WHITEPAPER

The End of All-Access Banking:

Moving Toward a “Least Data” Banking Protocol

Version 1.0 RFC (Request for Comment)

This is a living document. Implementation feedback and technical contributions are invited.

Executive Summary

The current financial ecosystem relies on a structural security liability: "All-Access" data extraction. To verify a simple transaction, users often unknowingly expose their entire financial history to third-party apps and aggregators — far beyond what any single transaction requires.

The **Least Data Banking Protocol (LDBP)** is a bank-side open standard that replaces all-access data extraction with Boolean verification — returning only True or False to Finance Apps rather than raw account data.

Key Points:

1. The current open banking model requires Finance Apps to receive months of transaction history, exact account balances, and access to all accounts under a single credential set — data that has no bearing on whether a specific transaction should be authorized.
2. This architectural over-collection creates documented harm: in 2022, Plaid — the aggregator powering bank account linking for Venmo, Stripe, and Cash App — settled a $58 million class action lawsuit for collecting and profiting from user financial data without meaningful consent.
3. LDBP replaces data extraction with Boolean verification at the bank layer: instead of returning a raw balance, the bank's Policy Enforcement Point returns a single True or False — does this account hold sufficient funds for this specific transaction — and nothing else leaves the bank.
4. A bank implements LDBP once. Every Finance App connecting to it automatically operates under least-data principles without requiring any app-side protocol changes, making LDBP a general open standard rather than an app-specific solution.
5. LDBP directly implements the data minimization mandates of GDPR Article 5, CFPB Section 1033, PSD3, and the EU AI Act's principle of least-privilege access — transforming regulatory obligations from legal assertions into architectural guarantees.

LDBP v1.0 is published as a Request for Comment. Implementation feedback, technical contributions, and conformance proposals are welcomed via the GitHub repository.

What Actually Happens When You Link Your Bank Account

When you sign up to pay online, say via Stripe, you may unknowingly be providing financial companies persistent access to your accounts and financial activities — far beyond what any single transaction requires.

During setup, the Finance App redirects you to your bank's login screen — *creating an appearance of informed consent*, since you are authenticating directly with your bank rather than handing credentials to the app. But the authentication flow obscures what access is actually being granted.

But here is what actually happens when you sign into your bank app: depending on the aggregator and the permissions granted, **the Finance App can gain access to your bank accounts and financial transaction history.** If you have more than one account with Bank of America, one Savings and two Checking accounts, the aggregator may be authorized to access all of those, not just the account they'll be drawing money from. This is not hypothetical: in 2022, Plaid — the aggregator powering bank account linking for Venmo, Stripe, and Cash App — settled a $58 million class action lawsuit for collecting and profiting from user financial data, including transaction history from accounts the user had not intended to share, without meaningful consent. *LDBP addresses the structural problem the settlement could not:* changing the architecture so over-collection is technically impossible, not just contractually prohibited.

LDBP addresses the structural problem the settlement could not: changing the architecture so over-collection is technically impossible, not just contractually prohibited.

The Finance App uses this bank authorization to monitor your balance and transactions—e.g. how much you pay your mortgage and who is your lender—information that has nothing to do with you purchasing a t-shirt online. They collect this data from your bank activities to monitor and predict your spending patterns, and in documented cases have shared or monetized user transaction data with third parties. This data is a goldmine of information for Finance Apps—they know what you are spending your money on, your patterns, when you have cash. And there are significant business opportunities for them from this data, including upselling and risk profiling.

This business model leads to a number of serious problems:

- **Privacy Violation:** Users are unaware they are handing over the keys to their entire financial life, including isolated accounts like college savings.
- **No Clear Consent:** Users are not clearly informed when they are asked to sign into their bank account when setting up a payment system.
- **Security Vulnerability:** Finance Apps create massive "data haystacks" that are prime targets for breaches.
- **Economic Loss for Banks:** Banks are essentially giving away their most valuable asset—transaction data—for free to third-party aggregators.

Why is this allowed to happen?

- **Legacy Debt:** Most legacy cores were built for batch processing and "all-or-nothing" access, rather than real-time, per-transaction consent checks.

- **Misaligned Incentives:** Banks lack the granular logic to share only what is necessary, even if they view transaction data as a competitive moat.
- **Lack of Alternatives:** Until now, only full-access APIs or “screen scraping” models existed.

The Governing Principles

A governed finance-banking solution prioritizes the customer’s privacy and consent, with full transparency.

- **Least Data:** Only collect information that is necessary for the transaction. Having access to a person’s every account and corresponding transaction history is too much data.
- **Purpose Limitation:** The data collected must not be used for purposes outside the financial transaction. User data must not be sold or shared with third parties.
- **Transparency:** The user has the right to know what they are providing consent for, presented in language easy to understand.
- **Consent:** The user has the right to grant access only to a specific account, and the right to revoke a Finance App’s access to their data.

The “Least Data” Banking Protocol (LDBP) Solution

The solution is easier than you can imagine: **Use a “Least Data” Banking Protocol.**

LDBP is a bank-implementable open protocol standard—not an app-specific solution. A bank implements LDBP once, and every Finance App connecting to that bank automatically operates under least-data principles. The protocol satisfies regulatory mandates (like CFPB Section 1033) while protecting the bank’s bottom line:

- **Scoped Account Isolation:** Users select one specific account (e.g., Checking) rather than exposing their entire portfolio. The Finance App is given permission to link only to the specific account selected by the user.
- **Boolean Verification:** Instead of returning a raw balance (e.g., $1,402.21), the API returns a simple `True/False` via the `/balance/verify` endpoint (the “is_enough” check). This ensures the Finance App confirms fund availability without disclosing account information. The result is a single Boolean reflecting both balance sufficiency and fraud evaluation. In cases where the return is `False`, see the Fraud Detection section.
- **Intent-Based Validation:** To prevent apps from gaming the system by guessing a balance through repeated checks (Binary Search), every request must be tied to a specific transaction "Intent ID" and subject to strict rate limiting. More sophisticated anti-probing techniques such as noise injection (probabilistic `True` responses near thresholds) or threshold randomization are implementation-defined options, subject to the bank's risk tolerance, regulatory environment, and consumer protection obligations under Regulation E and applicable consumer credit laws.

- **Atomic Execution:** Once `/balance/verify` returns `True`, the Finance App calls `POST /transfer/charge`—the single LDBP execution endpoint. This call atomically re-verifies balance sufficiency, evaluates fraud risk, and deducts funds in one locked PEP operation. There is no separate "execute" step and no gap between verification and deduction. For high-value transactions requiring human authorization between verification and execution, the app calls `/balance/verify` first, completes its approval workflow, then calls `/transfer/charge`—which re-verifies atomically at execution time, ensuring it always acts on current account state.

> *LDBP is designed for Modular Adoption. Banks can immediately mitigate a substantial majority of their data liability by implementing Phase 1 (Boolean Verification) without needing to overhaul their internal ledger for complex recurring-consent logic.*

Conformance: The LDBP Conformance Definition v1.0 defines the authoritative requirements for any implementation claiming LDBP conformance. Any modification that violates the five Least-Data Principles constitutes Principle Drift and may not be represented as LDBP. The Conformance Definition is available in the LDBP GitHub repository.

Standard (Plaid/Yodlee) vs. LDBP

Feature	Standard Model vs. LDBP (MVP)
Account Access	Standard: All accounts (Checking, Savings, CC) LDPB: One specific selected account only
Data Visibility	Standard: Up to 24 months of full history LDBP: Zero history. Verification only
Balance Info	Standard: Exact amount (e.g., $1,402.21) LDBP: Boolean/Threshold (is_enough)
Profiling	Standard: High (Categorizes spending) LDBP: Zero profiling — Boolean only

---

Execution: The Legacy Migration

To implement LDBP, financial institutions do not need a full core replacement. They simply need to implement a Translation and Governance Layer (an API Gateway) that sits on top of their existing ledger.

