Blockchain for Royalty Distribution

Abstract

Royalty distribution in music is a payments problem only on the surface. Underneath, it is a data and authority problem: rights are split into distinct bundles; ownership and administration change over time; usage happens at extreme volume across platforms that do not report in the same way; and money moves through long chains of specialists who each keep their own records. The result is familiar across the industry: payments arrive late, splits are disputed, and large amounts of usage cannot be matched to a payable owner when it first enters the system.

This paper examines whether blockchain systems can improve royalty distribution without pretending they can replace law, contracts, or existing collecting institutions. Chapter 1 maps the current landscape in 2026: how money flows across streaming, UGC, short-form video, and emerging AI-

generated content; where metadata fails; how cross-border routing increases delay; and why intermediaries remain necessary even as they add extra steps and reduce end-to-end visibility. Chapter 2 specifies a blockchain model in concrete terms: a shared, append-only rights registry; tokenized representations of economic interests and administration roles; split logic encoded in contract templates; report commitments and proof mechanisms for usage inputs; identity binding and compliance controls; and privacy designs based on permissioned networks, selective disclosure, and zero-knowledge proofs. Chapter 3 moves from architecture to operations: governance, onboarding, dispute isolation, legal enforceability, oracle risk, throughput limits, energy and cost considerations, and phased integration with existing royalty systems.

Royalties Are Not A Payments Problem

The paper's core claim is restrained: blockchain can help when it is used as a shared evidence layer and a settlement coordinator, not as a fantasy replacement for institutions. The decisive questions are not "public chain or private chain" and not "tokens or no tokens," but who can write rights state, how usage data is committed and checked, how disputes are contained, how identities are bound to payout endpoints, and how the system fits statutory and contractual reality across territories. The value of the model lives in a cleaner audit trail, fewer reconciliation loops, and more frequent distribution once funds enter the system—while keeping sensitive data off public ledgers.

Keywords

Blockchain; royalty distribution; music rights; metadata; ISRC; ISWC; DDEX; PRO; CMO; collective management; smart contracts; split contracts; recoupment; advances; audit trail; append-only ledger; oracles; report commitments; Merkle proofs; zero-knowledge proofs; permissioned networks; decentralized identity; verifiable credentials; KYC; AML; Travel Rule; stablecoins; micropayments; settlement;

cross-border royalties; UGC; short-form video; generative AI; digital replicas; fraud; spam; governance; dispute resolution; escrow.

Introduction

1. Problem statement: royalties fail for reasons that have nothing to do with payments rails

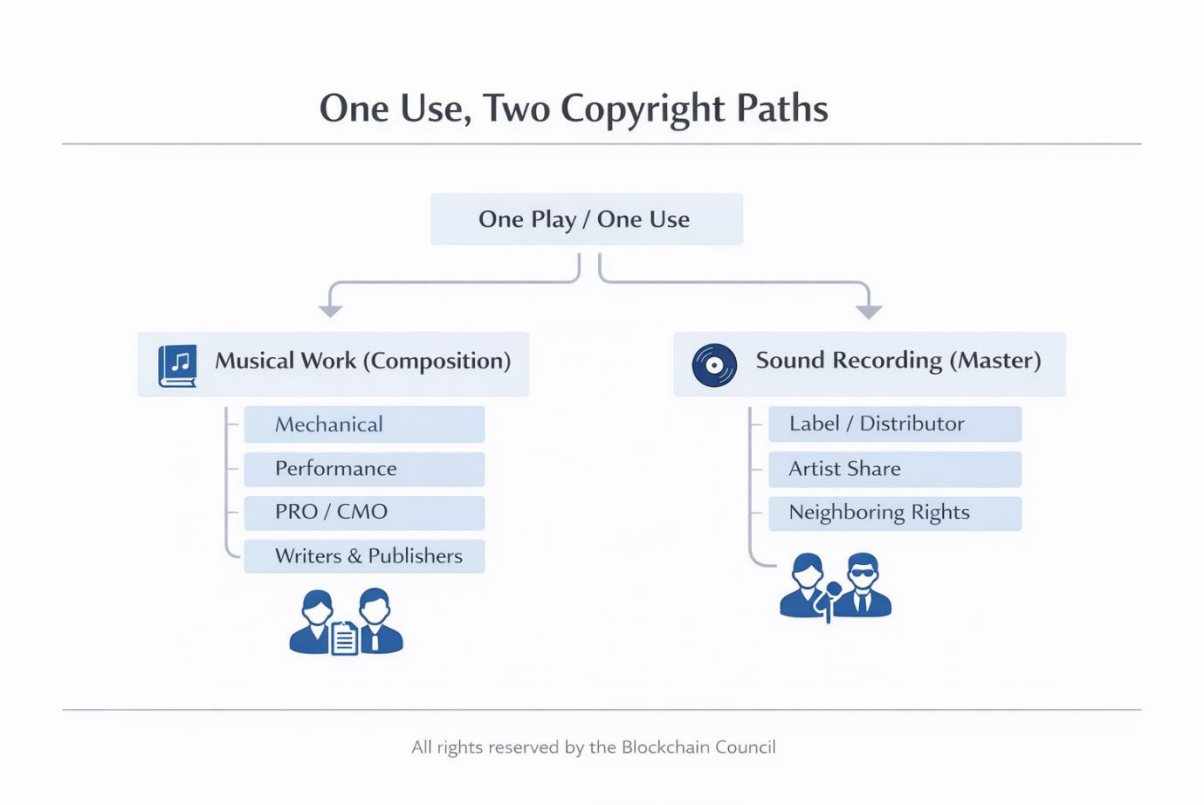
Royalty distribution sounds like a money problem. In practice, it is a question of evidence: who owns what, who is allowed to administer it, what happened on a platform, what a license permits, and which rule set applies in which territory and time period. Money can move quickly once these questions are answered. The real delays come from the system's inability to answer them at scale with a record that multiple parties accept.

The current system contains many strong parts. It can collect large sums from global platforms. It can send funds to millions of recipients. It can manage statutory licenses in some territories and private licenses in others. It can correct past errors, sometimes years later, as better ownership data

arrives. Yet those strengths sit on top of fragmented databases and institutional boundaries that are difficult to reconcile.

A creator's experience often exposes the gap between "the industry paid out billions" and "my usage produced money I cannot see or claim." A platform can show a dashboard of plays while the publishing side lags because the recording-to-work link is missing. A society can report distributions while a foreign share takes longer because it must pass through reciprocal routing. A distributor can pay the master share each month while the writer share sits in a pending bucket because splits were never finalized.

This paper treats blockchain as a set of tools for record-keeping and multi-party coordination. It does not treat blockchain as a substitute for the legal world. Courts, contracts, and statutory mandates still decide what rights exist and what remedies apply. Any serious system must accept that constraint up front.



2. What this paper does, and what it refuses to do

This is not a marketing case for "rights on chain." It is a practical design and operations analysis.

The paper does four things:

1. **It describes the 2026 baseline.** How royalties flow today, where metadata fails, and why new platform types make attribution harder.
2. **It specifies a blockchain model with clear boundaries.** It separates registry state, rights logic, and

payments/compliance. It defines what goes on-chain, what stays off-chain, and what must be proven.

3. **It turns "blockchain for royalties" into a set of testable questions.** Who can write state? How is usage introduced? What is the dispute process? How do upgrades work? How is identity handled? What are the costs?
4. **It presents an implementation playbook and a risk register.** It focuses on governance, onboarding, disputes, enforceability, scaling, oracle risk, cost, and integration.

It also refuses to do several things.

- It does not promise that a new technical layer makes ownership disputes disappear.
- It does not pretend that public disclosure of deal terms is acceptable at industry scale.
- It does not assume that platforms will move their revenue collection onto a chain.

- It does not claim that tokenizing royalty cash flows is legally trivial.

The paper's stance is blunt: blockchain helps only if it fits the real machinery of licensing, reporting, and enforcement.

3. Definitions: what "royalties" means in this paper

In music, "royalties" is a catch-all term. In this paper it refers to payments tied to exploitation of two main right bundles:

- **The musical work (composition):** rights held by writers and publishers.
- **The sound recording (master):** rights held by artists, labels, and recording owners.

The paper covers common royalty categories, including mechanical and performance on the composition side; sound recording public performance where applicable; negotiated licensing such as sync; and platform-specific schemes for UGC and short-form video.

It also includes newer categories that change distribution logic even when they do not map cleanly to legacy labels:

- pooled payouts for short-form video
- policy-driven monetization on UGC platforms
- platform rules designed to curb fraud and spam

The paper treats "royalty distribution" as the full pipeline: matching usage to assets, mapping assets to rights and shares, applying license rules and deal rules, then delivering payments and statements.

4. Why the problem is sharper in 2026 than it looked in 2016

A decade ago, many teams framed the problem as "streaming data is messy." That is still true. But the main shift is that the number of rights events has grown faster than the industry's ability to assign ownership facts.

Three forces make today's problem harder.

4.1 UGC turns uploads into licensing decisions

When users upload video, audio, edits, and mashups, the platform becomes a rights engine. It must identify copyrighted material, choose which policy applies, and route money

accordingly. That logic sits on top of imperfect matching and frequent disputes.

