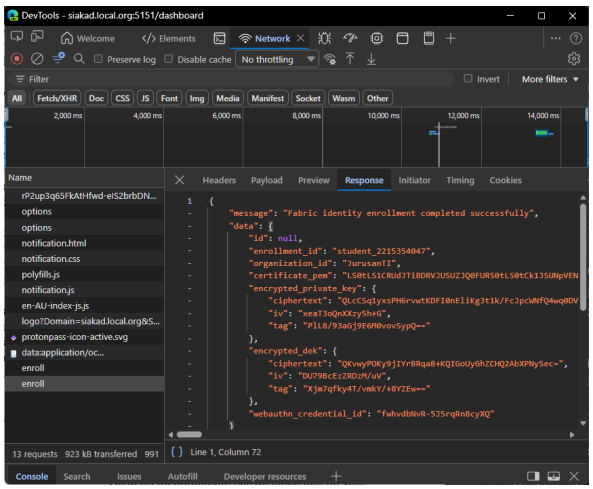
berurutan melalui **Gambar 5** hingga **Gambar 15**, sehingga setiap tahap dapat ditelusuri berdasarkan artefak request, response, dan log jaringan yang dihasilkan.

Tahap pertama adalah enrollment identitas. Klien mengirimkan permintaan ke endpoint enroll dengan menyertakan enrollment_id, kunci publik dalam format PEM, serta kunci privat dan DEK yang telah terenkripsi, sebagaimana ditunjukkan pada **Gambar 5**.



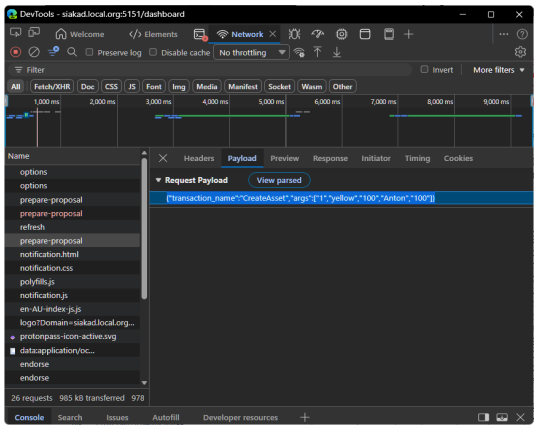
Gambar 5. Permintaan Enrollment ke Endpoint Enroll

Server merespons dengan pesan keberhasilan "Fabric identity enrollment completed successfully" sebagaimana diperlihatkan pada **Gambar 6**. Respons ini menandakan bahwa sertifikat X.509 pengguna telah diterbitkan oleh Fabric CA tanpa kunci privat *plaintext* pernah dikirim ke server.



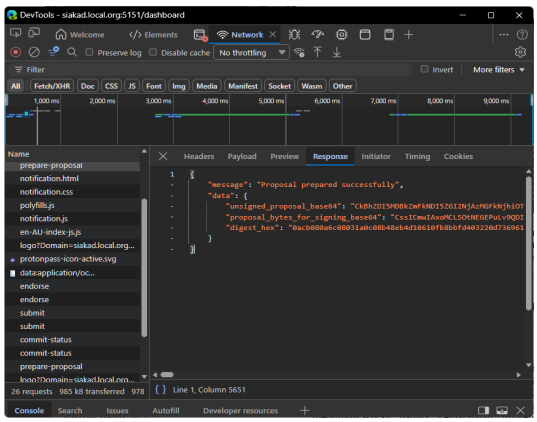
Gambar 6. Tanggapan Keberhasilan Enrollment Identitas Fabric

Gambar 7 memperlihatkan tahap persiapan proposal transaksi. Pada tahap ini, permintaan prepare-proposal memanggil fungsi CreateAsset pada *chaincode* asset-transfer-basic.



Gambar 7. Permintaan prepare-proposal untuk fungsi CreateAsset

Sebagaimana ditunjukkan pada **Gambar 8**, server mengembalikan proposal yang belum ditandatangani beserta digest yang harus ditandatangani pengguna.