For the banks, this "Bridge" will handle:

- **ID Strategy:** Generate an "Opaque Alias" for each Finance App so that if an app is hacked, the user's real account number remains safe.

- **Policy Engine:** An internal service that checks “Weekly/Monthly” rules (e.g., “Has Stripe already taken its $200 limit?”) before hitting the ledger.
- **Monetization Option:** Frame the API as a revenue-generating asset where banks have the option to charge basis-point (bps) royalties for every executed transaction.

What Updates Banks Must Do

To make LDBP work, banks must move beyond being simple “data buckets” and become Active Policy Enforcers. The necessary primary shift is technical: banks must implement a Translation and Governance Layer (an API Gateway) that sits on top of their legacy core.

1. Identity & Scoping Layer

Banks currently identify accounts by a 12-digit DDA (Direct Deposit Account) number—a “Permanent ID”. To support LDBP, they must add:

- **Scoped OAuth 2.0 Tokens:** The bank’s login flow must be updated so the user (not the app) selects a specific account, generating a token mathematically restricted to that account ID alone.
- **The Alias Strategy:** The bank must generate a Persistent Account Token (an “Alias”) unique to that specific Finance App. If Stripe is hacked, that Alias is useless for accessing any other bank or app.
- **Intent-ID Tracking:** The bank must support a session-based, single-use Intent-ID generated by the app for every transaction. A `False` response from `/balance/verify` or `/transfer/charge` immediately invalidates the Intent-ID—the Finance App must generate a new Intent-ID for any retry. A consumed Intent-ID (used in a successful `/transfer/charge` call) is permanently invalidated and must not be reused.

2. The Policy Enforcement Point (PEP)

Most banks currently lack a “Boolean Ledger”. They need to build an internal service that handles:

- **Consent Logic Engine:** A service that checks “Weekly/Monthly” rules before the API call even touches the core ledger.
- **Composite Atomic Verification (`POST /balance/verify`):** Instead of a `GET /balance` call that returns a dollar amount, the bank must build a `POST /balance/verify` endpoint (the “is_enough” check) that performs balance verification and fraud evaluation as a single atomic operation within the PEP. These must not be implemented as separate sequential calls. The response is a single Boolean reflecting the combined result of both checks. A `False` response is deliberately opaque—all failure reasons look identical to the Finance App.
- **Request Throttling:** To prevent “Binary Search” attacks, the bank must implement strict rate limiting (e.g., 5 calls per hour) on verification endpoints.

3. Core Ledger Adjustments

- **Atomic Transactions:** The bank must implement an "Atomic Wrapper" (a stored procedure) that performs Balance Verification + Fraud Evaluation + Fund Deduction in a single, uninterrupted step within the PEP. These three operations MUST NOT be separated into sequential calls. This prevents race conditions and ensures the execution endpoint (`POST /transfer/charge`) always acts on current account state.
- **Mandatory Idempotency Caching:** The API Gateway must cache `x-idempotency-key` values for 24 hours. If a network failure causes a Finance App to retry a successful transaction, the bank must recognize the key and ensure the user is not charged twice.

4. Consumer Protection & Webhooks

To comply with Regulation E while maintaining privacy, the bank needs:

- **The "Kill Switch" Portal:** A user-facing dashboard where a single click invalidates a `scoped_account_token`, breaking the app's access instantly. The portal must include a transaction log with human-readable status codes visible to the authenticated user only. Required status taxonomy: `approved, declined_insufficient_funds, declined_fraud_review, declined_rate_limit, declined_revoked, declined_cap_exceeded`. These status codes are strictly internal to the portal and must never be surfaced in API responses.
- **Real-time Consent Receipts:** An automated notification system that pushes a message to the user after every successful `/transfer/charge` execution, detailing the amount and the specific "Consent Policy" used.
- **Fraud Block Webhook (verification.blocked):** When the bank's internal fraud logic blocks a verification request, the bank must simultaneously trigger a webhook notification to the authenticated user. Payload must include: timestamp, app alias, intent-ID, and a human-readable reason for display in the Kill Switch portal. This notification path goes "bank → user directly", and never "bank → app → user." The API response to the app remains strictly `False` with no reason code.

> *Implementing banks are responsible for all consumer notification workflows triggered by internal flag events. Reason codes are strictly internal and must not be surfaced in API responses to third parties.*

MVP & Phases of Implementation

In order to make LDBP readily adoptable despite legacy systems, it is modular and can be applied in two phases.

Phase 1: A "Clean Pipe" (Low/Medium Effort)

- **Goal:** Replace raw data scraping with Boolean verification.
- **Mechanism:** The bank implements the `/balance/verify` endpoint (the “is_enough” check)—a composite atomic verification that checks balance sufficiency and evaluates fraud risk, returning a single `True/False`.
- **The Win:** This immediately solves the “Data Haystack” liability for the bank and the “Privacy Erosion” for the user.

By focusing on Just-in-Time (ATM-style) verification first, the barrier to adoption of safe banking is lowered while still solving the core privacy problem.

To be LDBP Phase 1-compliant, the bank only needs to implement four things:

1. **Opaque Scoped Tokens:** Stop sending the 12-digit DDA; send a one-time or app-specific Alias ID that is unique to each Finance App and mathematically restricted to the one account the user selected.
2. **The Boolean Wrapper:** A composite logic gate on top of the existing GET /balance that atomically evaluates balance sufficiency and fraud risk, returning a single `True` or `False`. No raw balance, no reason codes, and no transaction history leave the bank.
3. **Atomic Verify+Execute (`POST /transfer/charge`):** The single execution endpoint. When a Finance App calls `/transfer/charge`, the bank atomically re-verifies balance sufficiency, evaluates fraud risk, and deducts funds in one locked PEP operation. There is no separate execute step and no gap between check and debit.
4. **Basic Security & Revocation:** A hard rate limit (e.g., 5 calls/hour) on `/balance/verify` to prevent balance-guessing attacks, and a Kill Switch in the bank's Connected Apps portal allowing the user to instantly revoke a Finance App's access with a single action.

Phase 2: Risk Capping & Governance (Optional/High Effort)

- **Goal:** Introduce value-bound and time-bound “Virtual Allowances”.
- **Mechanism:** This is where the bank builds the Policy Enforcement Point (PEP) to track cumulative weekly spending.
- **Benefit:** This acts as a “Premium Security Tier” or enables banks to offer “Parental Controls” and “Subscription Management” as value-add services.
- **Risk Capping** (weekly/monthly limits), though ideal to implement early, is an optional “Premium” or Phase 2 feature.

Phase 1 Execution

By stripping the protocol down to Phase 1, the Level of Effort (LOE) for a bank moves from a multi-year digital transformation to a standard 6-to-9 month API roadmap item. Once a bank is LDBP-compliant, any Finance App connecting to it gains least-data protection automatically — with no additional protocol implementation work required from the app.

1. Identity & Scoped Tokens (LOE: Low to Medium)

- **Account Selector UI:** Adding a radio button group to the bank's existing OAuth consent screen — the screen presented after the user authenticates, where permissions are displayed — so that the user selects exactly one account before the token is issued. This is a consent screen modification, not a login screen modification, and does not touch authentication infrastructure.
- **Alias ID Generation:** Creating a simple mapping table that generates an Opaque Token (e.g., alias_8822_stripe) instead of passing the raw 12-digit DDA number. The Alias ID is unique per Finance App — a compromised Alias ID for one app provides no access through any other.

