Cyber Security Innovation Challenge 1.0

PROTOTYPE DEVELOPMENT

Stage – III

Team Name – MandelBrot

Team Lead Name – Mizanur Rahaman

Team Lead University – Jadavpur University

Problem Statement Domain – Mule Accounts & Collusive Fraud in UPI

Solution Subtitle – FinGuard-Real-Time Multi-Signal Mule Account Detection

Problem Statement & Context

The Mule Account Crisis in UPI

- UPI processed 13.1 billion transactions in Oct 2024 (₹20.64 lakh crore)
- Mule accounts: bank accounts used as pass-throughs for laundering stolen money
- Individual transactions look normal; fraud is in the coordination and network patterns
- Current rule engines can't see networks, go stale, and offer no nuanced risk scoring
- RBI mandates enhanced fraud monitoring; NPCI pushes for multi-dimensional detection

Our Formulation

- Build transaction graph $G = (A, E)$ from UPI account data
- Five-signal ensemble: Behavioral (25%), Graph (40%), Device (15%), Temporal (10%), ML (10%)
- $R(a) = \Sigma\, w_k \cdot S_k(a) + \text{Boost}(\{S_k(a)\})$
- Confidence boosting: +8 to +20 when 2-4 independent signals agree
- Risk tiers: CRITICAL (≥85), HIGH (70-84), MEDIUM (40-69), LOW (<40)
- Target: sub-50ms per-account scoring latency

13.1B txns/month | ₹20.64L Cr volume | 300+ banks on UPI | 5 detection signals | <50ms scoring

Review of Existing Solutions and Research

Capability	Rule-Based	Supervised ML	Graph Methods	FinGuard (Ours)
Temporal Pattern Detection	Static thresholds	Limited	None	Full (5 sub-signals)
Network/Graph Analysis	None	None	Yes	Yes (3 patterns + DFS)
Device Correlation	None	Partial	None	Full
Unsupervised (No labels)	N/A	No	Partial	Yes (IF + Z-score)
Real-Time Scoring	Fast	Moderate	Slow	Fast (<50ms)
Explainability	Clear	Black-box	Limited	Full (3-5 evidence)
Multi-Signal Ensemble	No	No	No	Yes (5-factor weighted)
Confidence Boosting	No	No	No	Yes (multi-signal)