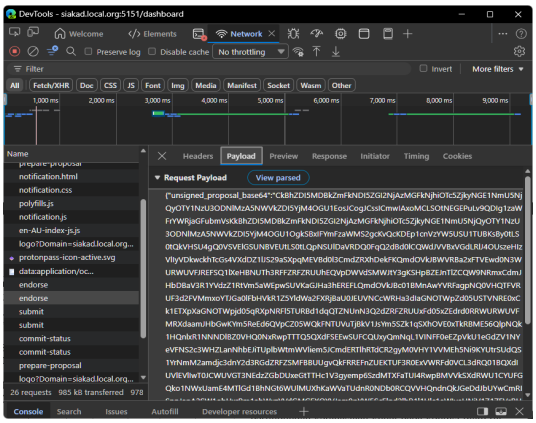


Gambar 8. Tanggapan prepare-proposal berisi proposal dan digest

Gambar 9 menunjukkan tahap endorsement, ketika klien membuka kunci privatnya di memori peramban, menandatangani digest proposal, lalu mengirimkannya kembali ke server.

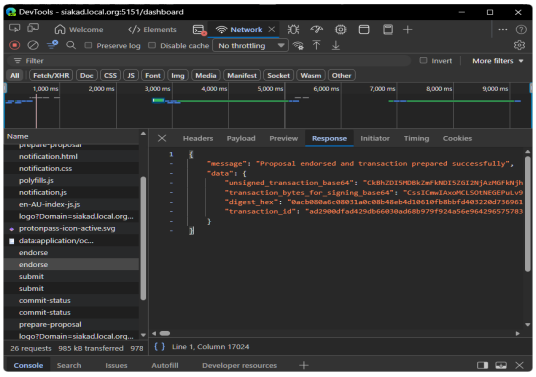
This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)



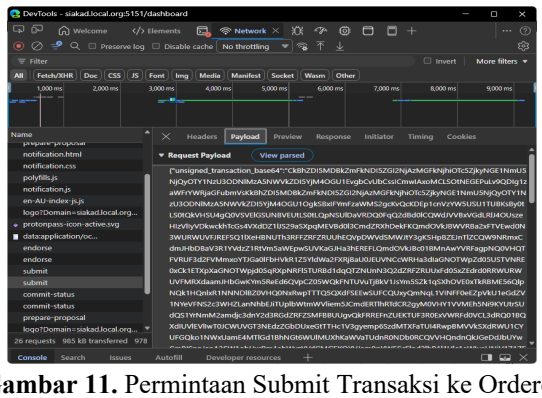
Gambar 9. Permintaan Endorse Dengan Proposal Bertanda Tangan Klien

Server meneruskan proposal tersebut ke peer untuk proses endorsement dan menyiapkan transaksi. Setelah proses tersebut berhasil, server merespons dengan pesan "Proposal endorsed and transaction prepared successfully" sebagaimana ditunjukkan pada **Gambar 10**.

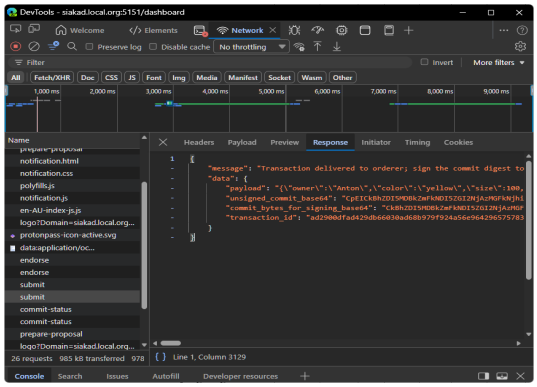


Gambar 10. Tanggapan Keberhasilan Endorsement dan Pembentukan Transaksi

Alur pengiriman transaksi yang telah ditandatangani ke orderer melalui endpoint submit ditunjukkan pada **Gambar 11**. Setelah itu, server mengembalikan respons yang meminta klien menandatangani commit digest sebagaimana ditunjukkan pada **Gambar 12**.

