LincPay Solutions Pvt. Ltd.

UPI Switch

Microservice Architecture

Issuer & Acquirer Services · Technology Stack · Integration Patterns · Security Architecture

12+	5000	99.99%	NPCI
Microservices	TPS Target	Uptime SLA	TSD v1.42

Agenda

01 **Architecture Overview**

System topology, zones, and NPCI connectivity

02 **API Gateway & Security Layer**

Rate limiting, mTLS, OAuth 2.0, Spring Cloud Gateway

03 **Issuer Microservices**

Transaction, VPA, Mandate, PIN/HSM, Debit posting services

04 **Acquirer Microservices**

Collect, Merchant, QR, Credit posting, Dispute, Settlement

05 **Shared & Infrastructure Services**

Fraud (LincShield), Notifications, Kafka, Redis, K8s

06 **Data Architecture**

PostgreSQL schema, Redis patterns, Kafka topics

07 **Security & HSM Deep Dive**

Thales K6, TR-31, PIN blocks, key rotation

08 **Scalability & Resilience**

K8s HPA, circuit breakers, DR strategy

Architecture Overview — System Topology

NPCI Cloud · UPI Gateway · VPA Mapper · Settlement SFTP · UDIR Portal

API Gateway Layer · Spring Cloud Gateway · OAuth 2.0 / JWT · mTLS Termination · Redis Rate Limiter · Nginx LB

ISSUER SERVICES

- Transaction Service
- VPA Service
- Mandate Service
- Balance / CBS Adapter
- PIN / HSM Service
- Debit Posting

ACQUIRER SERVICES

- Collect Request
- Merchant Service
- QR / Intent Service
- Credit Posting
- Dispute / UDIR
- Settlement Service

Event Bus: Apache Kafka · Redis Cluster · PostgreSQL · Elasticsearch · Kubernetes + Istio · Prometheus + Grafana

API Gateway & Security Layer

Spring Cloud Gateway

- Central entry point for all UPI traffic
- Dynamic routing by transaction type (Pay / Collect / Mandate)
- Predicates: path, header, VPA suffix-based routing
- Filters: JWT validation, request/response logging, timeout
- WebFlux (non-blocking, reactive) — handles 5000+ TPS
- Circuit breaker via Resilience4j at gateway level

Auth Service (OAuth 2.0 / JWT)

- OAuth 2.0 Authorization Server (Spring Authorization Server)
- JWT signed with RSA-256 (2048-bit key pair)
- Scopes: upi:pay, upi:collect, upi:mandate, upi:admin
- Token validity: 15 min access / 24 hr refresh
- Service-to-service: client credentials grant
- mTLS for NPCI-facing services (mutual certificate auth)

Rate Limiter (Redis Token Bucket)

- Redis-based token bucket via Lua atomic scripts
- Per-bank, per-VPA, per-IP rate limits configurable
- Burst capacity: 2x TPS allocation for peak windows
- 429 response with Retry-After header on breach
- Sliding window counters for fraud detection signals
- Zero-downtime config reload via Spring Cloud Config

Hardware Load Balancer

- Reverse proxy (TLS termination, HTTP/2)
- Layer 4 TCP load balancing to NPCI
- Upstream health checks every 5s, failover < 2s
- Sticky sessions for mandate flows (session affinity)
- Connection pooling: 10,000 concurrent upstream conns
- Access logs to ELK (all NPCI requests archived 7 years)

Issuer Microservices — Transaction, VPA & Mandate

Transaction Service

Spring Boot 3 · WebFlux · Kafka · PostgreSQL

- Handles Pay (P2P, P2M), Collect, and Refund flows per NPCI TSD v1.42
- Reactive pipeline: Netty TCP listener → JAXB XML parse → business validation
- Dedup check: Redis SETNX on TxnID + timestamp (30-day window)
- State machine: INITIATED → PENDING → SUCCESS / FAILURE / TIMEOUT
- Timeout management: 30s UPI standard, async retry with exponential backoff
- Publishes events to Kafka: txn.initiated, txn.completed, txn.reversed
- P99 latency target: <250ms internal (excluding NPCI roundtrip)

VPA Service

Spring Boot · PostgreSQL · Redis Cache

- VPA lifecycle: register, deregister, modify, link bank account
- VPA format validation: localpart@bankhandle (NPCI suffix rules)
- NPCI Mapper sync: push VPA registration via NPCI REST API
- VPA resolution cache: Redis TTL 5 min (reduces NPCI Mapper calls 80%)
- Multi-VPA per user: up to 5 VPAs, primary/secondary designation
- VPA status audit trail: all changes logged with IP + timestamp
- UPI Lite sub-wallet linkage to primary VPA