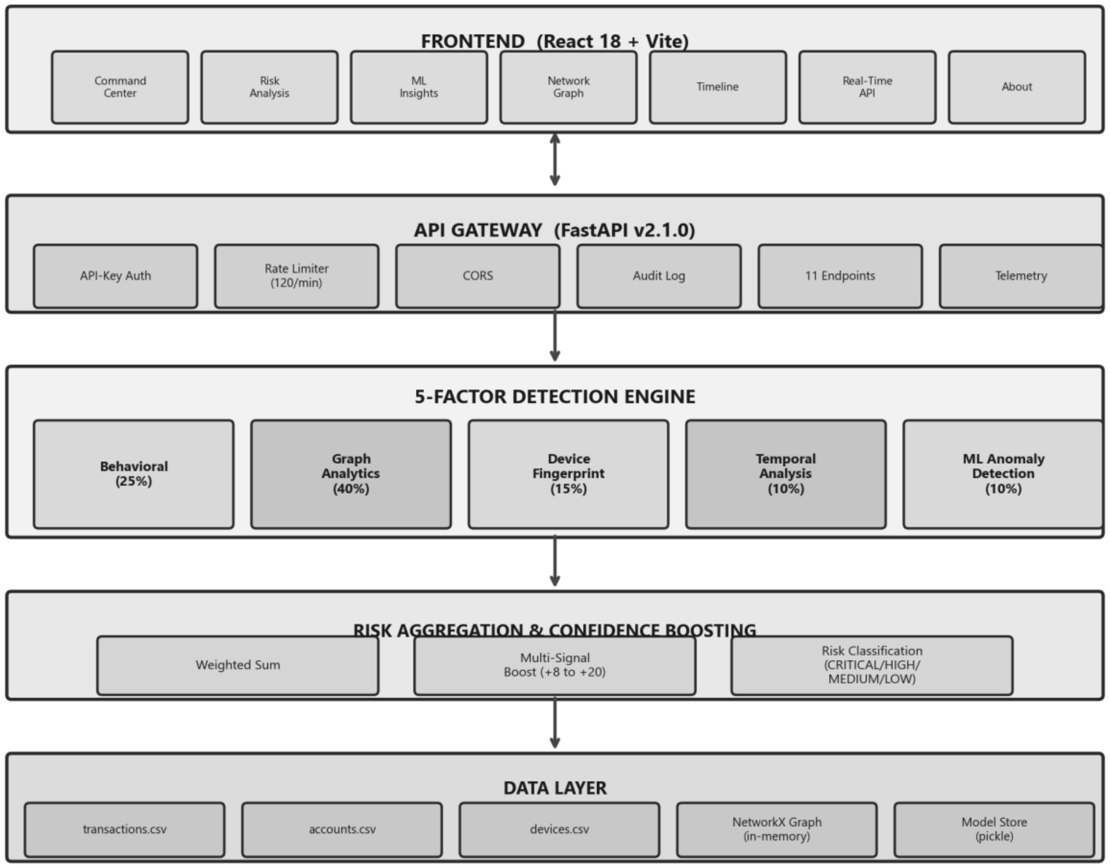
Key Research

- MuleTrack (Jambhrunkar 2025): Lightweight temporal learning for mule detection
- GNN Review (Cheng 2024): Graph methods beat classifiers for coordinated fraud
- Node2Vec (Caglayan 2022): Graph embeddings improve laundering detection
- Community Detection (Huang 2025): Mules found via graph communities
- Isolation Forest (Liu 2008): Unsupervised anomaly detection without labels
- Neo4j (2023): Industry case studies on graph-based fraud detection

Gap: No system combines all 5 signals (behavioral + graph + device + temporal + ML) with explainability and confidence boosting. FinGuard fill gap.

Proposed Solution – Architecture & Approach

FinGuard: System Architecture



Five Detection Modules

- Behavioral (25%): Velocity, flow asymmetry, amount anomalies, new account flags
- Graph Analytics (40%): Star, chain, circular patterns. Custom DFS/BFS. $O(V \cdot d)$
- Device Fingerprinting (15%): Multi-account device sharing + device rotation
- Temporal (10%): Burst detection, night activity, velocity spikes, bot signatures
- ML Anomaly (10%): Custom Isolation Forest (NumPy), 17 features, Z-score ensemble

Risk Aggregation

$$R = 0.25 \cdot S_B + 0.40 \cdot S_G + 0.15 \cdot S_D + 0.10 \cdot S_T + 0.10 \cdot S_{ML} + Boost$$

Signals Agreeing	Boost
≥4 signals above threshold	+20
≥3 signals above threshold	+15
≥2 signals above threshold	+8
Graph≥30 AND Device≥15	+10
Behav≥40 AND Graph≥40 AND Device≥30	+12

Innovation and Novelty Elements

Five-Signal Ensemble

Combines behavioral + graph + device + temporal + ML. Redundancy: evading all 5 signals simultaneously is near-impossible. Others use 1-2 signals max.

Multi-Signal Confidence Boosting

Independent signals agreeing carries evidential weight. +8 to +20 boosts when 2-4 signals flag the same account. Same logic as ensemble ML, applied at risk aggregation.

Zero-Label ML Detection

Custom Isolation Forest in pure NumPy (~200 lines). No labelled training data needed. Deploy on a new platform on day one and it starts flagging outliers immediately.

Efficient Graph Algorithms

Custom DFS cycle detector with depth cap of 6 replaces exponential nx.simple_cycles(). BFS chain detection. $O(V \cdot d)$ time complexity.

Explainability by Design

Every detection module generates 3-5 specific evidence items as core logic, not an afterthought. Investigators see exactly why an account was flagged.

Production-Grade Security

API-key auth, rate limiting (120/min), restricted CORS, JSON audit logs with request IDs, non-root Docker. Table-stakes for banking systems.

Unique Selling Proposition (USP): Proposed Solution vs. Existing Solutions (Relevance to Industry)

Key Differentiators

- 5 signals vs. 1-2: catches accounts that single-model systems miss entirely
- Graph-first (40% weight): reveals star, chain, and loop patterns of organized rings
- No training data required: Isolation Forest runs unsupervised from day one
- Full explainability: component breakdowns + evidence + confidence for every score

Feature	Static Rules	ML Model	FinGuard
Detection Signals	1 (rules)	1 (features)	5 (ensemble)
Graph Awareness	None	None	Full
Device Correlation	None	Partial	Full
Labels Required	No	Yes	No
Explainability	High	Low	High
Real-Time	Yes	Moderate	<50ms
Confidence Levels	No	No	Yes