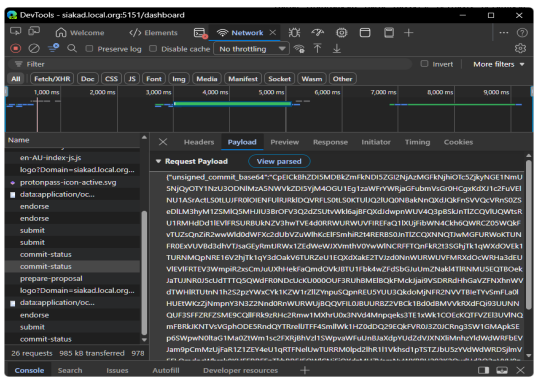


Gambar 11. Permintaan Submit Transaksi ke Orderer

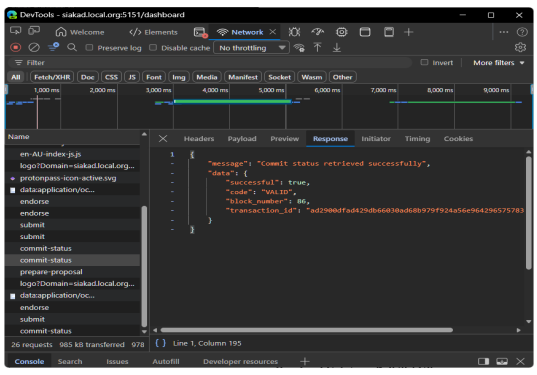


Gambar 12. Tanggapan Submit Dengan Instruksi Penandatanganan Commit Digest

Setelah commit digest ditandatangani, proses commit-status digunakan untuk mengonfirmasi keberhasilan transaksi. **Gambar 13** menunjukkan permintaan commit-status, sedangkan **Gambar 14** memperlihatkan respons yang menunjukkan nilai successful bernilai true, kode VALID, dan transaksi tercatat pada block_number 86.



Gambar 13. Permintaan commit-Status



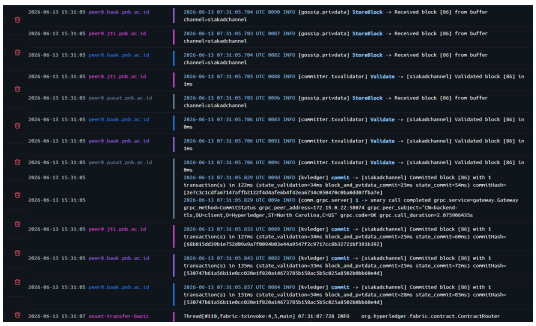
Gambar 14. Tanggapan Commit-Status

Keberhasilan ini diverifikasi pula pada sisi jaringan melalui catatan log peer Fabric. Sebagaimana ditunjukkan pada **Gambar 15**, block [86] berisi satu transaksi yang telah di-commit dengan status VALID pada seluruh peer. Dengan demikian, transaksi yang tanda tangannya dibentuk sepenuhnya di sisi klien

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

ISSN: 2598-8565 (media cetak)
ISSN: 2620-4339 (media online)

terbukti diterima dan dimasukkan ke dalam ledger oleh masing-masing peer.



Gambar 15. Log Peer Fabric: block [86] Ter-commit dengan Status VALID

Hasil ini menunjukkan bahwa tanda tangan yang dibentuk di peramban dapat divalidasi terhadap sertifikat X.509 pengguna, sehingga membuktikan bahwa *offline signing* Fabric dapat dilakukan sepenuhnya dari sisi klien. Dari sisi penyimpanan, server hanya memuat *ciphertext* (kolom encrypted_private_key dan encrypted_dek) tanpa pernah menyimpan kunci privat *plaintext*, sehingga memenuhi definisi non-kustodian yang digunakan dalam penelitian ini.

Kompatibilitas Autentikator

Kompatibilitas autentikator dievaluasi berdasarkan kemampuan masing-masing autentikator dalam mengembalikan keluaran PRF yang dapat digunakan untuk proses derivasi KEK. Pengujian dilakukan pada tahap registrasi dan transaksi karena kedua tahap tersebut membutuhkan keluaran PRF untuk membungkus atau membuka DEK. Ringkasan hasil kompatibilitas ditunjukkan pada **Tabel 5**.