UGC also changes incentives. Platforms want systems that avoid false claims at scale. Rights holders want systems that catch more uses and pay them. Users want systems that do not wrongly block their content. These goals collide.

4.2 Short-form video adds pooled economics

Short-form platforms often allocate money through pools rather than paying per asset in a way that resembles classic streaming statements. Pools can lower reporting overhead, but they make tracing a specific use to a specific payment line much harder.

Pools also make attribution more political. Small rule changes can shift money across large catalogs. That raises the stakes of audit and governance.

4.3 Generative AI raises authorship, identity, and spam pressure

Generative AI increases content volume and increases uncertainty about authorship and provenance. It also creates

new abuse patterns: voice imitation, fake artists, and mass-upload spam designed to skim from payout pools.

Even if law and policy settle over time, distribution systems must function while uncertainty remains. That means systems must support dispute isolation, identity checks, and fraud controls as first-class features.

5. The blockchain claim, restated plainly

Teams pitch blockchain for royalties with a familiar set of promises: cleaner data, fewer intermediaries, faster payments, better audit, fewer disputes. Those promises are often framed as if they come as a single bundle.

This paper breaks the claim into smaller, testable propositions:

- A shared rights registry reduces duplicate data entry and makes ownership changes easier to check.
- A shared audit trail makes it harder to change history quietly.
- Standard contract templates encoded in code reduce disputes about arithmetic and timing.

- Commitments and proofs for usage reports reduce disputes about whether reporting changed after the fact.
- Identity-bound payout endpoints reduce misdirected payments and simplify compliance.
- Batching and settlement on low-cost execution layers can shorten payout cycles once funds enter the system.

Each proposition comes with costs and tradeoffs. A shared registry requires governance. Proof systems require compute and operational discipline. Identity binding requires KYC workflows. Faster payouts require funding and settlement design that fits the way platforms actually collect money.

The goal is to judge each proposition on its own.

6. The core design constraint: global rights are not a single truth

Many “global rights graph” projects fail because they assume a single canonical truth. In reality, rights are often territorially partitioned and administratively delegated. A publisher may control certain territories through a sub-publisher. A society may have a mandate that changes what is collectable in that

region. A label may hold the master rights globally while an artist has different participation rules by market.

A usable system must represent “locally valid truth” without collapsing it into contradiction. That is a data model challenge and a governance challenge. It also shapes how disputes are handled, because many disputes are not about whether a work exists but about which local agreement governs a local slice of the world.

7. Method and scope

This paper is built as an applied systems analysis rather than a purely academic survey.

- It maps the current royalty pipeline and identifies failure points.
- It defines a blockchain-based model in terms of registries, rights logic, usage commitments, settlement, identity, and privacy.
- It evaluates implementation requirements through governance, operations, and risk.

The scope is intentionally broad across platform types because the most difficult attribution problems now sit at the boundaries: streaming to short-form, UGC to licensing pools, human-made tracks to AI-made tracks.

The scope is also intentionally conservative about legal claims. The paper assumes that legal enforceability must be anchored in human-readable agreements that reference on-chain state, and that statutory systems and societies will remain central in many contexts.

8. Evaluation framework: how to judge a blockchain royalty design

A blockchain system can be judged by whether it reduces mismatches and delays without creating new risks that are worse than the old ones. This paper proposes an evaluation frame organized around eight questions.

8.1 Authority: who can write rights state, and why should anyone accept it?

A registry is useful only if participants accept that “active state” is grounded in recognized authority. That requires clear roles, credentialing, and co-signing rules.

8.2 Evidence: what does the system record that was previously hard to prove?

A ledger is only as useful as the trail it preserves. The key evidence types in royalty systems are ownership changes, administrator assignments, license term versions, usage report commitments, funding events, disputes, and dispute outcomes.

8.3 Disputes: can the system contain conflict without blocking everyone?

Disputes must be isolated to affected shares. Funds must be escrowed for the disputed portion while undisputed shares continue. The system must support retroactive adjustments after resolution.

8.4 Identity and compliance: can the system pay legal persons safely?

Payout endpoints must be bound to verified identities. The system must support tax handling, sanctions screening, and AML obligations through clear role assignment.

8.5 Privacy: can the system provide audit and still protect sensitive data?

Deal terms, personal data, and granular usage logs cannot be published to public ledgers. A viable system must use restricted data sharing, commitments, and selective disclosure.

8.6 Throughput and cost: can the system handle volume without turning fees into the largest line item?

Per-play transactions are not realistic at mainstream scale. The system must batch, commit, and settle at intervals.

8.7 Interop: can the system fit existing standards and institutions?

Rights data and reporting already flow through standard formats and mandated institutions. A blockchain layer that cannot ingest and output those formats will remain a side project.

8.8 Operations: can the system be run safely for years?

This includes security, upgrades, monitoring, incident response, governance change processes, and user support, including key recovery.

9. Contributions: what a reader should get from this paper

A reader should leave with four concrete outputs.

1. A plain map of how royalties flow today and why the hard parts persist.
2. A reference architecture for blockchain-based rights state and settlement that can be argued about in detail.
3. A checklist of production requirements that go beyond "deploy a contract."
4. A risk register that treats governance and oracles as first-order problems.

If a reader wants to evaluate a proposed "blockchain for royalties" project, the paper offers a way to interrogate it:

- What is tokenized, if anything?
- Where is the authoritative registry state?

- Who can update it?
- How are usage reports introduced and checked?
- How are disputes handled without global freezes?
- Where is identity verified and enforced?
- Which data is public, which is restricted, and how are proofs handled?
- How does the system integrate with PROs, societies, publishers, distributors, and DSPs?

10. Reading guide: how the chapters fit together

- **Chapter 1** describes the current landscape in 2026 and the "pain map" any new system must face.
- **Chapter 2** specifies a blockchain model and explains the design choices hidden inside the phrase "blockchain royalties."
- **Chapter 3** focuses on production deployment and risk, with emphasis on governance, onboarding, disputes, enforceability, scaling, and integration.

Chapter 1 (2026 baseline): How royalties move now

Streaming still prints record revenue. Attribution and payment still fail in ways that leave money late, stuck, or hard to audit.

1.0 Why royalty distribution is still hard in 2026

Royalty distribution isn't one problem. It's a pile of problems that interact:

- Law and contracts split music into separate rights with separate pay rules.
- Usage happens at huge scale across many platform types.
- Reports arrive late, in mismatched formats, and with shifting time windows.
- Ownership data is split across databases, disputed, and tied to territory.
- Money travels through specialists, each with its own incentives, fees, and risk rules.

"Blockchain for royalties" is usually shorthand for: make the system easier to verify, easier to connect, and faster. You can't judge that pitch without a clear picture of today's plumbing.

By early 2026, streaming remains the center. Global recorded music revenue reached US$29.6B in 2024; streaming was 69.0% of that total and passed US$20B for the first time. Subscription streaming alone was over half of recorded music revenue. [IFPI]

On the composition side, creator collections are also at record levels. CISAC reports global creator royalties of €13.97B in 2024, with digital revenue above €5B for the first time and up 11.2%. [CISAC]

Meanwhile, the environment is louder and messier: short-form video creates new license shapes; UGC makes every upload a rights event; and generative AI drives both volume and author confusion. Topline numbers rise, but a lot of value stays delayed, disputed, unmatched, or paid with little end-to-end visibility.

1.1 The rights map: what is being monetized

You can’t explain “how royalties flow” without naming the rights.

1.1.1 Two core right bundles

Most music uses involve two separate copyrights (or close equivalents):

- **The musical work (composition):** the song as a work—melody, harmony, lyrics. Usually owned by songwriters and publishers (or writers who publish themselves).
- **The sound recording (master):** a specific recorded performance. Usually owned by a label, distributor, or the artist if they own their masters.

One stream can generate money for both. The payment routes split almost immediately.

1.1.2 Main royalty categories

Across jurisdictions, the big buckets look like this:

- **Mechanical royalties (composition):** tied to reproductions and certain digital uses; in interactive streaming, handled as streaming mechanicals (rules vary by territory).

- **Performance royalties (composition):** public performance via radio, venues, broadcast, and many digital uses; typically collected by PROs/CMOs.
- **Sound recording public performance and neighboring rights:** common outside the US; collected via neighboring rights systems. In the US, sound recording public performance is narrower, but statutory digital performance applies to certain non-interactive services.
- **Synchronization and related licensing:** film, TV, ads, games; negotiated deals, often with advances and complex backend splits.
- **UGC and short-form schemes:** platform deals, pools, and revenue shares that don't map neatly onto older labels.

Each bucket has its own data needs. Mechanical payments need work ownership and shares. Recording payments need master ownership and artist participation data. UGC often needs matching plus a rights-holder policy choice at scale (monetize, block, track).

1.2 How royalties flow today: the practical money routes

This is the supply-chain view: where money starts, what data is needed to route it, and where delays show up.

