Deterministic Peel Chains for Regulatory Compliance in Autonomous, Offline Environments

Christopher I. V. Farmer, LL.M. (University of Worcester) · Independent Researcher · civfarmer@gmail.com

Preprint; not yet peer reviewed. Version 1.1, October 2026. This document describes a working reference implementation, the *Forensic & Regulatory Intelligence Suite* (FRIS). All private-party data used to exercise the system is synthetic and labelled as such; the regulatory, sanctions and enforcement reference data are real, dated and source-linked, as described in §6.

DOI: https://doi.org/10.5281/zenodo.23462154

Licence: Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0). Commercial use needs a licence from the author: civfarmer@gmail.com.

Abstract

Financial-crime and regulatory-compliance analysis is increasingly performed with cloud-hosted, machine-learning tooling whose outputs are probabilistic, whose training data and model weights are opaque, and whose operation requires transmitting sensitive subject data to third-party infrastructure. For work that must be reproducible under evidentiary scrutiny and confidential by construction, these properties are liabilities rather than features. This paper describes an alternative architecture, realised in a working reference system (the Forensic & Regulatory Intelligence Suite, FRIS), organised around three commitments: (1) an *offline, dependency-free, single-file* client architecture, autonomous in the sense that it needs no server, external service or network connection, in which analysis executes entirely on the analyst's own machine and no subject data is transmitted anywhere; (2) a *deterministic peel-chain* method for tracing layered value transfers, in which value is represented as exact integers, every hop conserves value under an explicit, checked invariant, and the entire trace is byte-for-byte reproducible from its inputs — in deliberate contrast to probabilistic clustering and learned classifiers; and (3) a *provenance-first* data model in which every reported value is labelled by source, real and synthetic inputs are never mixed silently, and unresolved gaps (unknown ownership percentages, nominee interposition, circular holdings) are surfaced explicitly rather than imputed. We situate the design against the blockchain-analytics, RegTech/SupTech, deterministic-forensics and air-gapped-tooling literatures; specify the peel-chain algorithm and its value-conservation invariant with pseudocode; describe the 23-module engine framework that applies a common deterministic discipline across beneficial-ownership tracing, sanctions and adverse-media screening, an insolvency-recovery waterfall, and a structured compliance knowledge base; and discuss reproducibility, auditability, limitations and future work. No performance benchmarks are claimed, and any numeric example is labelled illustrative.

Keywords: deterministic algorithms; financial-crime compliance; anti-money-laundering (AML); beneficial ownership; peel chain; blockchain forensics; RegTech; air-gapped computing; local-first software; reproducibility; auditability; provenance; sanctions screening; insolvency waterfall

---

1. Introduction

The tooling used to investigate financial crime and to demonstrate regulatory compliance sits at an awkward intersection of three requirements that current mainstream architectures satisfy poorly at the same time: **confidentiality**, **reproducibility**, and **explainability**. An analyst tracing the movement of illicit value, resolving the ultimate beneficial owner of a layered corporate structure, screening a counterparty against sanctions lists, or modelling creditor recoveries in an insolvency handles data that is by definition sensitive — subject identities, suspected typologies, un-adjudicated allegations, privileged legal strategy. The same analyst may later have to defend the resulting findings: to a supervisor, to an auditor, to opposing counsel, or to a court applying an admissibility standard for expert evidence. Increasingly, the dominant tooling answers the first requirement (confidentiality) by moving the data *off* the analyst's machine to a managed cloud service, and answers the third (explainability) with machine-learning models whose outputs are scores rather than derivations. Both moves are in tension with reproducibility.

This paper describes a deliberately contrarian architecture that resolves the three requirements jointly by giving up the assumptions that create the tension. The reference implementation, the Forensic & Regulatory Intelligence Suite (FRIS), makes three design commitments.

First, it runs offline, with no runtime dependencies, as a single file. The complete application — user interface, analysis engines and an embedded data snapshot — is delivered as one self-contained HTML file (on the order of three megabytes) that a user double-clicks and that opens in any modern browser with no server, no database engine, no installation, and, critically, no network access. When analysis runs, subject data never leaves the analyst's machine. This is not a convenience feature bolted onto a networked product; it is the security posture. In regulated financial-crime work the most robust data-protection control is not encryption-in-transit but *the absence of transit*. An air-gapped, single-file tool is auditable in a way a cloud pipeline is not: there is no server-side state, no telemetry, no vendor with incidental custody of the subject's data, and nothing to subpoena from a third party.

Second, it traces layered value transfers deterministically. The central analytical primitive — which we call a *deterministic peel chain* — models the movement of value through a sequence of layers (crypto hops in the first instance) using exact integer arithmetic and an explicit conservation invariant, and the same traversal discipline carries over to holding companies and distribution tiers (§4.4). Given the same inputs, the method produces byte-for-byte identical output, every time, on any machine, with no reliance on ambient randomness, wall-clock time, platform-dependent floating-point behaviour, model weights, or network calls. This is the opposite of the probabilistic clustering and learned classification that characterise mainstream blockchain analytics, and it is chosen precisely for the property that probabilistic methods sacrifice: a result an analyst can *re-derive and defend*, line by line, rather than merely report.

Third, it is provenance-first. Every value the system reports is labelled by where it came from. Real, publicly-sourced regulatory and sanctions data is never silently mixed with synthetic demonstration data; each is tagged, and the regulatory instruments link to their official sources. Where the data does not support a conclusion — an unknown ownership percentage, a nominee shareholder masking the real owner, a circular holding that breaks clean attribution — the system says so, and marks the ownership *not fully attributable*, rather than imputing a plausible-looking number. Showing gaps, rather than filling them, is treated as a correctness property.

These commitments are unfashionable. The prevailing direction of travel in the field is toward more cloud, more machine learning and more inferred attribution. Our argument is not that probabilistic and cloud methods are wrong in general — for triage at population scale they are often the right tool — but that there is a distinct and important class of work, *evidentiary* financial-crime and compliance analysis on *sensitive* data by *individual* analysts, for which

determinism, offline operation and explicit provenance are the correct defaults, and for which a small, framework-free, fully-inspectable codebase is an asset rather than a limitation.