Tabel 5. Matriks Kompatibilitas Autentikator WebAuthn PRF

Autentikator	Jenis Autentikator	Registrasi PRF	Transaksi PRF	Keterangan
Proton Pass	Penyedia *passkey* tersinkron	Berhasil	Berhasil	Mengembalikan keluaran PRF saat pendaftaran dan dapat digunakan untuk derivasi KEK saat autentikasi
Google Password Manager	Penyedia *passkey* tersinkron	Berhasil	Berhasil	Mengembalikan keluaran PRF saat pendaftaran dan dapat digunakan untuk derivasi KEK saat autentikasi
Windows Hello	Autentikator platform	Berhasil	Gagal	Mengembalikan keluaran PRF saat pendaftaran namun gagal mengembalikan nilai keluaran PRF saat autentikasi
iCloud Keychain	Penyedia *passkey* tersinkron	Berhasil	Berhasil	Mengembalikan keluaran PRF saat pendaftaran dan dapat digunakan untuk derivasi KEK saat autentikasi

Berdasarkan **Tabel 5**, ketiga penyedia *passkey* tersinkron yang diuji (Proton Pass, Google Password Manager, dan iCloud Keychain) mengembalikan keluaran PRF baik pada tahap registrasi maupun transaksi sehingga dapat digunakan dalam alur non-kustodian yang diusulkan. Sebaliknya, Windows Hello sebagai autentikator platform berhasil mengembalikan keluaran PRF saat registrasi tetapi gagal pada tahap autentikasi (transaksi) sehingga tidak dapat menyelesaikan pembukaan DEK meskipun tampak mendukung PRF pada saat pendaftaran. Temuan ini menunjukkan bahwa dukungan PRF tidak dapat diasumsikan seragam antar-autentikator dan perlu diverifikasi secara khusus pada tahap autentikasi; pada periode pengujian, dukungan yang andal justru berasal dari penyedia *passkey* tersinkron, bukan dari autentikator platform yang diuji. Secara teknis, kegagalan ini terjadi karena autentikator platform yang diuji mengekspos kemampuan PRF (hmac-secret) pada tahap registrasi (navigator.credentials.create), tetapi tidak mengembalikan nilai keluaran PRF pada tahap autentikasi (navigator.credentials.get); dengan kata lain, kapabilitas PRF tampak didukung saat pendaftaran namun belum berfungsi penuh pada tahap evaluasi *assertion* untuk kombinasi sistem operasi dan versi peramban yang diuji. Karena pembukaan DEK bergantung pada keluaran PRF di tahap autentikasi, ketidaktersediaan nilai tersebut menyebabkan alur non-kustodian tidak dapat diselesaikan pada Windows Hello.

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Overhead Kriptografi di Sisi Klien

Bagian ini mengukur overhead kriptografi yang ditambahkan oleh skema di sisi klien, yaitu waktu komputasi murni untuk membuka dan menggunakan kunci privat pada satu kali penandatanganan transaksi. Pengukuran ini sengaja dipisahkan dari latensi interaksi autentikator dan verifikasi pengguna yang didominasi faktor manusia, sehingga angka yang dilaporkan hanya mencerminkan biaya operasi kriptografi itu sendiri. Lima tahap yang diukur adalah derivasi KEK, dekripsi DEK, dekripsi kunci privat, impor kunci ke Web Crypto API, dan penandatanganan ECDSA; rangkuman hasilnya disajikan pada **Tabel 6**.

Tabel 6. Hasil Pengukuran Overhead Kriptografi Sisi Klien

Operasi Kriptografi	Rerata (ms)	Median (ms)	Simpangan Baku (ms)	Minimum (ms)	Maksimum (ms)	p95 (ms)	Keterangan
Derivasi KEK (HKDF-SHA256)	0,038	0,036	0,008	0,032	0,067	0,052	PRF output diturunkan menjadi KEK
Dekripsi DEK (AES-256-GCM)	0,018	0,016	0,005	0,014	0,035	0,029	DEK dibuka menggunakan KEK
Dekripsi kunci privat Fabric (AES-256-GCM)	0,018	0,017	0,003	0,015	0,027	0,023	Kunci privat PKCS#8 dibuka menggunakan DEK
Impor kunci privat ke Web Crypto API	0,089	0,086	0,013	0,079	0,152	0,102	PKCS#8 diimpor sebagai kunci ECDSA P-256
Penandatanganan ECDSA	0,071	0,070	0,006	0,064	0,091	0,081	Signing menggunakan ECDSA dengan SHA-256
Total overhead kriptografi sisi klien	0,248	0,241	0,022	0,217	0,306	0,292	Diukur ujung-ke-ujung pada jalur penuh