1.2.1 The streaming-to-rightsholder pipeline

Most major interactive DSPs follow the same pattern:

1. **Revenue:** subscriptions first, ads second. [Reuters: IFPI coverage]

2. **Allocation:** the platform sets aside the contract-and-law share for rightsholders, then allocates it by usage share (usually pro rata).
3. **Split by right bundle:** money routes to recording rightsholders (labels, distributors, independent owners) and publishing rightsholders (publishers, PROs/CMOs, mechanical admins).
4. **Downstream splits:** labels pay artists per recording contracts; publishers pay writers per publishing terms.
5. **Reporting, matching, adjustments:** reports arrive on schedules; matching depends on identifiers and ownership databases; late reports and corrections create long-tail updates.

Even in modern systems, matching fails at scale. The US mechanical system is a clear example.

1.2.2 US streaming mechanicals: the MMA and The MLC

Under the Music Modernization Act, interactive streaming mechanicals are handled through The MLC's blanket license. The MLC receives money and reports from digital music

providers, matches usage to works and rightsholders, distributes, and then reprocesses unmatched usage as the database improves.

For 2024 usage, The MLC processed $1.01B and directly distributed $771.1M in 2024 blanket royalties. The reported average initial match rate was 84.3%, increasing to 89.1% after reprocessing (as of March 2025). The MLC also reports total distributions of $813.6M for 2024 usage including voluntary-license-covered value. [The MLC]

The backlog problem still exists. A US Copyright Office meeting summary (Nov 2025) states The MLC matched nearly $317M of roughly $397M transferred in Feb 2021 (79%) and distributed about $228.36M (more than 57.5%). [USCO meeting summary]

These figures matter for any “new infrastructure” idea:

- Better data can raise match rates over time.
- Money can sit for years when ownership and links are unclear.

1.2.3 US statutory digital performance: SoundExchange

For eligible non-interactive services in the US, statutory sound recording digital performance royalties are collected and distributed by SoundExchange. SoundExchange describes the statutory split: 45% to featured artists, 5% to a fund for non-featured artists, and 50% to the sound recording rights owner. [SoundExchange]

This shows what happens when law defines both the collector and the split: fewer negotiation layers, but the same dependence on accurate performer and recording data, plus long-tail claims and disputes.

1.2.4 Collective management and cross-border routing

CMOs and collecting societies remain central in many territories. CISAC's 2024 collection total (€13.97B) shows the scale. [CISAC]

Cross-border routes add delay and make tracing harder:

- Usage happens locally.
- A local society collects and routes shares to foreign societies under reciprocal deals.

- Foreign societies distribute to local members, sometimes via sub-publishers or admins.

Every hop adds processing time, data transformation, fees, and more places for identifiers and splits to drift.

1.3 Where metadata breaks

Royalty systems run on metadata. The industry acts like it's solved. It isn't.

1.3.1 What "metadata" means for royalties

For royalty routing, metadata includes more than title and artist:

- Identifiers: ISWC, ISRC, UPC/GTIN, IPI, ISNI, and the links among them.
- Work ownership shares, including territory limits.
- Recording ownership, roles, and points (featured artist, session players, producer points).
- Deal terms: territories, windows, carve-outs, UGC permissions.
- Matching references: fingerprints and reference files.

Standards exist to move this data. DDEX maintains message standards for releases and resources (ERN), usage and sales reporting (DSR), and richer descriptive and party data (MEAD, PIE). RIN is designed to carry recording credits so parties can route payments and credit properly. [DDEX]

Standards don't guarantee adoption, quality, or aligned incentives.

1.3.2 The creation gap: splits and contributors aren't ready

Many failures start before distribution:

- Writer splits aren't finalized at release.
- Contributors (especially producers and session players) aren't listed.
- Names and identifiers don't match across systems.
- Cross-border writing teams add publisher control and society complexity.

Usage starts the moment a track goes live. If a work isn't registered with correct shares, composition-side money has no

clean path and sits in unmatched or pending pools—then gets pushed through reprocessing cycles, as The MLC's match stats show. [The MLC]

1.3.3 The mapping gap: recordings don't link cleanly to works

A stream points to a recording. Publishing payments need a recording-to-work link. That link fails often because:

- One recording can include multiple works (medleys, samples, interpolations).
- Covers create new recordings tied to existing works.
- Remixes and edits create new recordings with different master ownership and credit splits.
- Local versions, translations, and adaptations complicate "same work" logic across territories.

When the link fails, master-side payments often move sooner (ISRC and label supply chains tend to be tighter) while publishing-side money lags.

1.3.4 The message gap: data degrades as it moves

Even when data exists, it gets lost or warped:

- Aggregators may drop credits on the way to DSPs.
- DSPs may not store or surface full composition data in reports.
- Reports arrive in different schemas and need normalization.
- Ownership updates arrive after usage, forcing retroactive adjustments.

The UK's industry agreement on streaming metadata exists because the sector has treated metadata quality as a shared problem that needs coordinated work. [UK metadata agreement]

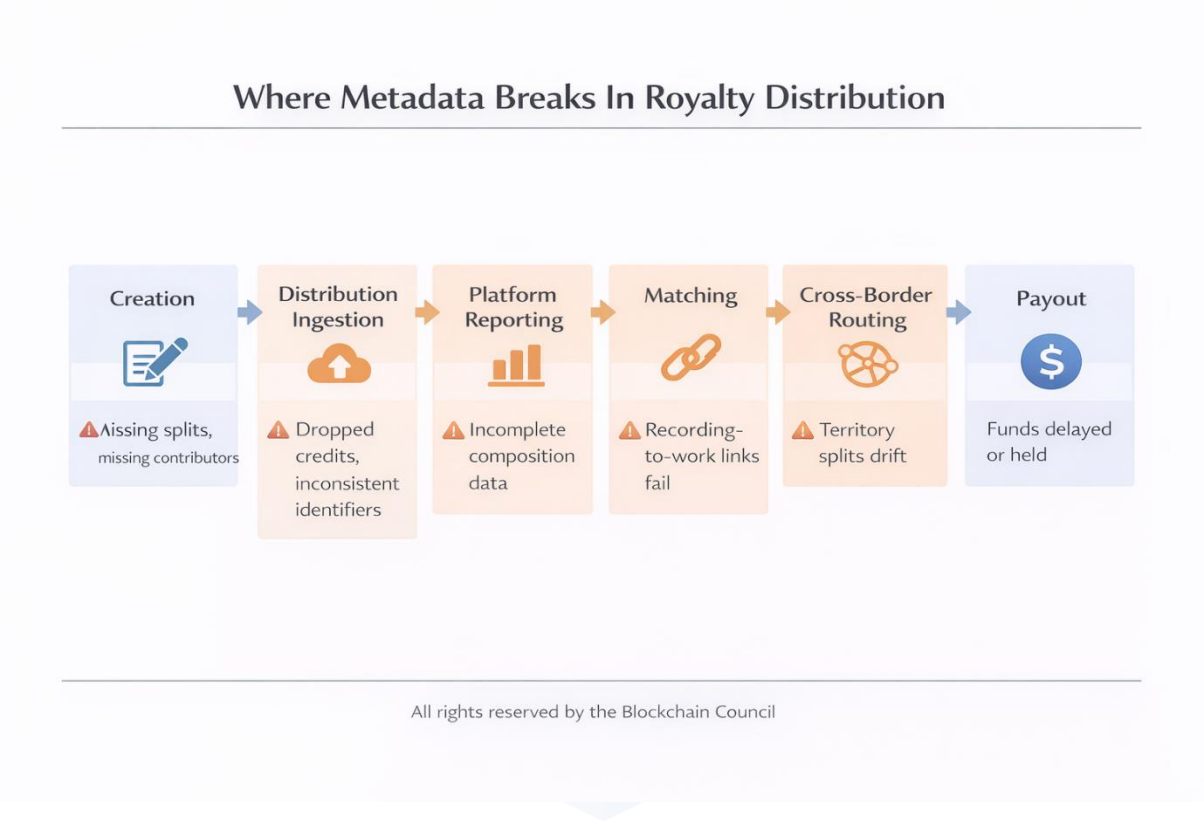
1.3.5 The territory gap: ownership and licensing are local

Rights are often split by territory:

- Publishers may control only certain regions.
- Sub-publishers administer locally.
- Society mandates differ.

- Statutes carve out or reshape rights.

So “global ownership” is often a set of locally valid truths. Any new infrastructure has to model that without pretending there is one universal record.



1.4 Why intermediaries add friction (and why they stay)

Intermediaries exist because the system needs specialization, aggregation, and risk buffering. They also create boundaries between databases, formats, incentives, and audit rights.

1.4.1 The intermediary map

Common roles include:

- Labels and distributors (recording ingestion, marketing, financing, accounting)
- Publishers and publishing admins (work registration, licensing, claims, collections)
- PROs/CMOs and collecting societies (performance and, in many places, mechanical collections)
- Neighboring rights societies (sound recording public performance in many territories)
- Rights tech vendors (fingerprinting, matching, rights data services)

WIPO's overview of the digital music ecosystem treats data exchange as core infrastructure and places DDEX standards in that stack. [WIPO]

1.4.2 Where intermediaries add drag

- **Time:** batches stack up; monthly platform reports can become quarterly society payouts and slower cross-border remittances.