The remainder of the paper proceeds as follows. Section 2 situates the design in the related literature on blockchain analytics, RegTech/SupTech, deterministic versus learned forensics, evidentiary admissibility, and air-gapped and local-first tooling. Section 3 describes the offline single-file system architecture and the determinism primitives on which the value-tracing and simulation engines draw. Section 4 specifies the deterministic peel-chain method, its value-conservation invariant, and its three operating modes, with pseudocode, and shows how the same abstraction recurs in beneficial-ownership tracing and the insolvency waterfall. Section 5 surveys the 23-module engine framework. Section 6 details the provenance-first data model and the reproducibility argument. Section 7 states limitations and future work honestly, and Section 8 concludes.

---

2. Background and Related Work

2.1 Blockchain analytics and the "peel chain"

The peel chain (called a "peeling chain" in the early measurement literature) is a well-documented money-laundering typology in cryptocurrency: a wallet receives a large amount and then, across a sequence of transactions, forwards most of its balance onward while "peeling" a small amount at each step toward a cash-out destination, so that the bulk of the value hops from address to address while thin streams exit to exchanges, merchants or mixers [17]. Recognising and reconstructing such chains is a core task of on-chain forensics.

The dominant commercial approach — associated with platforms such as Chainalysis, TRM Labs and Elliptic — combines *clustering heuristics* with *attribution intelligence*. The best-known heuristic, co-spend (or multi-input) clustering, assumes that addresses appearing together as inputs to one transaction are controlled by one entity [17]; behavioural and change-address heuristics extend this. These techniques are powerful and scale to the whole chain, but they are, by construction, **probabilistic**. Recent scrutiny has been pointed: a February 2026 practitioner commentary, whose author discloses having contributed to an amicus brief on the same question, reported preliminary out-of-sample tests in which even the oldest and most heavily used heuristic, co-spend clustering, "can fail badly under realistic circumstances", with error rates materially higher than vendors' public claims, and called the validation work done on these tools to date "grossly inadequate" [1]. Practitioner commentary stresses that heuristic attribution yields confidence levels, not certainties [2], and defence-side commentary that behavioural fingerprints are circumstantial and susceptible to deliberate mimicry, and that these uncertainties matter acutely when tracing output is offered as evidence [3]. Chainalysis itself frames blockchain forensics against the *Daubert* admissibility standard for expert testimony, which asks whether a method is testable, has a known error rate, and is generally accepted [4].

Our contribution is orthogonal to the heuristic-quality debate. Rather than seeking a better probabilistic estimate of *which real-world entity controls which address*, the deterministic peel-chain method addresses the narrower, tractable problem of *reconstructing and accounting for the value flow through an assumed or given chain, exactly and reproducibly*. It does not claim to attribute addresses to persons; it claims that, given a chain of transfers, the arithmetic of what was forwarded, peeled and paid in fees is conserved, checked, and identically reproducible. The two approaches are complementary [18]: probabilistic clustering proposes the chain; deterministic accounting audits it.

2.2 RegTech, SupTech and explainability

RegTech (regulatory technology) denotes technology used by regulated firms to comply with financial regulation efficiently; *SupTech* denotes technology used by supervisors to oversee them [5, 6]. A substantial body of work applies data analytics and machine learning to anti-money-laundering (AML) transaction monitoring, sanctions screening and know-your-customer (KYC) onboarding [7, 8]. A recurring theme — and a recurring criticism — is the tension between detection performance and *explainability*: a model that flags a transaction is of limited use to a compliance officer who must file a defensible suspicious-activity report, or to a supervisor who must understand why an alert fired, unless the reason is legible. FRIS takes an uncompromising position on this axis: every analytical output it produces carries a plain-language reason and is composed from named, weighted factors, so that "why did this score come out this way" is answerable by reading the derivation, not by interrogating a model. This is a deliberate trade of raw detection breadth for total explainability, appropriate to the evidentiary use case.

2.3 Deterministic forensics, reproducibility and admissibility

The digital-forensics literature treats reproducibility as foundational: investigators are expected to document every step so that findings are transparent and reproducible, and reliability-validation frameworks have been proposed to place forensic methods on a testable footing [9]. Under *Daubert v. Merrell Dow Pharmaceuticals* (1993) and its progeny, expert methods are assessed for testability, peer review, known error rate and general acceptance, and *Daubert* has become a de facto reliability standard for digital evidence [4, 10]. Recent work argues explicitly for *deterministic* forensic pipelines with formal specifications precisely because determinism underwrites replicability [11]. FRIS is designed to sit comfortably inside this frame: a deterministic computation has, in a strong sense, a *zero* method-induced error rate — run it again and you get the same answer — so the residual uncertainty is located entirely and visibly in the *inputs*, which is where, for evidentiary purposes, it belongs.

2.4 Air-gapped and local-first tooling

The security and privacy literatures have converged on a simple observation: the strongest way to prevent sensitive data from leaking to an adversary, a cloud vendor or a foreign jurisdiction is to keep it on-premises, and for the highest-sensitivity work, air-gapped [12, 13]. This is increasingly framed as a compliance and data-sovereignty requirement: where sector rules forbid certain data to be stored or processed on multi-tenant cloud infrastructure, "an air-gapped, on-premises deployment is often the only way to achieve compliance", and such a deployment lets an organisation "guarantee that data never leaves a specific geographic location or legal jurisdiction" [12]. In parallel, the *local-first software* movement articulates a set of ideals — the user retains ownership and custody of their data, the software works without a network, and the data outlives any vendor [14]. FRIS is an unusually literal instance of these ideals: because the entire application is one offline file with no backend, the analyst's data is never in anyone else's custody, the tool functions with the network cable unplugged, and a saved copy of the file keeps working indefinitely regardless of the author's continued involvement.

2.5 Beneficial-ownership transparency

Recommendation 24 of the Financial Action Task Force (FATF), revised in March 2022, requires countries to ensure "that there is adequate, accurate and up-to-date information on the beneficial ownership and control of legal persons that can be obtained or accessed rapidly and efficiently by competent authorities" [15]. In the context of legal persons, the FATF Glossary defines the beneficial owner as "the natural person(s) who ultimately owns or controls a

customer and/or the natural person on whose behalf a transaction is being conducted", including "those natural persons who exercise ultimate effective control over a legal person", and explains that ultimate ownership or control covers "situations in which ownership/control is exercised through a chain of ownership or by means of control other than direct control" [15, 16]. The *chain of ownership* is the connective tissue of this paper: resolving beneficial ownership through a chain of holdings is structurally the same problem as tracing value through a chain of transfers. FRIS treats them with the same deterministic traversal discipline, and — consistent with the transparency literature's insistence on *accurate* information — refuses to manufacture an ownership percentage it cannot derive, flagging the structure as not fully attributable instead.