Mandate Service

Spring Boot · PostgreSQL · Quartz · Kafka

- Mandate types: One-time, Periodic (daily/weekly/monthly), As-presented
- Mandate lifecycle: CREATE → PENDING_AUTH → ACTIVE → PAUSED → REVOKED
- Execution engine: Quartz Scheduler triggers mandates on due date
- Pre-debit notification: T-1 day notification via Kafka → Notification Service
- Amount constraints: fixed amount, max-amount ceiling per execution
- Intent URL / QR for NACH-like use cases
- Auto-expiry: mandates expire if not executed within validity window

Issuer Microservices — PIN/HSM, Balance & Debit Posting

PIN / HSM Service

Thales Luna K6 · payShield 10K · TR-31

- PIN block generation: ISO 9564 Format 0 (XOR with PAN) and Format 3
- NPCI master key (ZMK) injected via TR-31 key block on HSM
- Working keys: ZPK (Zone PIN Key), ZAK, ZEK — auto-rotated every 90 days
- PIN verification: HSM compares encrypted PIN block vs stored PVV/PVKi
- Wrong PIN counter: Redis atomic increment, block at 3 failures (24hr lock)
- PIN change flow: old PIN verify → new PIN block → HSM re-encrypt → CBS update
- HSM HA: primary + secondary in active-standby, <100ms failover
- All HSM commands logged to tamper-evident audit log

Balance / CBS Adapter

CBS Nexora API · Spring Boot · Circuit Breaker

- Adapter pattern: single interface, multiple CBS implementations (CBSNexora, Finacle, BaNCS)
- Synchronous balance enquiry: CBS REST call with 3s timeout
- Available balance vs. ledger balance distinction (per RBI guidelines)
- Mini-statement: last 10 transactions from CBS, formatted per UPI TSD
- Circuit breaker (Resilience4j): open after 5 failures, half-open retry after 30s
- Shadow mode: parallel CBS call to new adapter without affecting response
- Masked account number in all UPI responses (last 4 digits only)

Debit Posting Service

Spring Batch · Kafka · PostgreSQL

- Consumes Kafka topic: txn.auth.success → posts debit to CBS ledger
- Two-phase: reserve funds (hold) on auth → actual debit on settlement
- Retry with idempotency key: CBS posting retried up to 3 times, then UDIR raised
- Reversal handling: auto-reversal triggered on timeout / NPCI failure response
- Spring Batch job for EOD batch reconciliation vs NPCI settlement file
- Reconciliation mismatch alerts: published to ops Kafka topic → Grafana
- All posting records archived to Cassandra for 7-year regulatory retention

Acquirer Microservices — Collect, Merchant & QR Services

Collect Request Service

Spring Boot · Kafka · Redis · WebFlux

- Initiates collect request: acquirer pushes to payer VPA via NPCI
- Expiry management: collect requests expire in 30 min (configurable)
- Push collect (P2M): merchant-initiated, customer approves in app
- Collect request state: SENT → PENDING → ACCEPTED / DECLINED / EXPIRED
- Redis TTL tracking: auto-expiry with Kafka tombstone on expiry
- Pending collect query: customer app polls for pending collects on VPA
- Batch collect: bulk collect requests for utility billing integrations

Merchant Service

Spring Boot · PostgreSQL · Redis

- Merchant onboarding: KYC verification, MCC code assignment, VPA creation
- Merchant categories: P2M, BBPS biller, e-commerce, offline POS
- Transaction limits: configurable per merchant tier (daily / per-txn)
- Merchant hierarchy: PSP → Sub-merchant → Terminal level
- Merchant VPA format: merchantname@bankhandle (NPCI registered)
- Real-time merchant dashboard: transaction volume, success rate, settlement
- Merchant deactivation: blacklist propagated to Redis for instant block

QR / Intent Service

Spring Boot · ZXing · Redis · Kafka

- Static QR: merchant VPA encoded, customer-entered amount
- Dynamic QR: amount + order ID + expiry embedded (30-min TTL in Redis)
- UPI Intent URL: upi://pay?pa=vpa&pn=name&am=amt&tr=refid
- QR formats: NPCI Bharat QR (EMVCo compliant), UPI QR v2.0
- Deep link generation for UPI apps (BHIM, PhonePe, GPay redirect)
- QR scan analytics: Kafka event per scan → merchant analytics dashboard
- Auto-void: unscanned dynamic QR TTL expiry with Redis keyspace events