Jika diuraikan per operasi, biaya komputasi tidak terbagi rata. Bagian terbesar berasal dari dua operasi berbasis kurva eliptik, yaitu impor kunci privat ke Web Crypto API (0,089 ms) dan penandatanganan ECDSA (0,071 ms), yang bersama-sama mencakup sekitar dua pertiga (kurang lebih 65%) dari total biaya. Dua proses dekripsi AES-256-GCM justru yang paling ringan (masing-masing sekitar 0,018 ms), sementara derivasi KEK dengan HKDF-SHA256 berada di antara keduanya (0,038 ms). Hasil pengukuran juga stabil: nilainya hampir tidak berubah dari satu percobaan ke percobaan berikutnya, dengan simpangan baku total hanya 0,022 ms dan 95% pengukuran berada di bawah 0,292 ms—dekat dengan nilai tengahnya (median 0,241 ms).

Angka pengujian ini terikat pada satu lingkungan uji, yaitu kombinasi peramban, sistem operasi, dan perangkat tertentu, sehingga besarannya bisa berbeda di perangkat lain—terutama ponsel kelas bawah. Meskipun begitu, pola umumnya diperkirakan tetap sama: operasi ECDSA yang paling menentukan biaya, dan total overhead-nya tetap jauh lebih kecil daripada waktu yang dibutuhkan untuk interaksi pengguna. Dengan kata lain, kesimpulan bahwa skema ini layak untuk penggunaan interaktif tidak bergantung pada perangkat keras tertentu yang diuji.

Secara umum, beban komputasi kriptografi pada operasi-operasi tersebut tergolong kecil dibandingkan latensi interaksi autentikator dan verifikasi pengguna yang didominasi faktor manusia. Dengan demikian, overhead yang ditambahkan oleh skema ini di sisi klien dapat dipandang layak untuk penggunaan interaktif; pengukuran ini ditujukan sebagai bukti kelayakan, bukan sebagai klaim keunggulan keamanan.

Implikasi praktis dari hasil ini cukup jelas. Total overhead rerata sebesar 0,248 ms berada jauh di bawah ambang persepsi manusia, sehingga tidak dirasakan oleh pengguna pada penggunaan interaktif. Sebagian besar biaya tersebut berasal dari operasi ECDSA yang memang inheren pada penandatanganan Hyperledger Fabric, sedangkan komponen yang ditambahkan oleh skema perlindungan kunci—derivasi HKDF-SHA256 dan dekripsi AES-256-GCM—hanya menyumbang sebagian kecil dari total. Dengan demikian, mekanisme pembungkusan kunci non-kustodian tidak menambah beban yang berarti dibandingkan alur penandatanganan Fabric biasa. Mengingat Hyperledger Fabric dirancang

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

untuk throughput tinggi, yaitu lebih dari 3.500 transaksi per detik pada konfigurasi tertentu (Androulaki et al., 2018), overhead sebesar ini di sisi klien tidak menjadi penghambat kinerja jaringan secara keseluruhan.

Jika dikaitkan dengan penelitian terdahulu, hasil ini menjawab persoalan yang telah lama diidentifikasi. Studi tentang manajemen kunci menunjukkan bahwa pengguna awam kerap kesulitan menyimpan, mencadangkan, dan memulihkan kunci atau frasa pemulihan secara aman (Eskandari et al., 2015; Sharma et al., 2024). Pendekatan yang diusulkan mengurangi beban tersebut karena pengguna tidak perlu mengelola berkas kunci mentah secara langsung; kunci pembungkus diturunkan secara deterministik dari keluaran PRF autentikator pada saat autentikasi. Berbeda dengan dompet non-kustodian berbasis cloud yang menyimpan material kunci pada layanan tersinkron (Stamatopoulou et al., 2025), pendekatan ini menyimpan kunci privat Fabric di server hanya dalam bentuk terenkripsi dan membukanya sementara di sisi klien sehingga server tidak pernah memegang kunci dalam bentuk plaintext. Dengan demikian, temuan fungsional dan kinerja pada penelitian ini memperkuat kelayakan manajemen kunci non-kustodian yang sebelumnya banyak terkendala aspek kegunaan.