---

3. System Architecture

3.1 Offline, dependency-free, single-file delivery

FRIS is delivered in two principal forms from one codebase. The multi-file form is a static bundle — an HTML entry point, JavaScript modules and a data file — that can be hosted on any static web server with no backend. The single-file form, `FRIS-Standalone.html`, inlines the entire application and an embedded data snapshot into one self-contained document (on the order of three megabytes) that opens directly from the local filesystem (`file://`) in any modern browser and works with no network connection whatsoever. There is nothing to install, no runtime to provision, and no server to stand up. For the analyst, "deployment" is copying a file.

This is enabled by a deliberately austere technology choice: the system is **framework-free**. It uses no front-end framework, no bundler-imposed runtime, no content-delivery-network dependency and no third-party npm packages at runtime; the server build, which is not published with the static and single-file forms, uses only the host platform's standard library (in the reference implementation, the built-in HTTP, test and SQLite facilities of Node.js 22 LTS). The user interface is a vanilla ES-module single-page application, and all visualisations — ownership network graphs, transaction-flow diagrams, Sankey recovery flows, jurisdiction maps — are hand-built SVG rather than charting libraries. The absence of dependencies is not asceticism for its own sake; it is what makes a trustworthy single-file, offline artifact *possible*. A dependency graph of hundreds of transitive packages cannot be audited by one analyst, cannot be guaranteed free of network calls, and cannot be frozen into one inspectable file with confidence. A codebase with zero runtime dependencies can.

3.2 One codebase, one API contract, many hosts

A key architectural device lets the same analysis code run identically whether or not a server is present. In the served build, the UI talks to a small local HTTP API. In the offline build, a shim intercepts the browser's `fetch` so that any request to the internal API path is answered *in memory* from the embedded data snapshot and the same pure analysis engines, returning the same shapes the server would. The consequence is that the identical UI and engine code executes unchanged in three settings — against a local server, as a hosted static bundle, and as a wholly offline single file — with the offline case simply substituting an in-memory responder for the network. The project also reports packaging the same codebase as an installable desktop application and a mobile application, again without forking the analysis logic; like the server build, those packages are not published with the static and single-file forms. The analytical core is written once and is provably the same across every host, which matters for reproducibility: a result obtained in the offline file is computed by the same functions as a result obtained on the server.

3.3 Determinism primitives

Two small, shared, thoroughly-specified primitives together eliminate the usual sources of non-determinism from the engines that trace or distribute value and from those that need random-looking values.

Exact money arithmetic. In the engines that trace or distribute value — the peel chain, the insolvency waterfall and the asset-tracing engine — monetary and value quantities are not represented as IEEE-754 floating-point numbers, because most decimal fractions cannot be represented exactly in binary floating point and the resulting drift is unacceptable where totals must reconcile to the last unit. Instead, every amount is an *integer number of minor units* held in an arbitrary-precision integer (`BigInt`): fiat is held at two-decimal scale (e.g. CHF 1,234.56 becomes the integer 123456), crypto value at eight-decimal ("sats") scale, and rates and percentages at six-decimal scale. A documented rounding policy governs the few places rounding is unavoidable: ratio operations round half-away-from-zero, and pro-rata distribution uses the *largest-remainder* method so that the sum of the parts exactly equals the pool being distributed, with no rounding leakage and a deterministic index-order tie-break. Because the arithmetic is integer arithmetic, it is exact, associative in the ways the algorithms rely on, and identical on every platform. In these engines floating point touches money only where a decimal input or rate is converted to fixed point on entry, in percentages and wallet balances derived for display, and in the waterfall's illustrative present-value discount. The other engines hold amounts as ordinary JavaScript numbers, which are exact for whole numbers below 2^{53} (the transaction-monitoring totals, for example), and hold percentages and scores in floating point.

Seeded pseudo-randomness. Where the system needs "random-looking" values — to generate synthetic demonstration data, or to drive the generative peel-chain simulator — it never calls the ambient random-number generator or reads the wall clock. It uses a small, well-understood, self-contained pseudo-random generator (a `mulberry32` core seeded through an `xmur3` string hash) that is a pure function of an explicit seed. The determinism guarantee is stated plainly in the code: identical seed and identical call sequence yield identical output, with no reliance on the platform RNG, the current time, or any ambient state. A synthetic dataset or a simulated chain is therefore fully reproducible from its seed, and two analysts on two continents with the same seed obtain the same artifact.

The engines themselves are written as **pure functions over an input snapshot**: given the same snapshot they return the same output, with no mutation of shared state, no I/O and no network. Leaving aside the timestamps and clock-derived identifiers the application writes into its own records (the audit trail, generated reports and imported records), three hidden inputs remain in the reference implementation. The first is the current date, from which the regulatory-horizon module counts the days to each instrument's effective date and sorts instruments into upcoming and overdue, so those figures change from day to day, as does the urgency component of the operational-impact score given to newly imported instruments. The second is the host's locale: the market-abuse surveillance and asset-tracing engines write quantities into their text in the host's number format, so the same alert reads "9,000 shares" on one machine and "9.000 shares" on another, and sorted lists such as the entity directory follow the host's collation (under Lithuanian rules, for example, names beginning with Y sort among those beginning with I). The third is the host's time zone: the compliance-review engine reads an instrument's effective year in local time, so on a machine west of UTC an instrument that takes effect on 1 January is treated as taking effect the year before, and a control citing that year is not flagged as outdated; no comparison within the snapshot is affected, but text an analyst supplies can be. The ingestion parsers, by contrast, record a written-out date such as "10 March 2026" as the day written, whatever the host's time zone. Determinism in FRIS is thus not a property that has to be tested for case by case; it is a consequence of three structural rules — integer money where value is traced or distributed, seeded-only randomness, pure engines — applied throughout the engine layer, with the three

exceptions just noted.