- **Fees and breakage:** each entity charges admin fees or keeps certain funds under its rules; creators rarely see the full take rate end to end.
- **Opaque statements:** rightsholders often can't trace platform gross → allocation → distributor deductions → label recoupment → publisher fees → society fees → foreign deductions → FX.
- **Incentives:** platforms want low cost and stable licensing; labels want bargaining power; publishers want correct shares and compliance; societies want defensible rules; creators want faster, clearer, correct payouts.
- **Disputes:** intermediaries hold funds while claims settle, lowering platform legal exposure but extending creator payment times.

1.4.3 Minimum viable attribution

A common rule is: pay whoever you can identify with confidence; hold the rest. It reduces risk, but it shifts outcomes:

- Big catalogs with clean data get paid sooner.

- Long-tail and independent creators get hit harder by missing splits and claim gaps.

The MLC's match rates and the continued existence of large unmatched pools show this in a modern system built for the job. [The MLC]

1.5 Streaming complications: scale, models, micro-payments

Streaming changed the unit economics of rights management.

1.5.1 Micro-events at massive volume

A play is worth fractions of a cent. That creates two pressures:

- Reporting overhead grows huge compared to per-event value.
- Being wrong can cost more than the event is worth, so systems rely on pooling, thresholds, and batching.

Platforms have formalized thresholds and fraud controls that affect payout. Spotify has been covered for changes such as minimum stream thresholds for certain track payouts, framed as shifting tiny payments and reducing abuse. [AP]

1.5.2 Pro rata versus artist-centric models

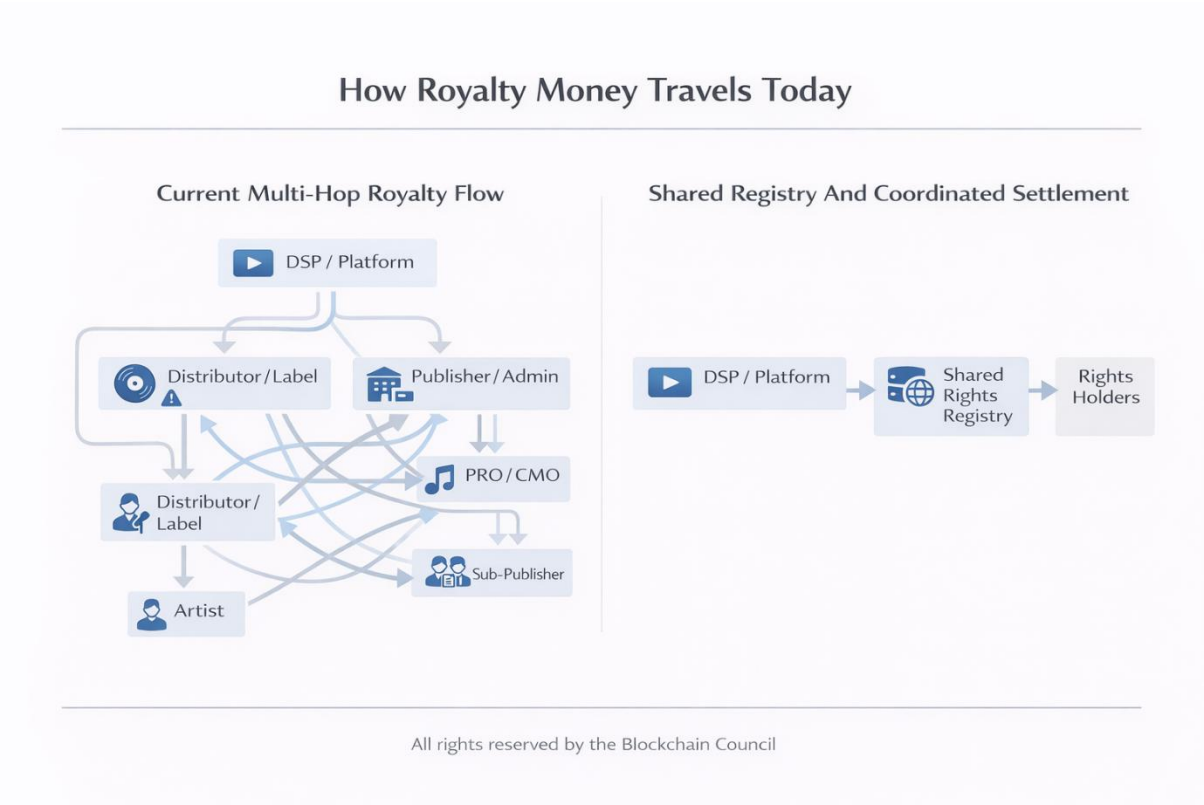
Pro rata allocation still dominates, but reform pressure is rising. Deezer and Universal Music Group announced an artist-centric model meant to pay more for active listening and limit fraud. Deezer also partnered with SACEM on publishing-side pay logic in the context of that model. [UMG; Deezer]

These models add new dependencies: define "engagement," detect fraud, and apply rules consistently across territories and licenses.

1.5.3 Concentration of payout power

Spotify reports more than $10B distributed in 2024 and about $60B paid since launch. Reuters reported Spotify's payout to the industry exceeded $11B in 2025. Spotify also says it represents about 30% of recorded music revenue, stressing its weight in 2025 growth. [Spotify; Reuters]

When a few platforms dominate, small policy choices in reporting and matching ripple through the whole system.



1.6 UGC platforms: every upload is a rights event

UGC changes the shape of rights management. Streaming services mostly distribute licensed recordings. UGC platforms ingest user uploads and then try to identify, monetize, or remove rights-bearing material at scale.

1.6.1 YouTube as the reference case

YouTube’s Content ID scans uploads against reference files from copyright owners and applies the owners’ chosen policy

when it finds a match. Policies can differ by owner and territory: monetize, block, or track. [YouTube Help]

By late 2025, YouTube reported paying more than $8B to the music industry in the 12 months from July 2024 to July 2025. [TechCrunch]

1.6.2 Short-form pool mechanics

Short-form feeds often rely on revenue pools and platform formulas that account for music licensing costs. YouTube's Shorts policies state that 45% of net YouTube Premium revenue allocated to monetizing creators is paid out, and that part of Premium revenue is set aside to cover music licensing costs. [YouTube Shorts]

Pooling can cut transaction overhead, but it makes it harder to trace "my song in that clip" to "this payment line."

1.6.3 A larger dispute surface

UGC complicates attribution because:

- Users upload edits, mashups, pitch shifts, and other variants that strain matching.

- Multiple rightsholders can claim the same segment.
- Dispute processes differ by platform.
- Music may be incidental background audio.

UGC systems must balance false positives (wrong claims) and false negatives (missed claims). Both cost money and create disputes.

1.7 Short-form social licensing turbulence (TikTok case)

Short-form platforms drive discovery, but their licenses are volatile.

1.7.1 The TikTok–UMG dispute and AI terms

In early 2024, negotiations between TikTok and Universal Music Group became public, leading to removals before a new deal. Reuters reported TikTok removing Universal Music Publishing content during the stalemate; UMG said TikTok represented about 1% of its revenue, while TikTok argued the catalogs represented a large share of popular music in some regions. UMG later announced a new licensing agreement

with TikTok, and coverage pointed to better compensation and AI-related protections. [Reuters; UMG; Pitchfork]

For royalty systems, the lesson is simple: modern deals bundle classic royalty terms with new product formats and AI restrictions. Any system that tracks rights and payments has to carry those conditions through huge volumes of UGC uses.

1.7.2 Payout logic is in flux

Reporting in early 2026 suggested TikTok was considering moving toward per-play payout logic rather than a single payment per video, reflecting long-standing pressure on how short-form platforms value music. [Music in Africa]

For researchers, the key point is that “usage” in short-form is not the same as a stream. It might be a view, an audio use, a creation event, or an engagement threshold. Each definition changes reporting and distribution outcomes.

1.8 AI-generated content: attribution chaos meets unsettled law

Generative AI brings two problems at once:

- More content, faster uploads, more edge cases.
- Murkier provenance: authorship, voice cloning, and training-related claims.

1.8.1 Copyrightability shapes whether royalties attach

In the US, the Copyright Office's AI report (Part 2) stresses that copyright requires human authorship. Purely machine-made material does not qualify, while outputs with enough human contribution may qualify. [USCO AI Part 2]

If an output isn't copyrightable, platforms can still pay under contract, but the legal footing for ownership and disputes is weaker.

1.8.2 Digital replicas push identity into licensing terms

The Copyright Office's AI report (Part 1) addresses digital replicas—voice and likeness imitation—and points to urgent protection needs. This lines up with industry deals that now add AI restrictions and safeguards, including in short-form licensing. [USCO AI Part 1; Pitchfork]

Performers' unions have also built contract frameworks around consent and disclosure for replica use, pushing "attribution" beyond songwriter credits into identity and permission. [SAG-AFTRA]

1.8.3 Training on copyrighted works raises new pay questions

The Copyright Office's AI training report (Part 3, pre-publication) describes disputes over fair use, licensing feasibility, and active litigation and policy proposals. [USCO AI Part 3]

If training compensation becomes a normal practice, it will require new accounting primitives: dataset provenance, opt-outs, training logs, and mapping training use back to rightsholders.