Diskusi Keamanan dan Batasan

Properti keamanan utama yang dicapai adalah perlindungan kerahasiaan kunci pada kondisi diam (*at rest*): apabila basis data atau server bocor secara pasif, yang diperoleh penyerang hanyalah *ciphertext*, sementara untuk membuka kunci diperlukan keluaran PRF yang hanya dapat dihasilkan melalui autentikator pengguna dengan verifikasi pengguna. Namun, arsitektur ini tidak melindungi terhadap sejumlah ancaman lain yang perlu dinyatakan secara jujur. Karena dekripsi dan penandatanganan terjadi di peramban menggunakan kode yang dilayani oleh server, penyerang yang berhasil mengompromikan integritas *frontend* dapat mencuri kunci pada saat kunci tersebut didekripsi; demikian pula *malware* pada perangkat klien atau pembacaan memori peramban berpotensi mengakses kunci selama jendela penggunaannya. Selain itu, pada konfigurasi uji ini—yang menggunakan *passkey* tersinkron—keamanan kunci pembungkus pada akhirnya bergantung pada keamanan akun penyedia *passkey* pengguna beserta mekanisme sinkronisasinya, bukan pada kepemilikan perangkat keras. Oleh karena itu, penelitian ini tidak mengklaim properti keamanan berbasis perangkat keras. Terdapat pula *trade-off* yang melekat: *passkey* tersinkron meningkatkan kegunaan dan kemudahan pemulihan multi-perangkat, tetapi memperluas permukaan serangan dibandingkan autentikator yang terikat perangkat keras.

Untuk menilai postur keamanan arsitektur secara lebih sistematis, ancaman dianalisis menggunakan kerangka STRIDE (*Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, dan Elevation of Privilege*). Analisis ini merangkum properti keamanan yang dicapai beserta risiko residual yang telah diuraikan di atas, sebagaimana ditunjukkan pada
Tabel *7*.

Tabel 7. Analisis Ancaman Keamanan Berdasarkan Kerangka STRIDE

Kategori STRIDE	Ancaman pada Sistem	Mitigasi / Risiko Residual
Spoofing (pemalsuan identitas)	Penyerang menyamar sebagai pengguna sah untuk memicu penggunaan kunci.	Autentikasi WebAuthn tahan phishing dengan verifikasi pengguna; passkey terikat pada origin.
Tampering (perusakan data)	Modifikasi ciphertext kunci di server atau pengubahan isi transaksi.	AES-256-GCM mendeteksi perubahan ciphertext; integritas transaksi dijamin tanda tangan ECDSA dan validasi MSP Fabric.
Repudiation (penyangkalan)	Pengguna menyangkal telah menandatangani suatu transaksi.	Tanda tangan ECDSA berpasangan dengan sertifikat X.509 pengguna dan tercatat permanen pada ledger Fabric yang immutable.
Information Disclosure (kebocoran informasi)	Kebocoran kunci privat atau DEK dari server maupun saat pemrosesan.	Envelope encryption: server hanya menyimpan ciphertext; kunci dibuka sesaat di memori dan memerlukan keluaran PRF. Risiko residual: kompromi frontend atau malware/pembacaan memori klien.
Denial of Service (penolakan layanan)	Pengguna tidak dapat menandatangani karena autentikator atau keluaran PRF tidak tersedia.	Bergantung pada ketersediaan autentikator; pada passkey tersinkron, kehilangan akses akun penyedia menjadi risiko residual.
Elevation of Privilege (peningkatan hak akses)	Pihak tak berwenang atau server terkompromi memperoleh kemampuan menandatangani.	Penandatanganan memerlukan keluaran PRF dari autentikator pengguna sekaligus kunci ECDSA-nya; kompromi server pasif saja tidak cukup.