3.4 Verification posture

The project reports a self-verification harness: unit tests over the engines, and a headless end-to-end harness that renders the application's views and exercises the critical analyst workflows against the same in-memory API used by the offline build, together with a check that the standalone file renders from `file://`. Neither the tests nor the harness is published with the static and single-file forms, so we report them as the project's own statements rather than as independent benchmarks, and we make no performance or accuracy claims from them; their relevance here is architectural, namely that a deterministic, dependency-free system *can* be exhaustively and repeatably self-tested precisely because it has no non-deterministic or networked components to make tests flaky.

---

4. The Deterministic Peel-Chain Method

4.1 Definition and invariant

We define a **deterministic peel chain** as a value-conserving, integer-exact, reproducible traversal of a layered value transfer. Let the value at the head of the chain be an exact integer V_0 in minor units. The chain proceeds through hops $h = 1, 2, \ldots$; at each hop the current value V is partitioned into three non-negative integer parts —

- a **peel** p_h, the amount siphoned off at this layer toward a cash-out or side destination;
- a **fee** f_h, the cost of the transfer; and
- a **forwarded residual** $V - p_h - f_h$, which becomes the value at the next hop —

subject to the exact conservation invariant that value is neither created nor destroyed:

$$V_0 \;==\; \text{residual_final} \;+\; \Sigma\, p_h \;+\; \Sigma\, f_h$$

This invariant is not assumed; it is *computed and checked* at the end of every trace, and because all quantities are `BigInt` integers, the equality is exact rather than approximate. The invariant is the method's correctness contract: a peel chain that does not conserve value to the unit is reported as a mismatch rather than presented as sound. This is the property that makes the output defensible — an auditor can re-run the trace and confirm the books balance exactly, on any machine, with no tolerance band.

4.2 The core algorithm

The generative form of the method (used to model a laundering peel chain for typology testing and detector validation) is specified below in pseudocode, faithful to the reference implementation. `bp` is the peel fraction expressed in integer basis points; all division is integer division; `toMinor` converts whole units to minor-unit integers once, on entry, and everything after that conversion is exact integer arithmetic. (In the reference implementation the conversion multiplies by 10^8 in floating point and rounds to the nearest integer: deterministic under IEEE-754, and exact for inputs of up to eight decimal places below 2^{25} (about 33.5 million) units, but not for every larger input, nor for inputs carrying more decimal places.)

```
function simulatePeelChain(config):
    cfg  ← normaliseConfig(config)          # clamp/validate all parameters
    rng  ← SeededRng(cfg.seed + ":peel")    # pure function of the seed only
    V        ← toMinor(cfg.initialValue)    # BigInt minor units
    fee      ← toMinor(cfg.feePerTx)
    minPeel  ← toMinor(cfg.minPeel)
```

```
bp       ← round(cfg.peelPercent * 10000)   # peel fraction in basis points
totalPeeled ← 0 ;  totalFees ← 0 ;  hops ← [] ;  t ← startTime(cfg)
mainAddr ← synthAddr(rng)               # seeded synthetic address

for h in 1 .. cfg.hops:
    if V ≤ fee + minPeel:                # estate exhausted -
        break                            #   STOP; never fabricate empty hops

    peel ← (V * bp) / 10000              # integer division
    if peel < minPeel:   peel ← minPeel
    if peel > V - fee:   peel ← V - fee  # never overspend the residual

    destType ← scheduledEvent(cfg, h)    # configured bridge/mixer hop: no draw
    if destType = none:                  # otherwise a seeded, reproducible draw
        destType ← rng.weighted(destinationDistribution(cfg, h))
    destAddr ← synthAddr(rng)            # seeded cash-out address
    interval ← seededInterval(rng, cfg)  # seeded time step
    t ← t + interval
    nextAddr ← synthAddr(rng)            # seeded next-hop address
    forwarded ← V - peel - fee

    totalPeeled ← totalPeeled + peel
    totalFees   ← totalFees   + fee
    hops.append({ hop:h, from:mainAddr, to:nextAddr,
                  forwarded, peel, fee, destType, destAddr, ts:t })

    V        ← forwarded                 # residual becomes next hop's value
    mainAddr ← nextAddr

residual ← V
conserved ← ( toMinor(cfg.initialValue) == residual + totalPeeled + totalFees )
summary   ← { residual, totalPeeled, totalFees, conserved }
alerts    ← summaryAlerts(summary, hops)    # rule-based, on the finished chain
return { hops, summary, alerts }
```

Two features deserve emphasis. First, the loop **terminates honestly**: when the residual can no longer cover a fee plus the minimum peel, the chain stops rather than emitting degenerate or fabricated hops to reach a target length — a small but characteristic instance of the provenance-first principle applied to computation. Second, the *only* stochastic elements — the destination type (except at a configured bridge or mixer hop, where it is fixed and no draw is made), the synthetic main-chain and cash-out addresses, and the time interval between hops — are drawn from the seeded generator, in the order shown, so the whole chain, including its "randomness", is reproducible from `(seed, config)` alone.

4.3 Three operating modes

The peel-chain machinery appears in three complementary modes, each deterministic.

Generative (simulation). As above, `simulatePeelChain` produces a synthetic peel chain from a seed and a configuration (initial value, hop count, peel percentage, fee, time cadence, and optional bridge/mixer events). This drives the *typology laboratory*, where an analyst composes a laundering scenario — placement, then layering via shell entities and a crypto peel chain, then integration — and observes which detectors fire and where the gaps are. Because the chain is seeded, a scenario is a *shareable, reproducible fixture*: a colleague loading the same seed sees the identical chain.

Analytic (detection). Separate, pure detection passes — the transaction-monitoring rules and the on-chain detection battery — consume an arbitrary set of transactions and wallets and raise explained alerts. The peel-chain rule flags a trace when at least a configured minimum number of distinct hops (eight by default) are recorded as its peel hops: it relies on transaction records that carry a pattern label and a trace identifier, as the synthetic dataset and the typology laboratory's chains do, and counts the distinct hop numbers in each trace (or, where none is recorded, its transactions) without checking that they are consecutive. It does not infer the peel topology from unlabelled transfers; separating the forwarded spine from the peeled side-outputs on amounts and timing alone is not implemented (§7). The simulator also raises its own summary alert once a generated chain reaches fifty hops. The peel-chain rule is complemented by related deterministic detectors — fan-in and fan-out above a degree threshold (with known service infrastructure such as exchanges deliberately excluded, since high in-degree at an exchange is expected and benign), abnormal transfer velocity within a time window, and convergence of several distinct upstream wallets onto one exchange address. Each alert carries a stable key — in the monitoring engine an identifier derived from the rule and the primary subject, in the on-chain detection battery the alert type and the wallet or transaction concerned — together with a severity and a plain-language reason. Nothing here is learned or probabilistic; each detector is a transparent rule over the data, so an analyst can state exactly why an alert exists and reproduce it.