Business Model & Market Viability

- Target: 300+ banks and 50+ PSPs on UPI, most still using static rule engines
- Market: fraud losses in thousands of crores; each investigation costs ₹15K-25K in analyst time
- Pricing: Starter (₹50K/mo), Enterprise (₹3-5 lakh/mo with SLA), Infrastructure (custom licensing)
- Deployment: SaaS API for small banks, on-premise Docker/K8s for large banks and NPCI

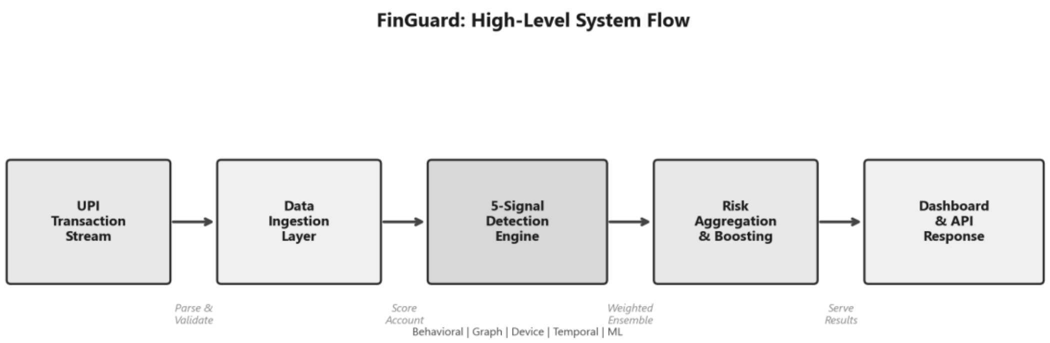
Prototype Demonstration & Real-World Deployment

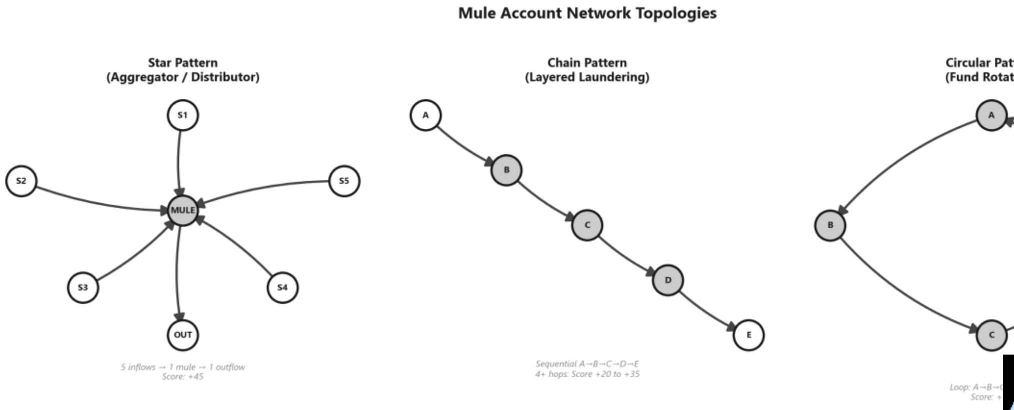
Technology Stack

Component	Technology	Version
Backend API	FastAPI + Uvicorn	2.1.0
Frontend	React + Vite	18.x / 5.x
Language	Python	3.11
Graph Engine	NetworkX	3.2.1
ML Engine	Custom Isolation Forest	NumPy 1.26
Containers	Docker + Compose	Multi-stage

Test Scenarios (6 Mule Patterns)

Scenario	Pattern	Expected Risk
Star Aggregator	5→1 mule→1 dist.→3 sinks	CRITICAL/HIGH
Circular Network	4-node loop + shared device	CRITICAL
Chain Laundering	5-node sequential chain	HIGH
Device Ring	3 accounts, 1 shared device	HIGH/MEDIUM
Rapid Onboarding	1-day acct, 13 txns in 30min	CRITICAL
Night Smurfing	12+ txns between 1-4 AM	HIGH