1.8.4 Deepfakes and spam strain payout integrity

Labels and platforms report large volumes of impersonation and spam. Sony Music says it requested removal of more than 75,000 AI deepfakes of its artists. Reporting in 2025 described

Spotify removing large quantities of spam tracks as AI lowered the cost of mass upload. [MBW; The Guardian]

Royalty pools are finite within a period. Fraud and spam can siphon value, distort allocation, and push platforms to impose stricter rules that also hit legitimate long-tail creators.

1.9 Synthesis: the 2026 pain map

By early 2026, digital revenue is strong and attribution is still messy. Any new system has to deal with:

1. **Fragmented data across rights and territories**
   Composition and recording data live in separate systems, and territory splits mean "one global truth" often doesn't exist.
2. **Probabilistic matching at scale**
   Even purpose-built entities like The MLC rely on reprocessing and still carry unmatched pools. [The MLC]
3. **Multi-hop money flows**
   Intermediaries solve real problems, but each hop adds delay, fees, and fewer ways to audit the full chain.

4. **Platform payout schemes that reduce traceability**
   Pools and policy-driven UGC monetization make it harder to tie a specific use to a specific payment line. [YouTube Shorts]
5. **AI-driven volume and uncertainty**
   Authorship rules, voice replicas, training disputes, and spam floods add new claim surfaces that older systems weren't designed to track. [USCO AI Part 1]

This is the benchmark for evaluating blockchain-based royalty infrastructure: not as a payment trick, but as a candidate data integrity and audit layer that still has to live inside law, contracts, and platform business rules.

Chapter 2 (Blockchain model): Rights on-chain, splits in code, payouts at scale

2.0 What a blockchain royalty model is trying to do

A blockchain royalty system isn't one upgrade. It's a set of design decisions about:

- how rights are represented in software

- where ownership and license facts live, and how they change
- how usage becomes a payable obligation
- how money reaches recipients
- how privacy, compliance, and disputes are handled

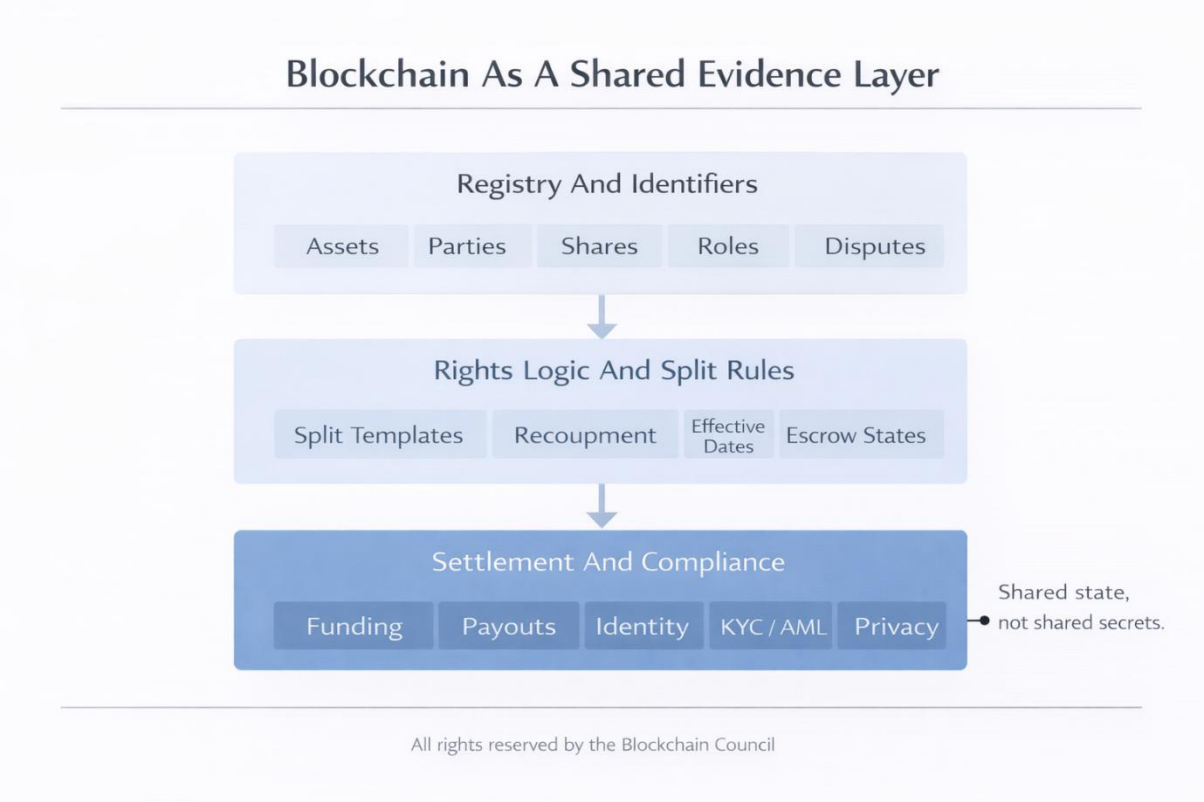
The 2026 target isn't "put music on chain." It's to make the rights and payment pipeline easier to verify, easier for machines to read, and less dependent on endless reconciliation between siloed databases. Chapter 1 described the baseline: broken metadata, weak links between works and recordings, and multi-hop accounting. A blockchain approach treats rights state and allocation state as something multiple parties can check, not a set of private spreadsheets.

It helps to separate three layers that teams keep mixing together:

- **Layer A: registry and identifiers** — what the asset is, who the parties are, and what is being claimed

- **Layer B: rights logic** — what entitlements exist, under what conditions, and how splits are computed
- **Layer C: payment and compliance** — how value moves, how identity is verified, what stays private, and how disputes freeze or redirect funds

The rest of this chapter defines those layers in enough detail to judge an implementation, not just the pitch.



2.1 Tokenized rights: what gets tokenized, and what a token means

“Tokenized rights” is an umbrella label for very different instruments. Treating them as the same is how projects collapse when they hit contracts, regulators, or real disputes.

2.1.1 Token as evidence of a claim vs token as the right

There’s a hard line between:

- **Token-as-pointer (evidence):** the token points to an off-chain agreement. Holding the token supports a claim, but the agreement creates the legal entitlement.
- **Token-as-right (the instrument):** the token is treated as the authoritative representation of the right inside the system, including transfer and partition rules.

Most serious systems end up hybrid. The token controls the software workflow and records evidence. Enforceability still sits in contracts that reference on-chain state.

2.1.2 What can be tokenized in music

A practical taxonomy:

1. **Royalty participation (economic interest):** a defined share of cash flows from a defined revenue stream (for a specific asset, territories, and time period).
2. **Administration rights (collection authority):** the ability to register, claim, and collect on behalf of creators—closer to agency or collection assignment.
3. **Licensing rights (permission to use):** a credential that grants a license for a defined use (snippets, stems, UGC tiers), best when one party controls the right stack.
4. **Governance rights (voting):** voting on policies such as pricing tiers or permitted uses, useful only when tied to enforceable rights and clean data.

Royalty participation is the most visible category, and also the one that runs into regulation fastest. If the design aims for scale, compliance has to be built in from day one.

2.1.3 Fractionalization: fungible, non-fungible, and partitioned structures

Music rights already operate in shares. Tokens just make the share math explicit.

Common representations:

- **NFT per asset:** one token per work/recording (or bundle), with fractional interests handled via multiple tokens or a vault that issues fungible shares.
- **Fungible token per pool:** one token represents a standard share of a defined revenue pool, with rules controlling who can hold it.
- **Partitioned tokens:** the same instrument can separate classes (locked vs transferable, territory tranches, share classes). ERC-1400 is often used as a reference point for this style.

Where transfer eligibility matters, permissioned standards such as ERC-3643 typically pair the token with an on-chain identity layer (for example, ONCHAINID) so the contract can block ineligible transfers.

2.1.4 Secondary-sale “royalties” are not music royalties

NFT marketplaces use “royalties” to mean a creator fee on secondary sales. That is not streaming, mechanical, performance, neighboring rights, or sync income.

EIP-2981 standardizes how a marketplace can read royalty payment information. It does not force payment. It's a signaling interface.

A music royalty system is about distributing underlying music revenue, not resale fees. Mixing these concepts creates systems that look neat and fail the first time a real license or audit shows up.

2.1.5 Tokenization requires identifier discipline

At scale, a token must tie to industry identifiers:

- **Recordings:** ISRC
- **Works:** ISWC
- **Releases:** UPC/GTIN
- **Parties:** IPI, ISNI, and related identifiers

On the finance side, token instruments also need stable identifiers. ISO 24165 defines Digital Token Identifiers (DTIs); ISO 24165-1 has an updated 2025 version.

A workable model uses both: music identifiers to anchor the asset, and token identifiers to anchor the instrument.

2.2 Registries: what belongs in an append-only ledger

People love “immutable” because it sounds like “correct forever.” Real systems need an append-only history with rules about who can post updates.