Aggregative (tracing/visualisation). For presentation, raw hops are aggregated deterministically into an *entity-level* flow graph — source of funds, one or more grouped peel-relay nodes, the landmark "gates" (mixers and cross-chain bridges) that break provenance, and the exchange, merchant and deposit cash-out clusters — laid out left-to-right in tiers. This mirrors how established investigation tools present a trace from first deposit to final cash-out over an entity graph rather than as an unreadable hop-by-hop hairball, but the aggregation here is a deterministic function of the trace, and colour is never the sole channel (shape, icon and label are redundant), so the visualisation is reproducible and accessible.

4.4 The same abstraction in ownership and recovery

The peel chain is, abstractly, a *monotone traversal of a layered structure that reports gaps rather than imputing them*, and the same traversal discipline recurs elsewhere in the suite; in the two value-flow cases, the peel chain itself and the insolvency waterfall, it is combined with exact-integer, invariant-checked accounting.

Beneficial-ownership chains. Resolving ultimate beneficial ownership walks *up* a chain of holdings, "peeling" effective control through each layer by multiplying ownership percentages (illustratively: if A owns 60% of B and B owns 50% of C, A's effective interest in C is 30%). The ownership-chain walk is cycle-safe (a circular holding is reported, not followed forever) and terminates at a natural person, a flagged ultimate owner, or an entity with no further recorded owner — a trust or foundation, for example, which the reference implementation does not itself flag as a gap. Crucially, an *unknown* percentage anywhere on a chain propagates as unknown rather than being guessed — the effective percentage becomes null, and the structure is flagged *not fully attributable* (see §6). This is the ownership analogue of the peel chain's honest termination: where the arithmetic runs out of exact inputs, the method reports a gap. Ownership percentages are not a conserved quantity, however, so no conservation invariant applies, and in the reference implementation the effective percentage is computed in floating point and rounded to two decimal places at each layer: a very small effective interest can therefore be reported as zero (two successive holdings of 0.4% give an exact effective interest of 0.0016%, which the walk reports as zero).

Insolvency-recovery waterfall. Distributing an insolvency estate first satisfies creditors holding fixed charges or mortgages from their own collateral, any shortfall ranking with the ordinary unsecured claims; the free estate then cascades value *down* a strict priority ladder:

expenses, then preferential classes, then floating-charge holders, then ordinary unsecured creditors, then subordinated and equity tiers, with each tier satisfied in full before the next is reached. The default order is configurable and illustrative; it is not a statement of any jurisdiction's statutory ranking. Where a tier's pool is insufficient, the shortfall is shared pro-rata among that class's admitted claims using the largest-remainder method, so the class distribution reconciles to the exact unit with no leakage — the same exact-integer, value-conserving discipline as the peel chain, applied to a downward cascade instead of a forward chain. No headline figure is hard-coded; every output is derived from the asset register, the creditor register and an explicit assumption set, so an auditor can perturb an assumption and see the reconciled effect.

In each case the deterministic traversal that surfaces gaps rather than filling them is the reusable primitive, and in the peel chain and the waterfall it carries integer-exact, invariant-checked arithmetic; the domains differ, the discipline does not.

---

5. The Engine Framework

FRIS is organised as a framework of **23 engine modules** — self-contained ES modules under a single `engines/` directory — layered over a set of static reference-data modules. Twenty-one of the twenty-three are domain analytics; the remaining two are the shared determinism primitives of §3.3 (the exact-money library and the seeded PRNG), on which the value-tracing and simulation engines draw. The engines are pure over the data passed to them and carry no user-interface code, so the same engine that answers the offline build's in-memory API also answers the server's HTTP API. Below we group them by function; the four areas the introduction highlighted are treated first.

Beneficial-ownership and control. The graph engine provides the ownership/control algorithms over the entity network: shortest connection paths, k-hop neighbourhoods, direct owners, the transitive sets of upstream controllers and downstream holdings, circular-ownership detection, and the effective-percentage ownership-chain walk and trace-to-ultimate-beneficial-owner described in §4.4. A companion structure-summary routine derives, for a root entity, the chain depth, jurisdiction count, offshore proportion, nominee and dormancy signals, and — the provenance-critical output — whether beneficial ownership is *fully attributable*. A flight-risk engine composes these structural signals into a transparent, weighted 0–100 score in which every point is attributable to a named factor.

Sanctions and adverse-media screening. The screening engine takes a name, alias, entity or wallet address and scores it against a dated, source-linked subset of real public sanctions designations (drawn from the OFAC SDN, UN Security Council, EU consolidated and UK OFSI lists), blending a token-set coverage ratio with Jaro-Winkler and normalised edit-distance similarity, adding a per-token evidence floor so that two unrelated names sharing only characters cannot manufacture a match, and corroborating on date-of-birth and country where available. Every candidate carries an explainable "why it matched" breakdown and a suggested strength band; sorts are stable (score descending, then list severity, then identifier ascending). The adverse-media engine is a transparent rule-based classifier that scores a news-style item's text against per-category keyword signals and rolls results up to the linked entity. Both are deterministic and self-explaining, and both are explicit that they demonstrate matching mechanics and are not a system of record — the live official lists govern any operational decision.

Insolvency-recovery waterfall. The waterfall engine implements the priority-cascade distribution of §4.4 over an asset register, a creditor register and a configurable assumption set (recovery deltas, cost assumptions, discount rate, disputed-claim treatment, clawback probability, FX rates and a configurable priority order), on the exact-money primitives. A cross-suite asset-tracing "capstone" engine then *joins* the ownership graph, the on-chain wallet

balances and the insolvency estate into one "where is the recoverable value and who controls it" view, banding each traced asset as recoverable, contested or frozen from real jurisdiction offshore/secrecy signals and the sanctions overlay — and labelling the only newly-synthesised links (which wallet cluster a debtor controls; a per-asset recovery-likelihood) explicitly as illustrative.