Acquirer Microservices — Credit Posting, Dispute & Settlement

Credit Posting Service

Spring Batch · Kafka · PostgreSQL

- Consumes Kafka topic txn.credit.success → credits merchant CBS account
- NPCI net settlement file processed at EOD (T+0 settlement for UPI)
- Credit hold: funds held in nostro until NPCI confirms net position
- Failed credit alerts: raised to UDIR if CBS credit fails after 3 retries
- Spring Batch chunk processing: 10,000 credits/batch with skip/retry policy
- GL entries generated: Debit nostro → Credit merchant account
- Reconciliation report: credited vs NPCI file match rate published daily

Dispute / UDIR Service

Spring Boot · Elasticsearch · Kafka

- UDIR (UPI Dispute & Issue Resolution) TSD v2.0 compliant
- Dispute types: amount mismatch, duplicate charge, goods not delivered, technical decline
- Dispute lifecycle: RAISED → UNDER_REVIEW → RESOLVED / UPHELD / REJECTED
- TAT enforcement: 3-day auto-escalation to NPCI if bank doesn't respond
- Elasticsearch index: full-text search on TxnID, VPA, amount, date range
- Evidence upload: document attachments stored in MinIO (S3-compatible)
- Chargeback integration: auto-chargeback flow for eligible disputed transactions

Settlement Service

Quartz · SFTP · ClickHouse · Spring Batch

- NPCI settlement file download via SFTP at EOD (T+0, T+1 cycles)
- File formats: NPCI CSV settlement report, NACH-style batch files
- Net settlement calculation: all debits vs credits per bank per day
- Multi-bank settlement: separate ledger per bank with consolidated NPCI view
- ClickHouse analytics: settlement trends, recon rates, exception rates
- Settlement alerts: Grafana dashboard + SMS to ops team on mismatch
- 7-year archive: all settlement files stored in encrypted MinIO bucket

Shared Services — Fraud (LincShield), Notifications & Config

Fraud / Risk — LincShield

Spring Boot · ML Models · Redis · Kafka

- Real-time fraud scoring: every debit transaction scored before CBS posting
- Rule engine: velocity checks, amount anomaly, new device, unusual hours
- ML models: Isolation Forest (anomaly), XGBoost (binary classify fraud/legit)
- Feature store: Redis — user history, device fingerprint, geolocation
- Score threshold: >0.85 → block, 0.6-0.85 → step-up OTP, <0.6 → pass
- CFCFRMS integration: report suspicious transactions to RBI portal
- Feedback loop: confirmed fraud fed back to model via Kafka retraining pipeline

Notification Service

Spring Boot · SMS · FCM · Kafka · Redis

- Consumes Kafka: txn.completed, txn.failed, mandate.predebit topics
- Channels: SMS (transactional DLT-registered), FCM push, in-app, email
- Template engine: dynamic SMS templates (VPA, amount, TxnID, timestamp)
- DLT compliance: all SMS templates registered with Jio/Airtel DLT platform
- Rate limiting: max 10 SMS/hr per customer (regulatory cap)
- Deduplication: Redis to prevent duplicate notifications on retry
- Failed notification retry: exponential backoff (1s, 5s, 30s, 5min)

Config & Audit Services

Spring Cloud Config · Vault · ELK · Jaeger

- Spring Cloud Config Server: centralized YAML configs per bank per env
- HashiCorp Vault: secrets (DB passwords, API keys, HSM credentials) at rest
- Dynamic refresh: @RefreshScope beans updated without restart via /actuator/refresh
- Audit service: every API call logged with user, IP, request hash, response code
- Jaeger distributed tracing: trace UPI txn across 6+ microservices end-to-end
- ELK Stack: Logstash ingest → Elasticsearch index → Kibana dashboards
- Retention policy: audit logs 7 years (RBI mandate), operational logs 90 days