2.2.1 What the registry should record

Record facts that gain value when multiple parties can verify them:

- asset identity and linkage (work, recording, release, party links)
- claimed ownership shares and admin roles, including territory limits
- versioned license term templates (not necessarily full legal text)
- payment routing endpoints bound to identity
- dispute flags and state changes (claimed → contested → resolved)
- hashes of off-chain documents so integrity can be proven without publishing sensitive terms

2.2.2 Append-only as an audit trail

A practical model:

- don’t delete old ownership claims; supersede them
- don’t edit metadata in place; publish a correction event
- don’t erase disputes; record the resolution and the proof it references

The value is the trail: who claimed what, when, under what authority, and what changed after disputes or new evidence.

2.2.3 Who can write: governance and spam control

Rights registries cannot be open free-for-all write access unless “active state” requires strong verification.

Common models:

- **Consortium writes:** only verified entities can publish or co-sign updates.
- **Open claims, gated activation:** anyone can post a claim, but it becomes active only when required parties co-sign.

- **Hybrid:** core ownership changes require permissioned endorsements; long-tail claims can be posted and promoted after checks.

Whatever the choice, it should reflect existing authority flows instead of pretending they aren't there.

2.2.4 Data model: splits are conditional entitlements

Royalties aren't one percentage. They are conditional entitlements:

- territory (US, EU, rest-of-world)
- right type (mechanical, performance, neighboring rights, sync)
- time (term, windows, reversion)
- deal state (recoupment, caps, floors, escalators)
- usage class (subscription stream, ad-supported, short-form pools)

A registry that can actually drive payments needs a rights graph:

- **nodes:** works, recordings, parties, licenses, statements, payments
- **edges:** embodied-in, owned-by, administered-by, licensed-to, paid-to
- **attributes:** splits, constraints, effective dates, proof references

This graph already exists inside PRO databases, publisher systems, label catalogs, and reporting messages. The difference is making it portable and checkable across organizations.

2.3 Smart contract splits: automating math without turning deals into a code mess

Code can automate splits. It cannot fix bad input.

2.3.1 Split contract building blocks

Minimum pieces:

- recipients (addresses or payout endpoints)
- share rules (fixed shares, conditional shares, waterfalls)
- triggers (funding events, reporting periods)

- state transitions (recoupment status, rights transfer, dispute freeze)
- admin controls (upgrades, emergency pause, dispute flags)
- audit events (history of inputs, outputs, balances, proofs)

Music often needs nested splits: top-level divide between master-side and publishing-side allocations, then further splits inside each side.

2.3.2 Waterfalls, advances, and recoupment

A lot of music finance is recoupment against advances and cost pools:

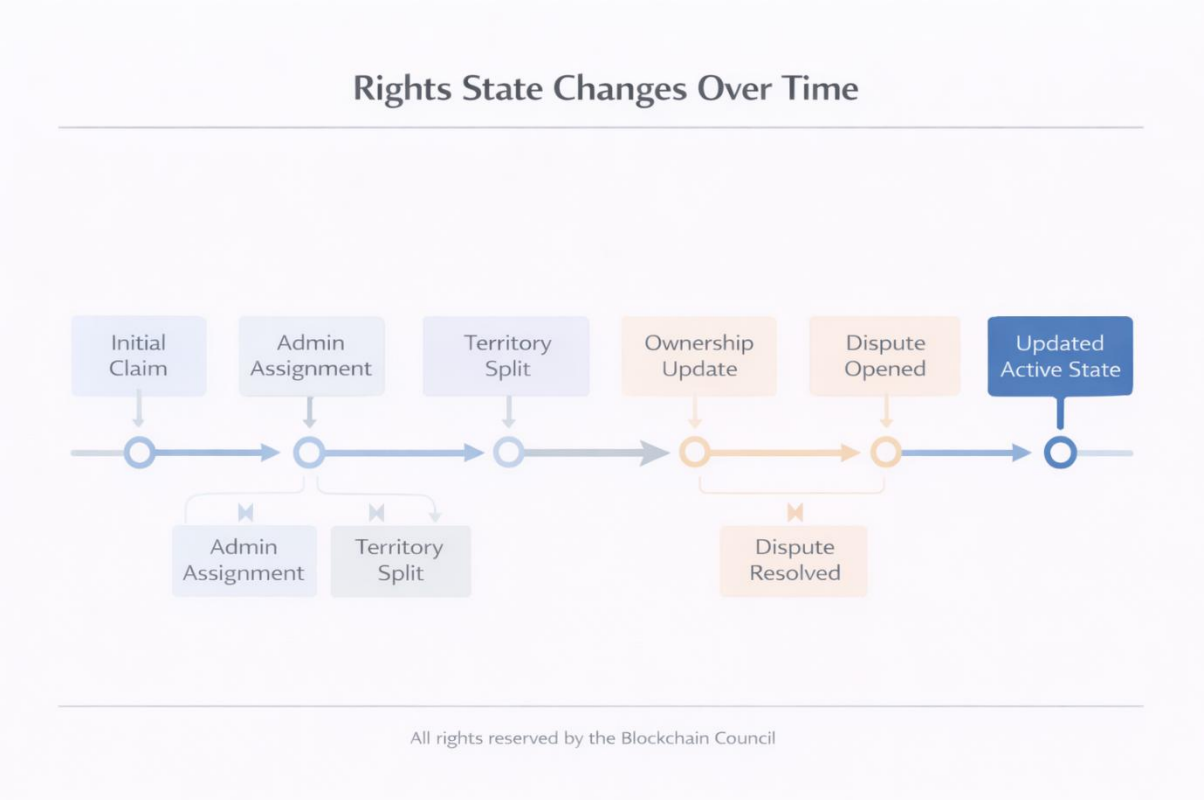
- label advances recoup against artist royalties
- publisher advances recoup against writer royalties
- distribution fees and marketing deductions apply before "net" exists
- joint ventures add more layers

A contract can implement this only if the terms are expressible as deterministic rules. That pushes systems toward templates.

A practical approach is **templates plus parameters**:

- template: standard recoupment waterfall
  - parameters: advance amount, recoupable categories, rate, cap, cadence, territory scope
- template: producer points on net receipts
  - parameters: points %, net definition, exclusions, audit window

Templates make audits and interoperability easier because different parties can compare behavior against the same logic.



2.3.3 Change is normal: upgrades and transfers

Deals change. Catalogs move. Admins change. Estates appear. Laws change.

So split contracts need:

- versioned upgrades with clear approval rules
- migration paths when rights transfer
- recorded authorization proofs (multi-sig, credential-backed permissions)
- human-readable terms anchored by hashes so parties can prove what they agreed to

2.3.4 Oracles: bringing usage onto chain

Contracts can’t see streams or matches. Usage data must be fed in.

The realistic approach is to reuse existing reporting formats and add verifiability:

- **off-chain report + on-chain hash:** publish the report off-chain; record a hash on-chain for timestamp and tamper detection

- **on-chain summary + inclusion proofs:** post totals plus a Merkle root so a recipient can prove their line items were included

This doesn't guarantee accuracy. It makes quiet edits harder.

2.4 Micropayments and "real time": what it can mean in practice

"Real time" is marketing unless the system can handle huge event volume at low cost and with clear settlement rules.

2.4.1 Why per-play on a base chain doesn't work

Mainstream services generate billions of events. Writing each as a chain transaction doesn't scale.

So systems aggregate:

- collect events off-chain
- post commitments and settle on-chain in intervals
- pay out more often once funds are in the system

2.4.2 Lower-cost execution layers

Many designs settle on an execution layer built for cheaper transactions, with periodic settlement to a base layer.

Ethereum's Dencun upgrade, anchored by EIP-4844 (blob-carrying transactions), was intended to lower rollup costs by improving data availability economics. For royalties, the impact is simple: smaller transfers become less expensive, which makes more frequent settlement plausible. It still doesn't mean one transaction per play.

2.4.3 Three payout architectures

1. **Periodic funding, ongoing payouts:** platforms deposit daily/weekly; recipients receive continuous or frequent payouts based on the latest verified splits.
2. **Usage-verified settlement:** funds release after a proof arrives for the period's report (signed attestations, Merkle proofs, or ZK proofs). Higher audit value, higher complexity.
3. **Payment channels / streaming protocols:** most relevant for direct fan payments or wallet-native platforms, not mainstream DSPs paying from fiat.

A hard constraint remains: most DSP revenue is collected in fiat and processed in reporting cycles. Blockchain can speed distribution after funds enter the system. It does not erase upstream accounting unless platforms adopt on-chain settlement.

2.4.4 Fiat, taxes, and refunds

Real royalty systems deal with:

- many fiat currencies
- tax withholding
- banking rails
- refunds and chargebacks
- regulatory reporting

Stablecoin settlement can cut friction, but it raises custody and compliance issues. If the platform or service layer is treated as a crypto-asset service provider, it may fall under regimes such as MiCA. MiCA's CASP rules applied from December 30, 2024, with possible grandfathering until July 1, 2026 depending on member-state choices.

2.5 Identity, KYC, and compliance: paying the right legal person

Royalties belong to legal persons: individuals, companies, estates, trusts. If a system can't bind an address to a verified identity, it can't pay the right party, apply tax rules, or satisfy AML requirements.