The compliance knowledge base and advisor. A large, structured, framework-free knowledge base encodes jurisdictions, size bands, controller/processor roles, regulatory frameworks and their applicability triggers, the concrete obligations and artefacts each framework requires, topic routing for natural-language queries, and a triage rubric. Its factual content is a paraphrase of real, publicly-sourced law and standards (GDPR, UK GDPR/DPA 2018, the revised Swiss FADP, ISO/IEC 27001 and 27701, CCPA/CPRA and other US state laws, DORA, NIS2 and the EU AML package), each cited to an official source, with numeric thresholds and dates flagged for re-verification and anything illustrative marked as such. The advisor engine routes a business profile and free-text query against this base to answer three questions a non-lawyer reaches for first — what law and internal-policy artefact applies (grouped by jurisdiction), whether the matter must be escalated to legal or can follow a defined process, and the concrete next steps — with stable sort orders and a standing "not legal advice" disclaimer on every result.

The remaining engines round out the analyst's workflow, all in the same explainable idiom: transaction monitoring with typology-rule alerting and a suspicious-activity-report builder; market-abuse trade surveillance with detectors mirroring real MAR/Dodd-Frank/CEA typologies over a synthetic order book; KYC/onboarding, which *composes* screening, adverse-media and country/sector risk into one weighted customer-due-diligence rating with a sanctions hard-stop; third-party/vendor composite risk scoring; a country-and-sector risk index; a real-instrument regulatory-horizon feed with defensive multi-format ingestion parsers and an operational-impact score; an obligation-to-control GRC register; a litigation-and-enforcement tracker over real public-record actions; a first-pass compliance-review engine that matches instrument obligations to internal-policy controls by weighted phrase overlap; and a quarterly financial-report engine. The point of enumerating them is not breadth for its own sake but *architectural uniformity*: every engine is pure, deterministic (subject to the three exceptions in §3.3), self-explaining and provenance-labelled, and those that trace value or simulate share the same two primitives — which is what makes the whole suite auditable as one coherent system rather than as twenty-odd disparate tools.

---

6. Provenance and Reproducibility

6.1 A provenance-first data model

The organising rule of the data model is that **every reported value is labelled by its source, and real and synthetic data are never silently mixed**. The system draws a hard line between two categories:

- **Real, dated, source-linked reference data.** The regulatory-horizon register comprises real, publicly-sourced instruments (including GDPR, MiCA, DORA, the revised Swiss FADP, Geneva's LIPAD and FINMA circulars), each linking to its official source. The screening watchlist reproduces a *dated subset* of genuine public sanctions designations from the OFAC, UN, EU and UK OFSI consolidated lists, each entry carrying its sanctioning authority, programme, real listing date, country where public, a paraphrased official reason for listing, and a source URL. One of those designations, OFAC's of the Tornado Cash mixer, was lifted on 21 March 2025 [19]; its entry is kept as a delisted record, with the delisting date, on the adverse-media list rather than the sanctions list. The enforcement tracker records real public-record enforcement actions. Reproducing

a sourced, dated subset of a consolidated sanctions list is that list's intended use — governments publish them precisely so firms can screen against them — and the system states clearly that it is a dated snapshot, not a live feed, and not a system of record.

- **Synthetic, seeded demonstration data.** The private-actor inputs — the corporate ownership network, the on-chain wallets and transactions, the insolvency estates and creditors — are entirely synthetic, generated deterministically from a fixed seed, and labelled as such. Nothing fabricated is attached to a real named party, with one class of exception: a few screening fixtures — an onboarding applicant, a vendor, and a trade counterparty and a cash-out wallet in the typology laboratory — carry a name taken from the real sanctions subset or, in the wallet's case, the address of the delisted Tornado Cash record, so that the screening path can be demonstrated; the name or address is real and everything attached to it is invented. The engines are real and deterministic; only the private-party inputs are fictional.

This labelling is pervasive: across the engine layer, provenance and honesty markers — disclaimers, "illustrative" tags, "synthetic" and "dated" annotations, and explicit source references — appear many hundreds of times, and every engine that touches real-world facts carries a standing disclaimer. The discipline is that a reader can always tell, for any value on screen, whether it is a real sourced fact, a paraphrase to be re-verified, or a synthetic demonstration figure.

6.2 Gaps are shown, never invented

The provenance-first stance extends from *where data came from* to *what to do when there is none*. The clearest instance is beneficial ownership. When the ownership-chain walk encounters a link with an unknown percentage, that unknown propagates — the effective percentage of the chain becomes null rather than a plausible guess — and the structure-summary routine sets an `incomplete` flag whenever any traced chain has an unknown cumulative percentage, a nominee is interposed (masking the real owner), or a circular holding breaks clean attribution. The system then reports the ownership as *not fully attributable*, and the flight-risk score treats incompleteness as a genuine, derived risk signal rather than papering over it. The same instinct governs the peel-chain simulator's honest termination (§4.2) and the asset-tracing engine's explicit labelling of its synthesised links as illustrative. Filling a gap with a confident-looking number is, in this design, a *bug*; surfacing the gap is the correct behaviour.

6.3 Determinism as reproducibility, and its evidentiary payoff

Because the engines are pure functions over an input snapshot, computed on exact-integer money where value is traced or distributed and with seeded-only randomness, **reproducibility is total** save for the three exceptions noted in §3.3: the same snapshot yields the same output, byte for byte, on any machine, offline. This is a stronger guarantee than "documented and repeatable"; it is *bit-identical*. The evidentiary payoff is direct. Where the *Daubert* line of authority asks for a tested method with a known error rate that others can reproduce [4, 9, 10, 11], a deterministic computation offers the strongest possible answer on the reproducibility axis: there is no method-induced error to estimate, because re-execution is exact. All genuine uncertainty is thereby relocated to the *inputs* — the assumed chain, the ownership data, the assumption set — where it is visible, labelled, and open to challenge, rather than being diffused into an opaque model. An analyst can hand a counterparty the same file and the same seed and invite them to reproduce every figure; disagreement can then be about the inputs, which is where, for defensible financial-crime and compliance work, disagreement should live.