Prototype Demonstration & Real-World Deployment

Detection Results: 100% Mule Detection, 0% False Positives

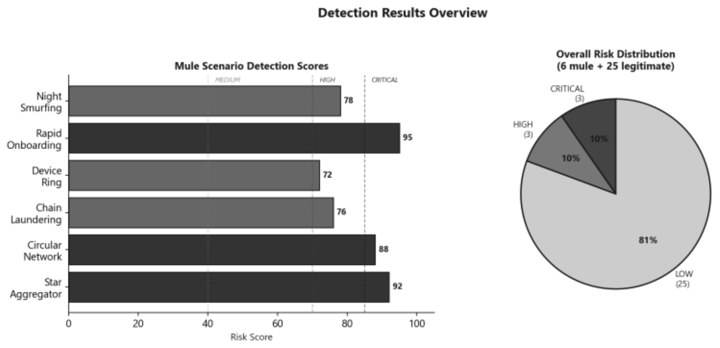
Scenario	Score	Level	Primary Evidence
Star Aggregator	92	CRITICAL	Star pattern (5→1 out)
Circular Network	88	CRITICAL	Cycle + shared device
Chain Laundering	76	HIGH	Deep chain (4+ hops)
Device Ring	72	HIGH	Shared device (3 accts)
Rapid Onboarding	95	CRITICAL	Burst + 1-day account
Night Smurfing	78	HIGH	Night activity (85%)

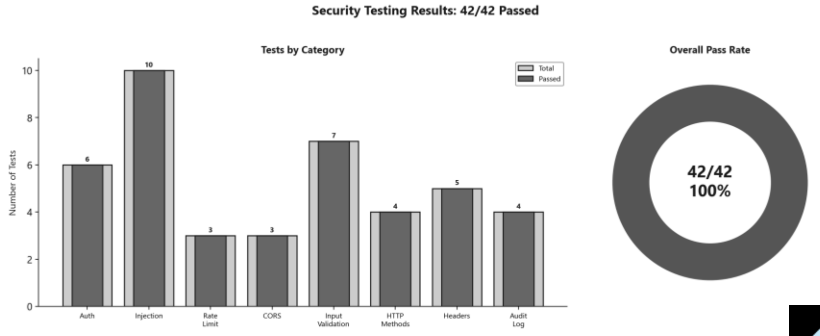
Security Testing: 42/42 Tests Passed

Category	Tests	Passed	Result
API Key Authentication	6	6	100%
Injection Attacks	10	10	100%
Rate Limiting	3	3	100%
CORS Policy	3	3	100%
Input Validation	7	7	100%
HTTP Method Restriction	4	4	100%
Security Headers	5	5	100%
Audit Logging	4	4	100%
	42	42	100%

Performance

Metric	Value
Single account scoring	<50ms
Batch (30 accounts)	<500ms
API startup	<3s
Memory footprint	<150MB





Limitations and Constraints

Current Limitations

- Synthetic Data: All testing on generated data; real-world patterns are messier
- Static Graph: Built once at startup from CSV; production needs incremental updates
- In-Memory Only: Everything in RAM; won't scale to UPI's billions of transactions
- No GNNs: Hand-crafted graph features; GNNs could learn subtler patterns
- Fixed Weights: Ensemble weights set manually, not learned from data
- Batch ML: Isolation Forest trains once; needs online/incremental learning
- Single-Hop Devices: Direct sharing only, no transitive multi-hop chains

Challenges Encountered

- Cycle Detection: nx.simple_cycles() hung on dense graphs; custom DFS with depth cap solved it
- Ensemble Calibration: Finding right signal weights required dozens of experiments
- Explainability vs. Privacy: Evidence items raise questions about information surfacing
- Cross-Platform: Windows dev → Linux Docker deployment caused path and encoding issues

Every limitation has a concrete production solution mapped in our 16-week MVP roadmap. These are engineering challenges with known solutions, not fundamental design flaws.

Roadmap and Strategy Towards Minimum Viable Product (MVP)

Phase 1: Infrastructure (Weeks 1-4)

- Apache Kafka for real-time ingestion
- PostgreSQL for persistent storage
- Neo4j graph DB with incremental updates
- Redis for sub-ms hot data cache

Phase 2: Advanced Detection (Weeks 5-8)