2.5.1 The baseline: Travel Rule and regulated intermediaries

For crypto-asset transfers, the FATF Travel Rule requires certain intermediaries to obtain, hold, and transmit originator and beneficiary information. Depending on architecture and jurisdictions, a royalties-on-chain system can trigger Travel Rule obligations.

2.5.2 Identity stack: DIDs and verifiable credentials

Modern systems often use:

- **Decentralized Identifiers (DIDs):** identifiers designed to avoid a single central registry.
- **Verifiable Credentials (VCs):** credentials issued, held, and verified with tamper-evident proofs.
- **Presentation protocols:** a standard way to request and present credentials.

For royalties, this supports:

- KYC credential issuance by approved providers
- proof that a wallet is controlled by the credential holder
- selective disclosure so a party can prove eligibility without publishing their full identity

2.5.3 What must be bound to a payout endpoint

A royalty model typically needs:

- legal identity record (including company registration details when relevant)
- tax identity and withholding data
- sanctions screening status (or proof of screening)
- wallet control proofs (signatures, DID-to-wallet linkage)
- role credentials (publisher admin, label, estate representative)

2.5.4 Transfer controls for regulated instruments

If tokens represent regulated economic interests, systems often require:

- only verified holders can receive or transfer
- jurisdiction restrictions
- lockups
- investor eligibility constraints

This is why permissioned token standards and identity-gated transfers matter: the contract must be able to refuse a transfer.

2.6 Interoperability with PROs and existing music data plumbing

A blockchain system that ignores PROs, CMOs, and DDEX won't scale. The usage data and legal mandates already live there.

2.6.1 Don't replace standards; connect to them

Use industry reporting and metadata formats as the intake and output layer:

- ingest release and resource metadata
- ingest work rights and ownership updates
- ingest credits data
- reconcile usage reports into payable obligations
- export statements back in formats current systems can ingest

The on-chain layer should carry checkable state and checkable commitments, while the message layer carries the operational detail.

2.6.2 Two workable integration models

- **Shared state, societies still collect:** PROs/CMOs keep collecting from licensees and distributing, but use the registry as a shared source for splits, updates, and dispute status.
- **Societies as network participants:** in a consortium chain, societies, major publishers, labels, distributors, and large platforms run nodes and co-sign rights states and distribution events, while private terms stay protected.

2.6.3 The non-technical constraints

Societies operate under national laws and reciprocal agreements. A system must represent:

- society mandates by right and territory
- local categories that don’t map neatly to global labels

- distribution rules that can't be fully public

Even the best code fails if societies can't treat the data as evidence, can't audit it, or can't fit it into statutory obligations.

2.7 Privacy: permissioned networks, selective disclosure, and ZK proofs

Royalty data is sensitive: ownership disputes, deal terms, and personal identity data. Publishing it all on a public chain is a non-starter for serious participants.

2.7.1 Permissioned networks and private data tools

Permissioned ledger platforms commonly support private payloads with shared hashes:

- private data collections (store the data only with authorized peers; commit hashes to the ledger)
- confidential identities and point-to-point transaction sharing
- private transaction managers where only involved parties see the payload

These designs keep shared state checkable without giving everyone the underlying details.

2.7.2 Public settlement with protected data

If settlement happens on a public chain, privacy can still be protected by:

- keeping sensitive data off-chain, encrypted
- recording hashes and commitments on-chain
- revealing details only to authorized verifiers

2.7.3 ZK proofs: prove facts without exposing the data

Zero-knowledge proofs let a party prove a statement without revealing the underlying records.

In royalties, that can mean:

- a platform proves payout totals match a committed report without exposing user-level logs
- an admin proves split calculations followed the approved template and parameters without exposing everyone's balances

- a recipient proves they passed KYC and meet eligibility rules without publishing identity details

2.7.4 A workable “what goes where” map

- **On-chain:** asset IDs, state hashes, contract addresses, payment event hashes, dispute flags, high-level settlement totals
- **Off-chain (restricted):** full legal text, contributor identity details, tax forms, user-level usage logs, sensitive commercial terms
- **Proof layer:** Merkle proofs or ZK proofs that let stakeholders verify integrity and correctness without full disclosure

2.8 End-to-end reference architecture

A complete model can be described as a pipeline:

1. **Identity onboarding:** rights holders obtain a DID plus a KYC credential from an approved issuer.

2. **Asset registration and mapping:** register works and recordings with industry identifiers and link recordings to underlying works.
3. **Token issuance (optional):** issue instruments that match the compliance stance (unrestricted for internal admin flows; identity-gated for regulated interests).
4. **Platform integration:** platforms ingest registry state and licensing terms through standard message formats or APIs that output those formats.
5. **Usage reporting commitments:** platforms produce usage reports off-chain and post on-chain commitments (hashes or Merkle roots).
6. **Funding:** platforms fund settlement contracts on a low-cost execution layer or a permissioned network.
7. **Distribution:** contracts pay according to recorded splits and deal state; privacy uses commitments and proofs when needed.

8. **Audit and disputes:** disputes freeze affected shares; undisputed shares keep flowing; reprocessing happens when mappings and ownership state improve.

2.9 Risks and limits: what blockchain won't fix

2.9.1 Bad metadata stays bad

A ledger doesn't repair missing splits or wrong mappings. It preserves the history. That helps accountability, not accuracy.

So the design needs verification workflows, incentives for accurate registration, and a dispute process that can freeze funds and then release them when resolved.

2.9.2 Oracle trust is still the weak point

If usage input is wrong, the system pays the wrong amounts faster.

Mitigations include:

- commitments plus audits
- multiple inputs (platform reports plus society reports where available)

- internal-consistency proofs (totals match committed line items)

2.9.3 Privacy and audit pull in opposite directions

Creators want visibility. Companies want confidentiality. Regulators want traceability. Users want privacy.

Selective disclosure is the only workable middle path.

2.9.4 Classification risk for tokenized royalty interests

Royalty participation tokens marketed broadly can be treated as securities or regulated financial products in many jurisdictions. Even without taking a legal stance, the technical design should support compliance: eligibility checks, transfer rules, and clear role separation for regulated service providers.

2.9.5 Key management and recovery

Losing a private key should not mean losing income. Systems need recovery paths suitable for creators and estates, plus secure custody options where appropriate.

2.10 What the blockchain model contributes in 2026

A well-specified blockchain royalty model can provide:

- a shared, append-only rights state that multiple stakeholders can check
- machine-readable splits and versioned deal logic in code
- more frequent settlement once funds enter the system, without per-play chain writes
- identity and compliance controls that bind payout endpoints to legal persons
- interoperability with existing music metadata and reporting workflows
- privacy through restricted data sharing and cryptographic proofs

None of this removes law, contracts, or platform business rules. It changes how the system records state, proves integrity, and moves money after usage has been measured and funds have been allocated.

Chapter 3 (Implementation and risks): Getting it into production

3.0 Implementation reality check: building plumbing, not a demo

A blockchain royalty system becomes real only when it survives the dull, expensive parts: shared governance, identity and compliance, disputes that lock money, ongoing upgrades, and integration with royalty infrastructure that won't be replaced on a whim. The hard problem isn't writing contracts. It's running a shared rights and settlement layer across organizations that do not share incentives, legal exposure, or data habits.

This chapter lays out what production deployment requires—and what can still go wrong even if the architecture looks clean.

3.1 Governance: who runs the network, who can write state, who can change rules

Governance comes first. Without it, a registry turns into competing claims and spam. With governance that is too centralized, the system recreates the same trust choke points it claims to remove.

3.1.1 Network governance models

Most viable deployments converge on one of three models:

- **Consortium permissioned network:** known institutions operate nodes under shared rules. This fits royalties because the main stakeholders already exist as identifiable entities (DSPs, labels, publishers, distributors, societies, large admins) and because privacy and compliance usually require vetting.
- **Public settlement, permissioned writes (hybrid):** settlement happens on a public chain, but rights-state updates require authorized signatures (for example, multi-party co-signing by recognized admins). This keeps settlement broadly verifiable without allowing anyone to post ownership state.
- **Open public registry (rare in production):** anyone can post claims. This can help long-tail inclusion, but it collapses under adversarial pressure unless "active state" is heavily gated and disputes can throttle bad claims.

3.1.2 What governance must cover

A production system needs explicit decisions in at least six areas:

1. **Membership and identity rules**

   Who can run nodes, who can submit updates, and what verification is required.

2. **Data rules**

   Canonical identifiers, required fields, validation checks, versioning, and how corrections are recorded.

3. **Contract control**

   Who can deploy, upgrade, pause, and retire settlement and split contracts.

4. **Economic rules**

   Fee schedules, minimum payout thresholds, dispute reserves, and how infrastructure costs are shared.

5. **Compliance rules**

   KYC expectations, sanctions screening responsibilities, reporting duties, and what happens when a recipient is blocked.

6. **Dispute and appeal rules**

   What counts as a dispute, what evidence is acceptable, timelines, and which forum decides.

A practical structure is two-tier governance:

- **Technical steering:** protocol, security, upgrades.
- **Rights and policy council:** data rules, authority roles, dispute standards, compliance controls.