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

KESIMPULAN

Penelitian ini menunjukkan bahwa ekstensi PRF WebAuthn dapat diintegrasikan dengan mekanisme *offline signing* Hyperledger Fabric untuk membentuk arsitektur ketika server aplikasi tidak pernah memegang kunci privat pengguna dalam bentuk *plaintext*, dan transaksi yang sah tetap dapat dibentuk dari peramban. Purwarupa pada studi kasus asset-transfer-basic membuktikan kelayakan teknis arsitektur tersebut. Selain kelayakan fungsional, pengujian kuantitatif menunjukkan bahwa overhead kriptografi di sisi klien tergolong kecil—rerata 0,248 ms per penandatanganan—sehingga tidak menghambat penggunaan interaktif. Penelitian ini memiliki sejumlah batasan: penelitian ini merupakan pembuktian konsep tanpa studi kasus domain yang spesifik; keamanan kunci pembungkus bergantung pada penyedia *passkey* ketika menggunakan kredensial tersinkron; dukungan ekstensi PRF antar-autentikator belum seragam; pengukuran kinerja masih terbatas pada satu lingkungan uji sehingga besarannya dapat berbeda pada perangkat lain dan belum mencakup pengujian pada skala pengguna yang lebih besar; serta belum dilakukan analisis keamanan formal maupun perbandingan kinerja terhadap skema penandatanganan terpusat. Arah pengembangan selanjutnya meliputi penerapan pada kasus penggunaan yang lebih praktikal, dukungan multi-perangkat, serta analisis keamanan yang lebih menyeluruh.

DAFTAR PUSTAKA

Androulaki, E., Barger, A., Bortnikov, V., Muralidharan, S., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Murthy, C., Ferris, C., Laventman, G., Manevich, Y., Nguyen, B., Sethi, M., Singh, G., Smith, K., Sorniotti, A., Stathakopoulou, C., Vukolić, M., … Yellick, J. (2018). Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. *Proceedings of the 13th EuroSys Conference, EuroSys 2018*, *2018-January*. https://doi.org/10.1145/3190508.3190538

Barker, E. (2020). *Recommendation for key management: Part 1 – General*. https://doi.org/10.6028/NIST.SP.800-57pt1r5

Chen, X., He, S., Sun, L., Zheng, Y., & Wu, C. Q. (2024). A Survey of Consortium Blockchain and Its Applications. *Cryptography*, *8*(2), 12. https://doi.org/10.3390/cryptography8020012

Dworkin, M. J. (2007). *Recommendation for block cipher modes of operation.* https://doi.org/10.6028/NIST.SP.800-38d

Eskandari, S., Barrera, D., Stobert, E., & Clark, J. (2015). A First Look at the Usability of Bitcoin Key Management. *Proceedings 2015 Workshop on Usable Security*. https://doi.org/10.14722/usec.2015.23015

Hyperledger Foundation. (2026). *Security model - Fabric Applications - Hyperledger Fabric Documentation*.

Krawczyk, H., & Eronen, P. (2010). *HMAC-based Extract-and-Expand Key Derivation Function (HKDF)*. https://doi.org/10.17487/rfc5869

Krichen, M., Ammi, M., Mihoub, A., & Almutiq, M. (2022). Blockchain for Modern Applications: A Survey. In *Sensors* (Vol. 22, Number 14). MDPI. https://doi.org/10.3390/s22145274

Sharma, T., Nair, V. C., Wang, H., Wang, Y., & Song, D. (2024). "I Can't Believe It's Not Custodial!": Usable Trustless Decentralized Key Management. *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 1–16. https://doi.org/10.1145/3613904.3642464

Stamatopoulou, P., Koukoularis, E., Katimertzis, C.-M., Katrakazas, G., Voutsinas, N., & Lagos, A. I. (2025). *wwWallet.org: a Cloud Based Non-custodial Digital Identity Wallet.* 136–145. https://doi.org/10.29007/s3hw

W3C. (2021). *Web Authentication: An API for accessing public key credentials – Level 2*. https://www.w3.org/TR/webauthn-2/

W3C. (2025). *Web Authentication: An API for accessing public key credentials – Level 3*. https://www.w3.org/TR/webauthn-3/

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.