2. The Boolean `/balance/verify` Endpoint (LOE: Medium)

2. **The Composite Logic Gate:** A simple API wrapper that takes an Amount from the app, runs the composite check (balance sufficiency + fraud evaluation) atomically within the PEP, and returns a single `True` or `False` Boolean.
- **Privacy Masking:** The response body is strictly limited to the Boolean, the echoed Intent-ID, and a timestamp — no raw balance, no reason codes, no transaction history.

3. Atomic "Check-then-Capture" — `POST /transfer/charge` (LOE: Medium to High)

- **Atomic Wrapper:** A stored procedure that verifies the balance, evaluates fraud risk, and earmarks or deducts funds in a single database lock. These three operations must not be separated into sequential steps.
- **Idempotency Handling:** A 24-hour cache for the X-Idempotency-Key to prevent double-charging during network retries.

4. Basic Security & Revocation (LOE: Low)

- **The Kill Switch:** A "Revoke Access" button on the bank's existing Connected Apps dashboard. A single user action must immediately invalidate the Scoped Account Token and terminate the Finance App's access — no grace period, no Finance App notification required.
- **Request Throttling:** A hard rate limit (e.g., 5 calls/hour) on the `/balance/verify` endpoint to prevent binary search attacks on the user's account balance.

Phase 1 LOE Summary

Component	Level of Effort / Technical Requirement
Scoped Tokens	**Low** — OAuth 2.0 account-level scoping
Boolean `/balance/verify`	**Medium** — Composite atomic wrapper: balance verification + fraud evaluation returning single Boolean
Atomic Execution (`POST /transfer/charge`)	**Medium/High** — Single-step Verify + Fraud Check + Deduct logic in one locked PEP operation

Kill Switch & Rate Limiting	**Low** — Connected Apps portal revoke button + hard rate limit (5 calls/hour) on `/balance/verify`

Effect on Existing Finance Apps

Here is how LDBP shifts the business landscape. Because LDBP is implemented at the bank layer, Finance Apps do not implement the protocol—they integrate with it. The disruption is to the data they previously received, not to their integration architecture:

1. The Loss of the "Financial Profile"

- **Today's Reality:** By harvesting 24 months of full transaction history, apps build a detailed financial profile. They know your salary, your rent, your gambling habits, and even when you're likely to need a loan.
- **LDBP Impact:** Since the app only sees a Boolean `True/False` and zero history, their ability to profile you for targeted ads or cross-selling financial products evaporates.

2. Death of "Data Brokerage"

- **Today's Reality:** Many aggregators treat user data as a product, selling anonymized "consumer spending trends" to hedge funds or marketing firms.
- **LDBP Impact:** By blinding the data miners, if they can't see merchant names (Merchant Metadata Masking) or transaction amounts, they have no product to sell to third parties.

3. The "Incentive Conflict"

- **The Resistance:** Apps will argue that they need this data for "Fraud Prevention" or "Identity Verification." See the *Fraud Detection* section for the full rebuttal.
- **Security > Mining:** By using Scoped Tokens and Atomic Transfers, the app actually reduces its own liability. If they don't hold the data, they can't lose it in a breach.

4. Profitability Impact

LDBP destroys illegitimate revenue streams while leaving legitimate ones intact.

Their core transaction revenue is untouched (e.g. interchange fees, payment processing margins, subscription fees).

However, their data brokerage and behavioral profiling businesses do not survive. But those revenue streams were built on data extracted without meaningful consent anyway, and repurposed beyond the transaction that justified its collection. This practice is already legally vulnerable — GDPR's Purpose Limitation principle prohibits using data for purposes beyond

those originally stated, and the CFPB's Section 1033 Final Rule explicitly bars secondary use of consumer financial data for advertising or data sales.

The one legitimate hard case is Buy Now, Pay Later (BNPL). Apps like Klarna and Affirm have a genuine underwriting dependency: their core product (instant micro-lending) requires rapid financial assessment that goes beyond a single-transaction Boolean. LDBP addresses this through a bank-mediated creditworthiness signal—a scoped proof such as is_debt_ratio_acceptable—that returns a verified result without transferring the underlying data. This shifts the underwriting intelligence back to the bank, where the asset and the liability both reside.

5. “Win” Shifts to the Bank

- **Revenue Recovery:** Currently, banks give transaction data to apps for free. Under LDBP, the bank takes back control.
- **Monetization:** Instead of the app making money by selling user data, the bank may choose to make money by charging the app a small fee (basis points) for the Safe Verification service.

Finance Apps Also Benefit

A standardized Boolean API eliminates the need to maintain separate aggregator integrations per institution

Finance Apps that do not hold raw financial data carry dramatically reduced breach liability and GDPR/1033 compliance cost.

Summary of the “Business Collision”

Feature	Finance Apps’ Current Approach	LDBP “Privacy by Design”
Visibility	User transaction activities	Zero visibility
Secondary Revenue	Insights from user transaction data / history	None (user data doesn’t exist)
User Relationship	“Sticky” via profiling	Transactional via trust

---

Fraud Detection

Fraud teams rely heavily on transaction history patterns—velocity checks, merchant category anomalies—to catch account takeover. Finance apps will argue that they need this data for "Fraud Prevention" or "Identity Verification." With LDBP, this responsibility returns to the bank. The bank retains fraud logic internally and returns `False` not just when funds are insufficient, but also when a transaction is deemed fraudulent.

Today's Fraud Responsibility Overlap

The current all-access model has created a diffuse and circular fraud responsibility structure:

- Banks run their own fraud models on transaction patterns.
- Aggregators like Plaid run identity/account verification fraud checks.
- Payment apps (Stripe, Venmo) run their own behavioral fraud models using the harvested bank data.

The problem is that Finance Apps have justified harvesting a user's full transaction history specifically because they inserted themselves into the fraud detection layer. This has led to a circular dependency — Finance Apps built their fraud models on data collected beyond what any individual transaction required, then cited those models as justification for continued over-collection.

Fraud Belongs to the Layer That Owns the Asset

Since the bank owns the account and the funds, account-level fraud rightly belongs to the bank. The Finance App's fraud responsibility should be scoped only to its own transaction—not the broader account.

This is exactly how credit cards work:

- Visa/Mastercard and the issuing bank own fraud detection on the account.
- The merchant has zero visibility into the cardholder's other transactions.
- Yet fraud is effectively managed.

LDBP restores this clean separation. The bank's internal Policy Enforcement Point is where account-level fraud logic should live. The Finance App gets a Boolean, and if the bank's internal fraud model flags the request, it simply returns `False`—with no data leakage about why.

> The bank's internal Policy Enforcement Point is where account-level fraud logic should live. The Finance App gets a Boolean — and if the bank's fraud model flags the request, it simply returns False, with no data leakage about why.

What the Bank's PEP Knows (And Keeps Internal)

The bank's Policy Enforcement Point (PEP) already knows why it is returning `False`. It could be any of the following:

- Insufficient funds
- Rate limit exceeded
- Fraud flag
- Revoked token
- Weekly cap hit

None of these reasons belong in the response to the Finance App. They all look the same from the outside: `False`. The reason is an internal bank state, and surfacing it violates the least-data principle LDBP is founded on.

What Should Actually Happen on a Fraud Flag

The notification path goes “bank → user directly”, and not “bank → app → user.”

Specifically:

- The bank’s internal fraud logic returns `False` to the app, no reason is given.
- The bank simultaneously triggers a `verification.blocked` webhook to the user with timestamp, app alias, intent-ID, and human-readable reason.
- The user sees this in their Kill Switch portal.