6.4 Auditability of the artifact itself

Finally, the offline single-file, dependency-free architecture makes the *tool* auditable, not just its outputs. There is no server-side logic a reviewer cannot see, no third-party runtime package whose behaviour must be taken on trust, and no network traffic to monitor because there is none. The complete behaviour of the system is contained in one inspectable file that runs on the analyst's machine. For a domain in which one must sometimes defend not only *what* a tool concluded but *how the tool works*, an artifact a competent reviewer can read end to end is a materially stronger position than a cloud service accessed through an API.

---

7. Limitations and Future Work

We state the limitations plainly, in keeping with the paper's posture.

Determinism does not confer real-world correctness. The method guarantees that value flow is accounted for exactly and reproducibly *given* a chain; it does not establish that the chain corresponds to real-world control, nor does it attribute an address to a person. That attribution problem — the province of probabilistic clustering and off-chain intelligence — is out of scope and, by our argument, complementary rather than substitutable. A deterministic peel-chain audit is only as sound as the chain proposed to it.

Synthetic private data limits demonstrated, not designed, capability. In the reference implementation the private-party inputs are synthetic by design, for confidentiality and honesty. This demonstrates the *mechanics* faithfully but does not exercise the engines against the messiness of real-world records (inconsistent identifiers, adversarial obfuscation, dirty data). Applying the engines to real data is an operational deployment question, not an architectural one, but it is untested here and should not be overclaimed.

Explainability is traded against detection breadth. Transparent, rule-based detectors are legible and reproducible but will not match the recall of a well-trained learned model on novel or adversarially-evolving typologies. FRIS deliberately chooses legibility for the evidentiary use case; a hybrid in which learned models *propose* and deterministic engines *audit and explain* is attractive future work, provided the provenance boundary between the two is kept crisp.

Peel chains are detected from labels, not inferred. The detection rule counts hops that the transaction records already label as peel hops (§4.3). Inferring the topology from unlabelled transfers — separating the forwarded spine from the peeled side-outputs on amounts and timing alone — is not implemented, so on unlabelled data the suite will not find a peel chain by itself.

Reference data is a dated snapshot, not a live feed. The real sanctions and regulatory data are captured, dated subsets, appropriate for demonstrating matching and triage mechanics but explicitly not a compliance system of record. An offline tool is, by nature, in tension with the need to screen against *current* lists. Reconciling the two — for example, a signed, dated, verifiable data-pack that can be dropped into the offline file and whose provenance and as-of date are cryptographically attested — is a natural and important extension.

Browser and single-file constraints. Delivering the whole system as one client-side file bounds the data volume it can embed and the compute it can perform relative to a server-backed pipeline, and ties it to the correctness of the host browser's engine. The exact-integer and seeded-PRNG choices mitigate cross-engine drift, but very large estates or chains would eventually strain an in-browser, single-file model.

Ownership arithmetic is rounded, not exact. The ownership-chain walk computes effective percentages in floating point, rounded to two decimal places at each layer, so a very small

effective interest can be reported as zero; and a chain that ends at a trust, foundation or other entity with no recorded owner is not flagged as a gap (§4.4). Moving the walk onto the shared fixed-point primitives, and treating such chain ends as not fully attributable, would close both gaps.

Formalisation. The value-conservation invariant and the largest-remainder reconciliation are specified and checked in code and prose here; a machine-checked formal specification of the peel-chain and waterfall invariants, and of the ownership-chain "not fully attributable" semantics, would strengthen the evidentiary story and is planned.

Beyond addressing these, future work includes: signed, verifiable reference-data packs with attested as-of dates; an exportable, self-describing "audit bundle" that pairs a result with the exact inputs and seed needed to reproduce it; the learned-propose/deterministic-audit hybrid noted above; and extending the provenance-first "show the gap" discipline into a first-class uncertainty representation carried through every engine's output.

---

8. Conclusion

Mainstream financial-crime and compliance tooling optimises for scale and detection breadth, and pays for them in confidentiality (data leaves the analyst's machine), reproducibility (outputs are probabilistic), and explainability (results are scores, not derivations). For a specific but important class of work — evidentiary analysis of sensitive data by individual analysts who may have to defend their findings — those payments are too high. This paper has described an architecture that refuses them. By running entirely offline as a single, dependency-free, fully-inspectable file, it keeps subject data on the analyst's machine and makes the tool itself auditable. By tracing layered value transfers as *deterministic peel chains* — exact-integer, value-conserving, invariant-checked, byte-for-byte reproducible — it produces findings an auditor can re-derive rather than merely accept, and it applies the same traversal discipline to beneficial-ownership resolution and insolvency recovery. And by being provenance-first — labelling every value by source, never mixing real and synthetic data silently, and surfacing gaps instead of imputing them — it locates all genuine uncertainty in the inputs, where it belongs and can be challenged. None of these commitments is fashionable, and each trades away some breadth. The claim of this paper is narrower and, we hope, more durable: that for reproducible, confidential, explainable compliance analysis, determinism, offline operation and explicit provenance are the correct defaults, and that a small, framework-free, deterministic codebase is not a step backward from cloud-scale machine learning but a different and defensible point in the design space.

References

[1] Reiter, J. *Hazy Transparency: Blockchain Forensics, the Co-Spend Heuristic, and the Legal Limits of Crypto Tracing.* Blockhead, 27 February 2026. https://www.blockhead.co/2026/02/27/hazy-transparency-blockchain-forensics-the-co-spend-heuristic-and-the-legal-limits-of-crypto-tracing/ (accessed 16 October 2026).

[2] Finanz-Forensik. *Crypto Forensics in Practice: Methods and Limitations.* https://finanz-forensik.de/en/crypto-forensic-methods/

[3] Sammis Law Firm, P.A. *Problems with Heuristics in Cryptocurrency Tracing.* Last updated 8 June 2026. https://criminaldefenseattorneytampa.com/asset-seizure-asset-forfeiture/cryptocurrency/tracing/heuristics/ (accessed 16 October 2026).

[4] Chainalysis. *What Is Blockchain Forensics? Definition, Process & Daubert Standard.* https://www.chainalysis.com/glossary/blockchain-forensics/ (accessed 16 October 2026).