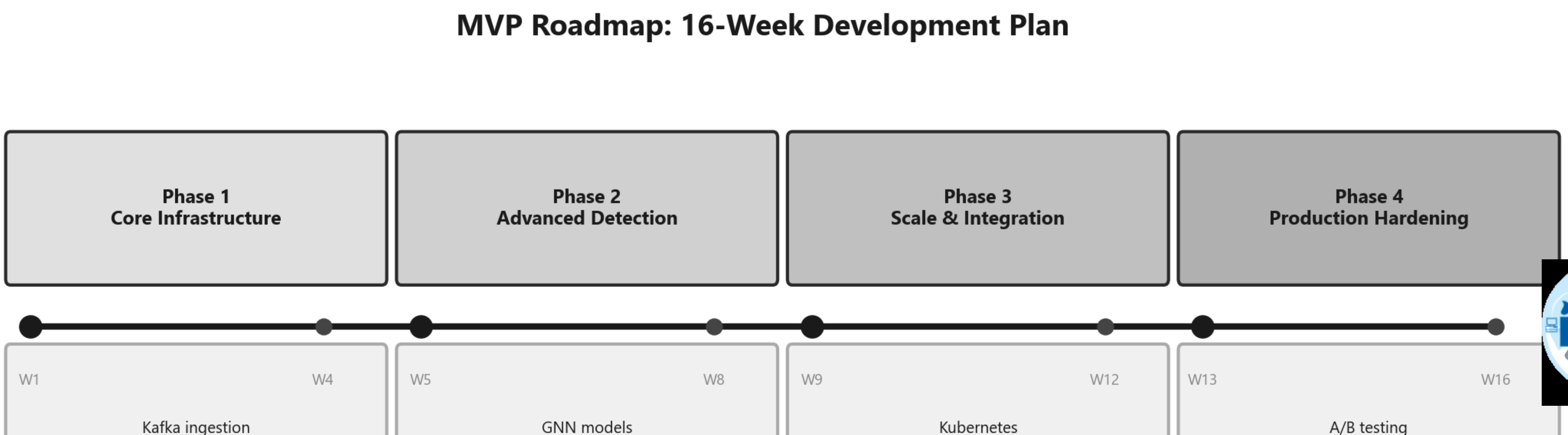
- GNN-based node classification
- Online/incremental anomaly detection
- Bidirectional graph analysis
- Multi-hop transitive device chains

Phase 3: Scale & Integration (Weeks 9-12)

- Kubernetes with horizontal autoscaling
- UPI switch plugin for inline scoring
- Case management workflow
- Investigator feedback loop

Phase 4: Production Hardening (Weeks 13-16)

- A/B testing framework (shadow mode)
- Automated SAR generation (RBI)
- SOC 2 certification
- Sub-10ms scoring at 10,000 TPS



Team Composition and Individual Contributions

Team Leader

Name – [Name]

University – [University]

DOB – [DOB]

Contribution – System architecture, risk engine, ensemble weights, report

Member 1

Name – [Name]

University – [University]

DOB – [DOB]

Contribution – Graph analysis, behavioral analysis, DFS cycle detection, BFS chains

Member 2

Name – [Name]

University – [University]

DOB – [DOB]

Contribution – Custom Isolation Forest (NumPy), 17-feature pipeline, Z-score, SHAP

Member 3

Name – [Name]

University – [University]

DOB – [DOB]

Contribution – React dashboard (8 tabs), network graph vis, API integration, UX

Member 4

Name – [Name]

University – [University]

DOB – [DOB]

Contribution – Docker, security middleware, data generation, temporal analysis

References and Citations

[1] NPCI, "UPI Product Statistics," 2024. npci.org.in/what-we-do/upi/product-statistics

[2] RBI, "Master Direction on Digital Payment Security Controls," RBI/2020-21/74, 2021.

[3] NPCI, "UPI Fraud Monitoring and Risk Management Guidelines," 2023.

[4] S. Panigrahi et al., "Rule-Based and ML Methods for Fraud Detection," J. King Saud Univ., 2022.

[5] E. Lopez-Rojas et al., "Applying AI and ML in Financial Services," IEEE Access, 2022.

[6] G. Jambhrunkar et al., "MuleTrack: Temporal Learning for Money Mule Detection," IWANN, 2025.

[7] D. Cheng et al., "GNNs for Financial Fraud Detection: A Review," arXiv:2411.05815, 2024.

[8] M. Caglayan & S. Bahtiyar, "Money Laundering Detection with Node2Vec," Gazi Univ. J., 2022.

[9] Z. Huang, "Money Mules Detection on Transaction Graphs," ACM GAIB, 2025.

[10] Neo4j Inc., "Accelerate Fraud Detection with Graph Databases," Whitepaper, 2023.

[11] F. T. Liu et al., "Isolation Forest," 8th IEEE Int. Conf. Data Mining, pp. 413-422, 2008.

Project Links

Resource	Link
GitHub Repository	[Insert GitHub Repo URL]
Live Deployed URL	[Insert Deployed URL]
Demo Video (YouTube)	[Insert YouTube Link]
API Documentation	[Insert /docs URL when deployed]