This is stronger consumer protection than a flagged API response because:

- The app never knows it was flagged, it just sees `False`.
- The user is informed directly by the trusted party (the bank).
- A malicious app cannot use the flag signal to probe or adapt its behavior.

Exception: User Portal Only

There is a narrow case where a flagged state has legitimate value: the user’s own dashboard experience, not the API response.

If a user checks their Kill Switch portal and sees “Stripe verification — Blocked (fraud review),” that is appropriate. The user deserves to know why their transaction did not go through. But that information lives in the bank’s consumer-facing portal, surfaced to the authenticated user only—never in the API response to the third party.

The Transition Risk: Fraud Coverage Gap

Apps will make the case that fraud monitoring degrades under LDBP because they currently run sophisticated cross-merchant behavioral fraud models. This argument has partial merit and must be addressed directly.

Payment apps conflate two distinct fraud signals: **account-level behavioral data**, which belongs to the bank, and **transaction-level network data**, which the app legitimately generates through its own payment activity. LDBP restricts the former while leaving the latter entirely intact. An app that can only detect fraud using data it was never entitled to hold has built a dependency it should not have.

A second concern is that **banks have been gradually offloading fraud detection responsibility to aggregators and payment processors** for years. Their internal fraud models, particularly at mid-size and regional institutions, have atrophied. Asking banks to reclaim that responsibility is asking them to invest in a capability they have been comfortable not building. This is a real adoption barrier, and it is compounded by core processor dependency: FIS, Fiserv, and Jack Henry—who run the back-end ledgers for most US banks—have no commercial incentive to build Boolean verification or enhanced fraud logic into their platforms.

LDBP addresses this through three mitigations:

- **Federated Fraud Signals:** Banks share anonymized, aggregated fraud signals through a network such as Early Warning Services (which already runs Zelle's fraud backbone) without sharing individual account data. The fraud intelligence is collective; the account data stays private.
- **The Regulatory Lever:** CFPB 1033 and PSD3 create compliance pressure that gives banks the external mandate they need to justify infrastructure investment to their boards. LDBP adoption converts regulatory obligation into competitive differentiation.
- **Phased Responsibility Transfer:** Phase 1 LDBP does not require banks to have fully upgraded fraud detection on day one. The transition period—where both the app's behavioral model and the bank's Boolean layer coexist—provides a bridge while bank-side capability matures. This is a deliberate design feature of the phased model.

> *Fraud detection architecture, federated signal standards, and cross-institution interoperability are designated as active development areas for LDBP v1.0. Implementers are invited to contribute proposals through the GitHub repository.*

How LDBP Differs From Aggregator-Layer Approaches

Plaid's Signal product represents a step in the right direction—applying machine learning to predict ACH return risk and returning recommended actions. But Signal operates at the aggregator layer: Plaid still receives all account data from the bank first, then applies intelligence on top of it. The raw balance still leaves the bank. The transaction history is still accessed. Signal makes the all-access model more intelligent; LDBP makes it architecturally unnecessary.

Plaid's Signal makes the all-access model more intelligent. LDBP makes it architecturally unnecessary.

The distinction is not one of degree, it is one of where the intelligence lives. LDBP is a bank-side protocol standard. The verification logic lives at the bank, where the asset and the liability both reside. Any Finance App—whether it previously used Plaid, Yodlee, or any other aggregator—connects to the same LDBP Boolean interface once the bank implements it. No app-specific integration work is required beyond adopting the standard API contract.

LDBP's Incentive for Banks

Currently, banks give transaction data away for free to finance and Finance Apps. LDBP lets banks monetize this via basis-point royalties on every executed transaction, with the option to meter verification calls at institution discretion.

Banks have historically been complicit in the all-access model because their tech vendors—core processors like FIS, Fiserv, and Jack Henry—do not support granular scoping natively. This is a real adoption barrier beyond legacy debt: banks lack both the internal policy infrastructure and the commercial motivation to invest in it.

With LDBP, the bank's ROI case becomes concrete. Instead of giving away the asset, the bank becomes the active policy enforcer and charges for the verification service. The compliance cost reduction is equally significant: under CFPB 1033 and GDPR, holding user financial data creates ongoing legal surface area. If the bank shares only a Boolean result rather than raw account data, a breach at a third-party fintech exposes nothing—reducing the bank's downstream liability substantially.

> A breach at a third-party fintech exposes nothing — because the fintech never received anything worth stealing.

The Interoperability of LDBP and CFPB Section 1033

Policy Meets Protocol

While CFPB Section 1033 codifies the consumer's right to data portability, it leaves a technical vacuum regarding how that data is minimized and protected during transit and ingestion. LDBP fills this gap by transforming the legal mandate of "data minimization" into an automated technical reality.

1. From "Reasonably Necessary" to "Technically Impossible"

- **Section 1033 Limitation:** Legal certifications do not prevent a malicious or negligent actor from over-collecting data via the bank's API.
- **LDBP Advantage:** LDBP introduces Attribute-Level Scoping. Instead of a third party requesting a broad "Transaction History" scope, the protocol allows for binary or specific proof-based responses (e.g., "Does the user have >$500?" or "Has a direct deposit occurred in 30 days?") without moving the underlying PII or full ledger.

2. Solving the "Secondary Use" Enforcement Crisis

- **Section 1033 Limitation:** Once a third party ingests data, the CFPB has limited visibility into how that data is used within the company's internal silos.
- **LDBP Advantage:** LDBP utilizes Ephemeral Data Objects and Zero-Knowledge Proofs (ZKPs). By ensuring the third party never "possesses" the raw sensitive data—only the

verification of the data—the risk of secondary use is eliminated by design. If you don't have the data, you can't sell it.

> *Zero-Knowledge Proof (ZKP) implementation is designated as an advanced or Phase 2 capability. Phase 1 compliance does not require ZKP infrastructure. Banks may achieve equivalent data minimization outcomes through internal comparison logic and secure enclave processing, with ZKPs representing the cryptographically strongest instantiation of this principle.*

3. Reducing "Compliance Friction" for Banks

- **LDBP's Value-Add:** By adopting LDBP as their 1033 interface standard, banks significantly reduce their downstream liability. If a bank shares only a Boolean result rather than raw account data, a breach at a third-party fintech exposes nothing — because the fintech never received anything worth stealing.

Note: As of April 2026, CFPB Section 1033 is subject to ongoing litigation and CFPB reconsideration; LDBP's technical approach remains valid regardless of the rule's litigation status.

Implementation Matrix: LDBP vs. CFPB Section 1033 Requirements

1033 Regulatory Requirement	LDBP Technical Implementation
Data Minimization	**Proof-of-Attribute:** Replaces raw data sets with verified results
Revocation Rights	**Cryptographic Token Expiry:** Hard-coded access windows that self-terminate
No Secondary Use	**Secure Enclaves / ZKPs (Phase 2):** Data processed in a clean room and never stored
API Standardization	**OpenAPI/LDBP Specs:** A structured, repeatable schema for all fintechs

---

Agentic Governance: LDBP as the Financial Primitive for Autonomous Systems

The emergence of Agentic and Multi-Agent Systems (MAS) introduces a governance crisis that existing open banking frameworks were never designed to handle.

When a human links their bank account to Stripe, there is at least a moment of conscious consent—a login screen, a button click. When an AI agent executes a financial operation autonomously, that moment disappears. The agent acts on behalf of the user, often chaining

multiple decisions across multiple systems, with no human in the loop to catch over-collection, scope creep, or cascading authorization errors.

Standard APIs like FDX were designed for human-initiated, session-bound interactions. Giving an autonomous agent an all-access token under these frameworks is the equivalent of handing a contractor a master key to every room in your house because they need access to the kitchen—the blast radius of a compromised or misbehaving agent is the user's entire financial life, not just the transaction the agent was dispatched to complete.