Data Architecture — Kafka Topics, Redis Patterns & DB Schema

Apache Kafka Topics

upi.txn.initiated
New pay/collect request

upi.txn.completed
Success/failure result

upi.txn.reversed
Reversal/timeout

upi.mandate.predebit
T-1 mandate notification

upi.fraud.alert
LincShield high-risk

upi.credit.posted
Acquirer credit done

upi.settlement.ready
EOD settlement file

upi.ops.alert
Recon mismatch / error

Redis Key Patterns

dedup:{txnId} — TTL: 30d
30-day SETNX dedup

vpa:{vpa} — TTL: 5m
VPA resolution cache

ratelimit:{bankId} — TTL: 1m
Token bucket counter

session:{userId} — TTL: 15m
Auth session token

pin:fail:{vpa} — TTL: 24h
Wrong PIN count

qr:{qrId} — TTL: 30m
Dynamic QR payload

fraud:{userId} — TTL: 7d
Feature store (ML)

mandate:{id} — TTL: no-expiry
Active mandate status

PostgreSQL — Core Tables

upi_transactions
TxnID, VPA, amount, status, timestamps

vpa_registry
VPA, account, IFSC, status, bank_id

mandates
Mandate ID, type, amount, validity, state

merchants
MerchantID, MCC, VPA, tier, limits

dispute_cases
DisputeID, TxnID, type, status, TAT

settlement_files
FileID, date, net_amount, status

audit_log
UserID, action, IP, timestamp, hash

bank_config
BankID, CBS URL, limits, HSM slot

Security Architecture — HSM, TR-31 & Key Lifecycle

HSM Hardware (Thales Luna K6)

- FIPS 140-2 Level 3 certified hardware security module
- Key storage: ZMK, ZPK, ZAK, ZEK in tamper-proof boundary
- TR-31 key block import from NPCI (AES-256 wrapping)
- RSA-2048 signing keys for NPCI message integrity
- Dual-control m-of-n (3-of-5) key ceremony for ZMK
- Active-standby HA: <100ms failover between HSM units

Key Lifecycle & Rotation

- ZMK: injected offline via smart cards, 1-year rotation
- ZPK (Zone PIN Key): derived from ZMK, auto-rotated every 90 days
- ZAK / ZEK: MAC and encryption keys, 90-day rotation via NPCI
- TR-31 import: NPCI sends encrypted key block, HSM unwraps/stores
- Key rotation: zero-downtime — old key retained 24hr for inflight txns
- Audit: every HSM op logged to tamper-evident WORM secure log

PIN Block Flow — ISO 9564 Format 0

1. Customer enters PIN in UPI app
2. App encrypts PIN block (ISO Format 0: XOR with PAN)
3. Encrypted block sent over HTTPS to Switch
4. Switch forwards to HSM via PKCS#11 API call
5. HSM decrypts → re-encrypts with CBS ZPK → sends to CBS
6. CBS verifies PVV → auth result returned to NPCI

Scalability, Resilience & DevOps

Kubernetes HPA

- HPA based on CPU (70%) + Kafka consumer lag
- Min 2 pods, max 20 pods per service
- PodDisruptionBudget: min 1 pod always running
- Istio sidecar: automatic mTLS between pods

Circuit Breaker

- Resilience4j on all CBS and NPCI calls
- Config: 5 failures in 10s → circuit opens
- Half-open: 3 test calls after 30s
- Fallback: cached response or graceful error

DR Strategy

- Active-Passive DC: RTO < 30 min, RPO < 5 min
- Kafka MirrorMaker 2 for cross-DC topic replication
- PostgreSQL streaming replication (synchronous)
- NPCI mandated DR drill: quarterly

CI/CD Pipeline

- Jenkins pipeline: build → unit test → SonarQube → Docker build
- Docker image scan: Trivy (CVE check before push)
- Helm chart deploy to K8s: blue/green per service
- Rollback: helm rollback within 2 min on failed health check

Observability

- Prometheus: 200+ metrics per service (JVM, Kafka lag, CBS latency)
- Grafana: 5 dashboards (TPS, latency, error rate, HSM, settlement)
- Jaeger: distributed trace for every UPI transaction
- PagerDuty: P1 alert if success rate drops below 98% for 2 min

Performance

- Target: 5000 TPS sustained, 8000 TPS burst (15s)
- P50 latency: <80ms, P99: <500ms end-to-end
- JMeter load test: 3-hour sustained run pre-NPCI certification
- Connection pool tuning: HikariCP max-pool-size=50 per service

Production-Ready

UPI Switch Architecture

12+ microservices · NPCI TSD v1.42 compliant

Issuer + Acquirer fully separated service layers

Thales HSM · TR-31 · ISO 9564 PIN security

5000 TPS target · 99.99% SLA · K8s multi-AZ