[5] Unit21. *The Complete Guide to RegTech (Regulatory Technology).* 31 May 2023. https://www.unit21.ai/blog/regtech (accessed 16 October 2026).

[6] FinregE. *What is SupTech?* 14 April 2023. https://finreg-e.com/what-is-suptech/ (accessed 16 October 2026).

[7] Turki, M., Hamdan, A., Cummings, R. T., Sarea, A., Karolak, M., & Anasweh, M. *The Regulatory Technology "RegTech" and Money Laundering Prevention in Islamic and Conventional Banking Industry.* Heliyon 6(10), e04949 (2020). PMC7550909. https://doi.org/10.1016/j.heliyon.2020.e04949

[8] El Harras, M., & Salahddine, M. A. *Tracking Financial Crime Through Code and Law: A Review of RegTech Applications in Anti-Money Laundering and Terrorism Financing.* arXiv:2511.15764 (2025). https://arxiv.org/abs/2511.15764

[9] Stoykova, R., & Franke, K. *Reliability Validation Enabling Framework (RVEF) for Digital Forensics in Criminal Investigations.* Forensic Science International: Digital Investigation 45, 301554 (2023). https://doi.org/10.1016/j.fsidi.2023.301554

[10] *Daubert v. Merrell Dow Pharmaceuticals, Inc.*, 509 U.S. 579 (1993). See also: Maryman & Associates, *Daubert Digital Forensics Explained for Legal Professionals.* https://maryman.com/daubert-digital-forensics-explained-for-legal-professionals/

[11] Chaudhary, R., Ryan, R., Ferdosian, N., Karie, N. M., & Li, Q. *A Deterministic Forensic Preprocessing Framework for Heterogeneous Network Datasets: Formal Foundations, Implementation, and Empirical Validation.* arXiv:2606.11565 (2026). https://arxiv.org/abs/2606.11565

[12] Anders. *Understanding Air-Gapped Reporting & Analytics: Secure On-Premises Data Insights Without Cloud Access.* ONES blog, 28 February 2026 (page modified 20 August 2026). https://ones.com/blog/air-gapped-reporting-analytics-on-premises-solutions-explained/ (accessed 16 October 2026).

[13] Robert X. *The Rise of Air-Gapped AI and Privacy-First Innovation.* AICompetence, 16 October 2025. https://aicompetence.org/air-gapped-ai-and-privacy-first-innovation/ (accessed 16 October 2026).

[14] Kleppmann, M., Wiggins, A., van Hardenberg, P., McGranaghan, M. *Local-First Software: You Own Your Data, in Spite of the Cloud.* Onward! 2019 (ACM SIGPLAN), Athens, pp. 154–178. https://doi.org/10.1145/3359591.3359737

[15] Financial Action Task Force (FATF). *International Standards on Combating Money Laundering and the Financing of Terrorism & Proliferation: The FATF Recommendations* (2012, updated June 2026), Recommendation 24 and General Glossary, "Beneficial owner"; Recommendation 24 and its Interpretive Note were revised on 4 March 2022. https://www.fatf-gafi.org/content/dam/fatf-gafi/recommendations/fatf-recommendations-2012.pdf; glossary also at https://www.fatf-gafi.org/en/pages/fatf-glossary.html (both accessed 16 October 2026).

[16] Financial Action Task Force (FATF). *Guidance on Beneficial Ownership of Legal Persons* (10 March 2023), Box 2. https://www.fatf-gafi.org/en/publications/Fatfrecommendations/Guidance-Beneficial-Ownership-Legal-Persons.html; PDF: https://www.fatf-gafi.org/content/dam/fatf-gafi/guidance/Guidance-Beneficial-Ownership-Legal-Persons.pdf.coredownload.pdf (accessed 16 October 2026).

[17] Meiklejohn, S., Pomarole, M., Jordan, G., Levchenko, K., McCoy, D., Voelker, G. M., Savage, S. *A Fistful of Bitcoins: Characterizing Payments Among Men with No Names.* ACM

Internet Measurement Conference (IMC '13), Barcelona, 2013, pp. 127–140. https://doi.org/10.1145/2504730.2504747

[18] ChainAware.ai. *Forensic vs AI-Powered Blockchain Analysis: Why Predictive Intelligence Wins 2026.* 4 January 2026 (page modified 24 May 2026). https://chainaware.ai/blog/forensic-crypto-analytics-versus-ai-based-crypto-analytics/ (accessed 16 October 2026).

[19] U.S. Department of the Treasury. *Tornado Cash Delisting.* Press release, 21 March 2025. https://home.treasury.gov/news/press-releases/sb0057; Office of Foreign Assets Control, *Cyber-related Designation Removal; North Korea Designation Update and Removal*, recent action of 21 March 2025. https://ofac.treasury.gov/recent-actions/20250321 (both accessed 17 October 2026).

This is a preprint describing a working reference system; it has not yet been peer reviewed. Private-party data referenced herein is synthetic and labelled; regulatory, sanctions and enforcement reference data are real, dated and source-linked as described in §6. The system is expressly not a system of record and does not constitute legal advice.

Further reading and points of disagreement (July 2026 addendum)

The probabilistic baseline this architecture deliberately departs from has a canon, and honesty requires citing it: Fellegi and Sunter's theory of record linkage is the mathematically optimal treatment of matching under uncertainty, and nothing in this paper disputes its optimality *for coverage*. The disagreement is about the objective function. Probabilistic linkage maximises expected match quality; the peel-chain method maximises defensibility — byte-for-byte reproducibility under an explicit conservation invariant — and accepts reduced coverage as the price, surfacing every unresolved gap rather than imputing across it. In evidentiary settings that trade is the right one, and the reproducible-computational-research norms articulated by Sandve and colleagues state the general case for it. Where coverage matters more than defensibility, the correct tool is the probabilistic canon, used with its assumptions declared.

- Fellegi, I. P. & Sunter, A. B., "A theory for record linkage" (1969) 64 *Journal of the American Statistical Association* 1183–1210. https://doi.org/10.1080/01621459.1969.10501049.
- Sandve, G. K., Nekrutenko, A., Taylor, J. & Hovig, E., "Ten simple rules for reproducible computational research" (2013) 9 *PLOS Computational Biology* e1003285. https://doi.org/10.1371/journal.pcbi.1003285.