LDBP's scoped proof model is architecturally matched to this problem in a way that standard open banking APIs like FDX are not. An agent executing a $50 grocery reorder does not need the user's account balance—it just needs a Boolean confirmation that the funds exist for that specific transaction, tied to an Intent-ID that expires when the task completes. This is the right primitive for agentic financial operations: least-privilege by design, not by policy.

In a Multi-Agent System where an orchestrator delegates to sub-agents across multiple financial tasks, each agent receives only the verification surface it needs for its specific function—nothing persists, nothing accumulates, and a compromised sub-agent cannot laterally access account data beyond its scoped proof. Each agent in the delegation chain represents a potential over-collection surface; LDBP's architecture ensures that surface is bounded by the task, not the account. LDBP does not just protect users from fintech data harvesting—it provides the governance layer that makes autonomous financial agency safe to deploy at scale.

> LDBP does not just protect users from fintech data harvesting — it provides the governance layer that makes autonomous financial agency safe to deploy at scale.

For high-volume agentic use cases—subscription processing, recurring billing, multi-task agent queues—LDBP provides two batch-scale endpoints. `POST /batch/balance/verify` is an informational call that returns per-intent Boolean results for planning purposes but does not move or earmark funds. `POST /batch/transfer/charge` is the exclusive batch execution path: each intent is atomically verified and executed independently within the PEP, with no verify-execute gap. Partial success is valid—each intent is processed independently. This makes `/batch/transfer/charge` the natural primitive for agentic systems processing a queue of pre-authorized payments, where each sub-agent submits its intent and the bank settles atomically without any agent accumulating account state.

How Does LDBP Stack Up Against EU Regulations as of 2026?

This "Least Data" vision is actually closer to the legal reality in the EU than it is in the US. While many Finance Apps still try to harvest as much data as possible, the EU is aggressively moving toward a framework that mandates Data Minimization and User Control.

1. Screen Scraping: The “Illegal” Legacy

In the EU, screen scraping is being banned under PSD3 (Payment Services Directive 3) and the PSR (Payment Services Regulation).

- **The Rule:** Banks are no longer allowed to provide “fallback” interfaces that permit scraping.
- **The Shift:** They must provide dedicated, secure APIs that follow strict data access protocols.

2. GDPR and “Purpose Limitation”

- **The Conflict:** If an app needs to check a balance for a $50 transaction but harvests 24 months of history instead, it is technically violating GDPR because it is collecting excessive data not necessary for the specific purpose.
- **The Reality:** Despite the law, enforcement has been a cat-and-mouse game. However, EU regulators (like the Dutch Data Protection Authority) have started ruling that scraping personal data for commercial interests is often a GDPR infringement - in the context of screen scraping and unauthorized credential access.

3. FiDA: The “Open Finance” Expansion

FiDA (Financial Data Access) Regulation, currently in trilogue negotiations and expected to be implemented in the coming years, expands these rules beyond checking accounts to include insurance, investments, and pensions.

- **Consent Dashboards:** Banks are now required to provide transparent dashboards where users can see exactly who has access to their data and revoke it with one click—this is the exact Kill Switch LDBP specifies.
- **Standardized Sharing:** FiDA forces data holders to share data only under standardized technical schemes, preventing data haystacks from being created by third-party apps.

4. Where LDBP Goes Further Than Current EU Standards

- **Most EU APIs still return the raw balance amount to the app. LDBP goes one step further: the app only sees a `True` or `False`.** This aligns perfectly with the Privacy-by-Design and Data Minimization goals that EU regulators are currently championing but have not yet technically enforced at the code level.

The “All-Access” Finance App Landscape

Cash App and the P2P Data Model

Cash App — one of the most widely used peer-to-peer payment platforms in the US — illustrates how the all-access model creates both a privacy exposure and a security risk for users. Its documented data practices, confirmed by its own published privacy policy and federal regulatory action, make it one of the clearest examples of why architectural reform is necessary.

1. The Onboarding "Handshake"

- **The Aggregator:** When a user links their bank account to Cash App, the connection is facilitated through Plaid — a relationship confirmed by the $58 million Plaid class action settlement, which named Cash App as one of the apps for which Plaid provided bank account authentication services.[1]
- **The Data Collection:** Cash App's own Privacy Notice states it collects bank account and payment card numbers, transaction information including merchant names, amounts, and dates, and uses this data for purposes including marketing and behavioral profiling.[7] Starting in February 2026, Cash App's Privacy Notice explicitly states it may use data about shopping history, app browsing behavior, and card transactions to show users personalized advertisements for other brands outside of Cash App.[7]
- **Persistent Access:** Once a bank account is linked, Cash App maintains an ongoing connection to initiate transfers. Unlike a one-time card authorization, this access persists until the user explicitly disconnects it — a step most users never take.

2. The Security Consequence

The all-access model does not just create a privacy risk — it creates a documented fraud and security liability. In January 2025, the CFPB ordered Block (Cash App's parent company) to pay $175 million — $120 million in consumer refunds and a $55 million penalty — for failing to protect users from fraud and unauthorized transactions.[8] The CFPB stated: *"Cash App created the conditions for fraud to proliferate on its popular payment platform."*[8] Separately, 48 state regulators fined Block an additional $80 million for Bank Secrecy Act violations.[8]

This is the consequence the all-access model enables: a persistent, broad connection to a user's bank account, combined with inadequate fraud investigation, means that when something goes wrong, users have limited recourse and the bank bears costs it did not create.

3. How LDBP Would Change the Cash App Experience

Feature	Cash App Today	Cash App + LDBP
Initial Link	Bank account + credentials via aggregator	Scoped Alias ID only — no credentials shared
Data Collected	Account numbers, transaction history, behavioral patterns	Zero history. Boolean verification only
Advertising Use	Transaction data used for targeted ads (from Feb 2026)[7]	No transaction data exists to use

The Transfer	ACH pull with persistent bank access	Atomic verify+execute via `/transfer/charge`
Fraud Surface	Broad persistent connection exploitable by bad actors	Time-bound scoped token — blast radius limited to authorized amount

4. Why Apps Argue They Need This Access (And Why They Don't)

Finance Apps in this category typically argue they need broad account access for fraud prevention and identity verification. The CFPB's enforcement action against Cash App reveals the circularity of this argument: Cash App had broad access to user financial data and still produced fraud investigations the CFPB described as "woefully incomplete."[8] Broad data access did not produce effective fraud protection — it produced a liability.

Idempotency Keys and Intent-IDs solve the legitimate fraud prevention need technically. By linking each specific transfer to a unique ID, the bank can prove the user authorized that specific transaction, removing the need for the Finance App to maintain persistent account monitoring.

Other Finance Apps

Many of the most popular apps in the 2026 digital economy operate on the same all-access model, using third-party aggregators to create a persistent connection to the user's bank.

1. Peer-to-Peer (P2P) Payment Apps

Apps like Venmo and PayPal are the most significant data haystacks because they sit between the bank and the user's social life.

- **The Access:** These apps often require a permanent link to the bank to verify identity and ensure instant transfers.
- **The Pattern:** Through the aggregator, these apps can access substantial transaction history and account data to build a risk profile. This practice was the subject of a $58 million settled lawsuit against Plaid, which powers account linking for Venmo, Cash App, and other P2P apps.

2. "Buy Now, Pay Later" (BNPL) Services

Apps like Klarna, Afterpay, and Affirm appear to be simple payment methods but are deep-data divers—with a legitimate underwriting dependency (see Profitability Impact section).

- **The Access:** To approve a micro-loan in seconds, they often access bank account data to calculate debt-to-income ratio and liquidity.[4]
- **The Pattern:** They analyze spending patterns—how quickly the paycheck is spent—to determine how much credit to extend.

3. Investment & "Round-Up" Apps

Apps like Acorns and Stash position themselves as helpful savings tools.

- **The Access:** To round up change, these apps need access to transactions from your linked bank accounts to identify spending and calculate round-up amounts.
- **The Pattern:** This creates a broad view of spending patterns from linked accounts, which can inform product recommendations and upselling opportunities.

4. Gig Economy & Delivery Apps

Uber, Lyft, and DoorDash have increasingly moved into the Wallet space (e.g., Uber Money).

- **The Access:** They encourage bank linking for one-click payments or driver payouts
- **The Pattern:** Through bank linking, they create a persistent data connection that, under the current all-access model, provides visibility beyond what any individual payment transaction requires.

Summary of the "All-Access" Apps Landscape

App Category	Their "Hook" & Their Hidden Access
P2P (Cash App)	*"Send money to friends instantly"* Bank account linked via Plaid; transaction data used for behavioral advertising from Feb 2026; $175M CFPB order for fraud failures[8]
P2P (Venmo/PayPal)	*"Split the bill with friends"* Access to transaction history and linked account data via aggregator (subject of $58M Plaid settlement)
BNPL (Klarna)	*"Pay in 4 easy installments"* Bank account data accessed for underwriting; legitimate dependency but scope exceeds transaction need
Savings (Acorns)	*"Save your spare change"* Access to transactions from your linked bank accounts
Gig Economy (Uber)	*"One-click payments"* Persistent data connection through bank linking

---

Considerations & Concerns

Here is a breakdown of why LDBP is a necessary evolution, even if the problems feel solved today:

1. Are Race Conditions Already Solved?

- **Today's "Solution":** Banks use memo-posting to temporarily lower the Available Balance before final settlement. However, in the ACH world (which Stripe often uses), there is still a 1-2 day gap where a user can initiate a transfer and then drain their account at an ATM before the ACH clears, leading to an NSF return.
- **LDBP's Difference:** The Atomic Wrapper forces the balance check, fraud evaluation, and debit to happen in a single locked database operation—making the transaction Real-Time and irrevocable, similar to FedNow or RTP networks.

2. Can't It Work Like an ATM? (Just-in-Time)

- **Today's "Solution":** When Stripe wants to check a balance, it uses an API (like Plaid) to scrape the full balance (e.g., $1,402.21), stores it, and makes a decision.
- **LDBP's Difference**: The "Brain" is moved back to the bank. The is_enough(cost) API is exactly like an ATM's Just-in-Time check, but the ATM doesn't tell the merchant how much extra money the user has left. It only says `True` to the $20 asked for.

3. Do We Need a "Weekly" Check? How Is This Helpful?

- **Today's "Solution":** Once a user links their account to Stripe or Venmo, the aggregator can maintain a persistent connection that remains active until the user explicitly revokes it, a step most users never take. If the aggregator's servers are hacked, user financial data is at risk.
- **LDBP's Difference**: A weekly/monthly Consent Policy is like a virtual allowance. It limits the blast radius of a hack. Even if an attacker steals the token, they can only take the $50/week authorized, not the user's life savings.

4. Are Banks Already Going to a Specific Account?

- **Today's "Solution":** A user might select their checking account in a Stripe pop-up, but the underlying token often still grants Stripe visibility into the entire profile (Savings, CCs, etc.) to verify identity or evaluate for loans.
- **LDBP's Difference**: The Scoped Alias ID ensures that the bank's server literally rejects any request that is not for the specific checking account the user picked. It is an architectural Hard Wall, not a Pinky Promise policy.

Comparison: Today vs. LDBP

Feature	Today's Workaround	LDBP Design
Race Conditions	Fixed by NSF Fees and 2-day delays	Fixed by Atomic API Handshakes
Balance Checks	Apps scrape exact balance	Apps get a Boolean `True/False`

Safety	Persistent aggregator connections	Time-Bound/Value-Bound caps
Account Isolation	Policy-based (App shouldn't look)	Token-based (App can't look)

The Credit Card Precedence

Credit cards already prove that Boolean-style verification works at scale. Using a credit card is currently a much safer strategy than linking a bank account through a standard aggregator. While both involve data sharing, the blast radius and depth of data harvested are significantly different.

1. Data Harvesting Depth

- **Bank Account (The "All-Access" Pipe):** When linking a bank account via an aggregator, the app often scrapes full transaction history across all accounts to build a financial profile.
- **Credit Card:** The merchant generally only sees the specific transaction being made. They do not get a pipe into savings accounts or mortgage balances.

2. Liability and Protection

- **Bank Account (ACH/DDA):** If an app's All-Access token is stolen, an attacker could potentially drain real cash. While Regulation E provides some protection, recovering physical cash from a checking account is often a long, stressful process.
- **Credit Card:** You are spending the bank's money, not yours. If a card is compromised, you have zero liability for unauthorized charges.

3. The "Profiling" Gap

- **Bank Account:** The Finance app can use bank data to identify when income arrives and how quickly it is spent.
- **Credit Card:** The Credit Card app only knows that the card was Approved or Declined for a specific amount. It cannot see the total credit limit or other spending habits at competing stores.

LDBP is currently the only protocol standard that allows users to use their actual cash with credit-card-equivalent privacy and security through architectural enforcement rather than contractual policy.

LDBP as the Third Way

LDBP is designed to make a bank account behave with the safety of a credit card.

Feature	Bank Account (Today)	Credit Card	LDBP Approach
Risk	High (Real Cash at Risk)	Low (Bank's Money)	Low (Value-Bound Caps)
Visibility	Full History	Single Transaction	Boolean Only
Control	Permanent Pipe	Per-charge	Instant Kill Switch

Using a credit card provides a Privacy Wall that standard bank linking currently lacks—there is already precedent for the least-data approach. LDBP is currently the only protocol standard that allows users to use their actual cash (DDA) with credit-card-equivalent privacy and security through architectural enforcement rather than contractual policy.

Implementation-Defined Behaviors

LDBP deliberately leaves the following behaviors to the implementing institution. This is a design choice, not a gap. It enables adoption across institutions with different risk models, infrastructure, and regulatory environments.

- **Fraud Scoring Model and Internal Flag Thresholds:** The specific algorithms, data sources, and threshold values used by the bank's fraud logic are institution-defined. LDBP requires only that fraud evaluation is performed atomically within the `POST /balance/verify` and `POST /transfer/charge` operations and that its result is reflected in the Boolean response.
- C**onsumer Notification Format and Delivery Channel:** The `verification.blocked` webhook payload format and delivery mechanism (push notification, SMS, email) are institution-defined. LDBP requires only that notification is delivered directly to the authenticated user and not surfaced in the API response.
- **Attribute Verification Method:** Phase 1 adopters may implement Boolean verification via internal ledger comparison logic. Zero-Knowledge Proofs (ZKPs) are the cryptographically rigorous instantiation of this principle and are recommended for Phase 2 or for institutions handling high-value or regulated data environments. LDBP does not mandate a specific cryptographic method—only that raw attribute data is never surfaced in the API response.
- **Weekly/Monthly Cap Values (Phase 2):** The specific value-bound and time-bound limits for Virtual Allowances are user-configured and institution-implemented. LDBP specifies the mechanism, not the limits.

Future Work / Roadmap

The following areas are out of scope for the current release and designated for future LDBP versions. Implementers and researchers are invited to contribute proposals through the GitHub repository.

- **Fraud Detection Architecture:** Standardized interfaces for bank-side fraud signal exchange, including federated fraud signal protocols compatible with Early Warning Services and similar networks.
- **Cryptographic Threshold Hardening:** Formal specification of ZKP integration patterns for attribute verification, including proof schemes suitable for Phase 1 performance constraints.
- **Cross-Institution Interoperability:** Standards for Alias ID portability and Intent-ID coordination across multiple banks and payment networks.
- **BNPL Creditworthiness Signal:** Formal specification of bank-mediated scoped proof patterns (e.g., is_debt_ratio_acceptable) that support underwriting use cases without transferring underlying transaction data.
- **Agentic Authorization Standards:** Formal specification of Intent-ID lifecycle management for multi-agent delegation chains, including sub-agent scope inheritance and revocation propagation.

Appendix A: Defined Terms

User The individual who holds the bank account. The User is the person whose financial data is at stake, whose consent is required, and whose rights are protected by LDBP. The User is distinct from the Finance App and from the bank. In consumer contexts, the User is also the account holder, the data subject (under GDPR), and the consumer (under CFPB 1033). LDBP is designed primarily to protect the User's financial privacy and control.

Finance App / Payment App / App These terms are used interchangeably throughout this document to refer to any third-party application, platform, or service that requests access to a User's bank account data or initiates fund transfers on the User's behalf. This includes but is not limited to: payment processors (e.g. Stripe), peer-to-peer payment platforms (e.g. Venmo, Cash App), Buy Now Pay Later providers (e.g. Klarna, Affirm), investment and savings apps (e.g. Acorns, Stash), and gig economy wallet services (e.g. Uber Money). The Finance App is the party that connects to the bank via the LDBP API. It is not the implementer of the LDBP protocol — it is a consumer of it.

Bank The financial institution that holds the User's account, owns the account data, and is the implementing party of the LDBP protocol. Under LDBP, the bank is the Active Policy Enforcer — it performs verification, evaluates fraud risk, moves funds, and notifies the User. All verification logic lives at the bank. The bank never exposes raw financial data to the Finance App.

Aggregator A third-party data intermediary — such as Plaid or Yodlee — that sits between the bank and the Finance App, facilitating bank account linking by collecting the User's bank credentials or using open banking APIs to extract account data. Under the current all-access model, aggregators collect broad financial data from the bank and share it with Finance Apps. LDBP replaces the aggregator's data extraction role with Boolean verification performed directly at the bank's Policy Enforcement Point.

All-Access Model The current dominant paradigm in open banking, in which a Finance App or aggregator is granted broad, persistent access to a User's bank accounts — including all accounts, full transaction history, and exact balances — in order to perform any financial operation, regardless of how minimal the actual data requirement is for that operation.

Least Data The principle that only the minimum data necessary to complete a specific transaction should be collected, accessed, or transmitted. Under LDBP, the minimum necessary data for a payment transaction is a single Boolean: does the account hold sufficient funds? No raw balance, no transaction history, and no account metadata is necessary or permitted beyond this.

Boolean Verification / The "is_enough" Check The core LDBP verification mechanism. Rather than returning a raw numerical balance (e.g. $1,402.21), the bank's internal system evaluates whether a specified amount is available and returns a single `True` or `False` result. The Finance App receives only this Boolean — it never learns the actual balance. The term "is_enough" is used as the conceptual label for this check throughout the document. The corresponding API endpoint is `POST /balance/verify`.

DDA (Direct Deposit Account) The standard 12-digit account number used by banks to identify a User's checking or savings account internally. Under the all-access model, this number is frequently transmitted to Finance Apps and aggregators. Under LDBP, the DDA is never transmitted to any Finance App. It is replaced by an Alias ID that is opaque and app-specific.

Alias ID / Persistent Account Token An opaque, app-specific identifier generated by the bank as a substitute for the raw DDA number. The Alias ID is unique to each Finance App — if Stripe is issued alias_8822_stripe and that identifier is compromised, it provides no access to the User's account through any other app or bank. The User's real account number is never derivable from the Alias ID.

Scoped Account Token A cryptographic access token issued by the bank after the User selects a specific account. The token is mathematically restricted to that one account. The bank's Policy Enforcement Point will architecturally reject any request that attempts to access a different account using this token. Scoped Account Tokens are not standard OAuth bearer tokens — they encode account-level restrictions enforced at the PEP layer, not just at the application policy layer.

Intent-ID A session-based, single-use identifier generated by the Finance App for each transaction. The Intent-ID links a verification request to a specific execution, creating a verifiable authorization chain. Intent-IDs are single-use: a `False` response immediately invalidates the Intent-ID, and a successful execution permanently consumes it. Reuse of a consumed or invalidated Intent-ID is rejected.

Policy Enforcement Point (PEP) The bank's internal service that enforces LDBP rules before any API call touches the core ledger. The PEP checks token validity, account scope, fraud risk, weekly caps, and balance sufficiency — and returns a single composite Boolean result. The PEP is the architectural boundary that ensures raw financial data never leaves the bank. It is distinct from the core ledger, which holds the actual account data.

Atomic Operation / Atomic Wrapper A database operation in which multiple steps — in LDBP's case, balance verification, fraud evaluation, and fund deduction — are performed as a single indivisible unit. If any step fails, the entire operation is rolled back. Atomic execution prevents race conditions in which a User could spend the same funds twice through parallel API calls. Under LDBP, `POST /transfer/charge` performs all three steps atomically within the PEP.

Race Condition A timing vulnerability in which two or more operations access and modify shared state concurrently, producing unpredictable results. In banking, a common race condition occurs when a Finance App checks a balance, the User spends money elsewhere before the debit is processed, and the Finance App then executes a transfer against a balance that no longer exists. LDBP's Atomic Wrapper eliminates this vulnerability by combining the check and the debit into a single locked operation.

Idempotency / Idempotency Key The property of an operation that can be applied multiple times without changing the result beyond the first application. In LDBP, every fund transfer call includes an X-Idempotency-Key header. If a network failure causes the Finance App to retry a successful transfer, the bank recognizes the key and returns the original response without executing the transfer again. This prevents double-charging.

ACH (Automated Clearing House) The US electronic network used for direct bank-to-bank fund transfers, including direct deposits and bill payments. ACH transfers typically settle in 1-2 business days, creating a gap between when a balance is checked and when funds are actually debited. This gap is a known source of NSF (Non-Sufficient Funds) failures and fraud exposure. LDBP's atomic execution model eliminates this gap by performing the balance check and debit in a single real-time operation.

NSF (Non-Sufficient Funds) A condition in which a bank account does not hold enough money to cover a requested transfer. Under the current ACH model, NSF failures often occur because the balance check and the debit happen at different times. LDBP eliminates NSF failures caused by the check-to-debit gap through atomic execution at the PEP.

Kill Switch The user-facing mechanism by which the User can instantly revoke a Finance App's access to their account. Under LDBP, a single user action invalidates the Scoped Account Token, terminates all Finance App access, and voids any pending Intent-IDs — without requiring any action from the Finance App. The Kill Switch is implemented via `POST /auth/token/revoke` and is accessible through the bank's Connected Apps portal.

Notification Sovereignty The LDBP principle that when a verification is blocked for any internal reason — fraud flag, rate limit, revoked token, or weekly cap — the bank notifies the User directly, without routing the notification through the Finance App. The Finance App receives only a `False` response with no reason code. The User receives a human-readable notification via the bank's own channel. This ensures that security information about the User's account flows only between the bank and the User.

Principle Drift Any modification, extension, or implementation of LDBP that violates one or more of the five Least-Data Principles, regardless of intent, framing, or degree of compliance with other requirements. An implementation exhibiting Principle Drift may not claim LDBP conformance. The full Principle Drift taxonomy is defined in the LDBP Conformance Definition v1.0.

Open Banking A regulatory and technical framework that requires banks to provide third-party Finance Apps with programmatic access to customer account data, typically through APIs, with the customer's consent. Open banking regulations include CFPB Section 1033 in the United States, PSD2/PSD3 in the European Union, and equivalent frameworks in other jurisdictions. LDBP is designed as a Privacy-by-Design implementation standard for open banking APIs.

CFPB Section 1033 A provision of the Consumer Financial Protection Act codified as 12 CFR Part 1033, finalized by the CFPB on October 22, 2024. Section 1033 establishes the consumer's right to access and share their own financial data. It requires banks to provide developer interfaces (APIs) to authorized third parties. As of April 2026, the rule is subject to ongoing litigation and CFPB reconsideration. LDBP is designed to implement Section 1033's data minimization mandate as a technical guarantee rather than a legal assertion.

GDPR (General Data Protection Regulation) The European Union's comprehensive data protection law, effective May 25, 2018. GDPR establishes principles including Purpose Limitation (data collected only for stated purposes), Data Minimization (only necessary data collected), and the right to erasure. LDBP's architecture directly implements GDPR's Data Minimization and Purpose Limitation principles by ensuring Finance Apps never receive data beyond what is necessary for the specific transaction.

PSD3 / PSR (Payment Services Directive 3 / Payment Services Regulation) The European Union's updated payment services regulatory framework, successor to PSD2. PSD3 and the accompanying Payment Services Regulation ban the practice of screen scraping — using a User's login credentials to extract full bank account data — and require banks to provide dedicated, secure APIs for financial data access. LDBP exceeds PSD3's requirements by implementing Boolean-only verification, whereas PSD3-compliant APIs still typically return raw balances.

Regulation E (Electronic Fund Transfer Act) The US federal regulation, codified at 12 CFR Part 1005 and implementing the Electronic Fund Transfer Act (EFTA) of 1978, that establishes the rights, liabilities, and responsibilities of parties in electronic fund transfer transactions. Regulation E requires financial institutions to investigate and resolve consumer reports of unauthorized electronic fund transfers, provide error resolution procedures, and issue transaction receipts and periodic statements. Under Regulation E, consumers are entitled to limited liability for unauthorized transfers provided they report the error within specified timeframes. LDBP's Kill Switch, Real-time Consent Receipts, and verification.blocked webhook are designed to support a bank's Regulation E obligations by ensuring users are immediately notified of transfer activity and can revoke app access instantly — reducing the window during which unauthorized transfers can occur undetected.

Screen Scraping The practice of using a User's bank login credentials to programmatically access and extract data from the bank's web interface, as if logging in manually. Screen scraping was historically used by aggregators like Plaid to collect full account data. The practice is being banned under PSD3 in the EU. LDBP replaces screen scraping with server-to-server Boolean verification that never requires credential sharing.

FedNow / RTP (Real-Time Payments) US real-time payment networks operated by the Federal Reserve (FedNow) and The Clearing House (RTP) that enable near-instantaneous fund settlement, as opposed to the 1-2 day ACH settlement window. LDBP's atomic execution model is most naturally implemented on top of real-time payment infrastructure, though Phase 1 is achievable on standard ACH networks through the Atomic Wrapper mechanism.

Zero-Knowledge Proof (ZKP) A cryptographic method that allows one party to prove to another that a statement is true — for example, that a balance exceeds a threshold — without revealing any information beyond the truth of that statement. ZKPs are the cryptographically strongest implementation of LDBP's Boolean verification principle. ZKP implementation is designated as a Phase 2 or advanced capability; Phase 1 compliance achieves equivalent data minimization through internal comparison logic at the PEP.

A Note on Usage

The following terms are used interchangeably throughout this document and refer to the same concept:

- **Finance App, Payment App, App** — any third-party application connecting to the bank via LDBP
- **User, Account Holder, Consumer, Data Subject** — the individual who owns the bank account
- **Bank, Financial Institution, Implementing Institution** — the party that implements the LDBP protocol
- **Boolean Verification, is_enough check, Least-Data Check** — the core verification mechanism returning `True` or `False`
- **Kill Switch, Revocation, Token Revocation** — the user-initiated mechanism to terminate Finance App access
- **Alias ID, Persistent Account Token, Opaque Token** — the DDA substitute transmitted to Finance Apps

---

Appendix B: References

The following sources are cited in this document. Chicago citation style. All URLs verified as of April 2026.

[1] *In re Plaid Inc. Privacy Litigation*, Case No. 4:20-md-03056-DMR (N.D. Cal. July 20, 2022). Order Granting Final Approval of Class Action Settlement. Settlement administrator: https://www.plaidsettlement.com. Case summary: Lieff Cabraser Heimann & Bernstein, LLP. "Final Approval Granted to $58 Million Settlement in Plaid Consumer Privacy Lawsuit." July 21, 2022. https://www.lieffcabraser.com/2022/07/final-approval-granted-to-58-million-settlement-in-plaid-consumer-privacy-lawsuit/.

[2] Plaid, Inc. "Transactions." *Plaid Developer Documentation*. Accessed April 2026. https://plaid.com/docs/api/products/transactions/.

---

[3] Stripe, Inc. "Access Transactions for a Financial Connections Account." *Stripe Developer Documentation*. Accessed April 2026. https://docs.stripe.com/financial-connections/transactions. See also: "Financial Connections Balances." https://docs.stripe.com/financial-connections/balances.

---

[4] Crosman, Penny. "Eye on BNPL: Affirm's Underwriting Open Banking Reliance; Klarna Delves into P2P Payments." *Digital Transactions*, January 14, 2026. https://www.digitaltransactions.net/eye-on-bnpl-affirms-underwriting-open-banking-reliance-klarna-delves-into-p2p-payments/.

---

[5] Consumer Financial Protection Bureau. "Personal Financial Data Rights." Final Rule, 12 CFR Part 1033, Docket No. CFPB-2023-0052. *Federal Register* 89, no. 223 (November 18, 2024): 49084. https://www.federalregister.gov/documents/2024/11/18/2024-25079/required-rulemaking-on-personal-financial-data-rights. *Note: As of April 2026, this rule is subject to ongoing litigation (Forcht Bank, N.A. v. CFPB, No. 5:24-cv-00304, E.D. Ky.) and reconsideration by the CFPB. LDBP's technical approach remains valid regardless of the rule's litigation status.*

---

[6] Ng, Serena. "Introducing the Signal Payment Risk Platform: A Full Solution for ML-Powered Risk Assessments." *Plaid Blog*, October 1, 2025. https://plaid.com/blog/introducing-the-signal-payment-risk-platform/. See also: Plaid, Inc. "Signal." *Plaid Developer Documentation*. https://plaid.com/docs/signal/.

---

[7] Block, Inc. "Privacy Notice." *Cash App Legal*, effective February 9, 2026. https://cash.app/legal/us/en-us/privacy.

---

[8] Consumer Financial Protection Bureau. "CFPB Orders Operator of Cash App to Pay $175 Million and Fix Its Failures on Fraud." Press release, January 16, 2025. https://www.consumerfinance.gov/about-us/newsroom/cfpb-orders-operator-of-cash-app-to-pay-175-million-and-fix-its-failures-on-fraud/.

> *This document is published as LDBP v1.0 RFC. The API specification (v1.0), PRD (v1.0), Conformance Definition (v1.0), and supporting artifacts are available in the accompanying*

> *GitHub repository. Comments, implementation reports, and technical contributions are welcomed.*