Keep them separate. “Safe for the system” and “acceptable to the market” often conflict.

3.1.3 Governance artifacts: the paperwork that makes it workable

Governance becomes real through documents and processes, not slogans:

- consortium agreement (membership, liability, cost sharing)
- rulebook (schemas, authority roles, state transitions)
- security and key handling policy (including incident response)

- audit rights and audit interfaces
- dispute policy that maps on-chain actions to off-chain outcomes

If these don't exist, participants will demand them anyway—because nobody wants to litigate a protocol spec.

3.1.4 Node operation and trust boundaries

In permissioned settings, node operators become trust anchors:

- **DSP nodes** attest to usage commitments and funding events.
- **Publisher/label/admin nodes** co-sign ownership and split updates.
- **Society nodes** attest to mandates, reciprocal routing, and distribution events.
- **Auditor nodes** verify that commitments match disclosed statements without reading private payloads.

You don't remove trust. You decide where it sits and how it is checked.

3.2 Onboarding creators: identity, UX, and key loss

A system that only large catalogs can use is not a rights system. It's an internal reconciliation tool. Creator onboarding is central, and it has three constraints:

- identity must be verifiable
- privacy must be protected
- the UX cannot require users to become part-time security engineers

3.2.1 Identity primitives: DIDs and credentials

Most production designs use a credential model:

- a creator has an identity identifier
- an approved issuer attests to KYC completion and role claims
- the creator can prove eligibility to receive payments without publishing personal data on-chain

This also supports role-based authority: publisher admin, label admin, estate representative, society mandate holder.

3.2.2 AML, KYC, and the Travel Rule reality

If the system moves value—especially over crypto rails—assume AML obligations will attach somewhere. That forces a clear decision about where compliance is enforced:

- **at onboarding:** only verified recipients can receive payouts
- **at transfer:** payout endpoints are gated; ineligible transfers fail
- **at off-ramp:** fiat payouts run through regulated payment providers

Even “decentralized” deployments usually rely on regulated services for custody, stablecoin settlement, and fiat payouts.

3.2.3 Regional regulation as a design constraint

If the system touches Europe, MiCA matters in practice: authorizations, marketing constraints, and the line between regulated and unregulated activity. Treat this as an engineering constraint, not a legal footnote.

3.2.4 A production onboarding flow

1. **Identity verification**

   Creator completes KYC with an approved provider and receives a credential.

2. **Rights identity binding**

   The creator links industry identifiers through attestations from an admin, publisher, distributor, or society.

3. **Wallet binding and recovery**

   The creator binds a payout endpoint to their identity with signed proofs, with recovery options.

4. **Payment preferences and tax setup**

   Payout preferences, withholding, and jurisdiction flags are captured off-chain, with on-chain commitments when needed.

5. **Ongoing maintenance**

   Credentials expire. Banks change. Estates appear. Rights transfer. This is continuous ops.

3.2.5 Self-custody vs managed custody

Self-custody gives control and raises loss risk. Managed custody lowers loss risk and simplifies support, but brings back intermediaries. Real deployments tend to be hybrid:

- self-custody is allowed
- the default is a managed option with recovery and human support

That isn't ideology. It's what you do if you want creators to actually get paid.

3.3 Disputes: handling conflict without freezing the whole system

Rights disputes are steady-state. A system that can't contain them either freezes too often or pays the wrong party fast.

3.3.1 Disputes the system must handle

- ownership disputes (two parties claim the same share)
- split disputes (contributors disagree on percentages or roles)
- territory disputes (sub-publishing, mandates)

- derivative-use disputes (samples, interpolations, covers, remixes)
- identity disputes (compromised payout endpoint, bad binding)
- usage disputes (challenged reports, suspected fraud)

3.3.2 Isolate disputes to the affected shares

Avoid global freezes. A better rule:

- undisputed shares keep paying
- disputed shares go to escrow
- resolution releases escrow per the decision, with retroactive adjustments if needed

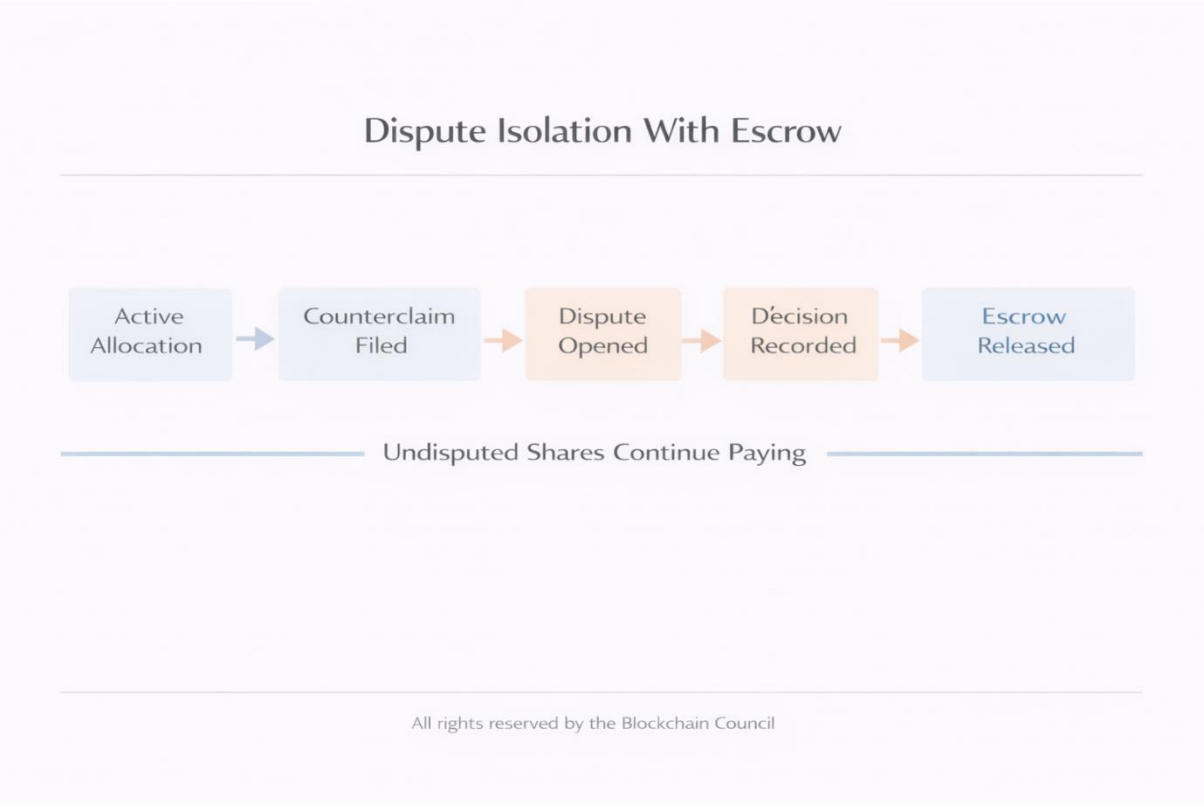
3.3.3 Dispute state machine

A practical workflow recorded as append-only events:

- claim submitted → pending verification
- verified claim → active allocation
- counterclaim → dispute opened
- dispute opened → disputed share escrowed

- evidence window → adjudication started
- decision → allocation updated, escrow released
- appeal window → escrow remains locked
- finality → state marked final for the period

Evidence stays off-chain where needed, but is hashed so later changes are detectable.



3.3.4 Forums: off-chain first, on-chain only by consent

For high-value disputes, parties will use courts or arbitration. On-chain dispute tools can help for smaller claims where speed matters and everyone consented up front.

If you want any arbitration result to matter, consent must be explicit in the onboarding agreement. Otherwise it's just a nicer form of "please."

3.3.5 Abuse controls

Dispute systems get attacked. Production controls include:

- rate limits and/or staking for certain claim actions
- credential checks for high-impact updates
- anomaly detection for suspicious usage patterns
- emergency pause controls under multi-party approval

3.4 Legal enforceability: code executes, law interprets

A court or arbitrator must be able to understand what was agreed, what happened, and why funds were paid or withheld.

3.4.1 Pair code with a readable agreement

Don't rely on code alone. Use a human-readable contract that:

- states legal intent
- references the code (address and/or hash)
- defines precedence if prose and code diverge
- sets jurisdiction and dispute forum

3.4.2 Upgrades, ambiguity, and change management

Music deals change. Code needs deterministic rules. The bridge is:

- standardized templates with parameters
- defined amendment paths (governance-approved)
- versioned upgrades with recorded authorization

3.4.3 Insolvency, liens, assignments, and forced updates

Royalties intersect with insolvency, recoupment, security interests, and assignments. The registry must represent these legal events and support governed updates compelled by lawful orders.

3.4.4 Financial regulation risk

Tokenized economic interests and trading can trigger securities or financial-product regimes. Even if the network doesn't take a legal position, the build should support compliance paths: identity gating, transfer restrictions, and clear separation of roles.

3.4.5 Privacy and data protection

Royalties include personal data and sometimes sensitive usage detail. Do not write personal data to an immutable public ledger. Keep sensitive data off-chain, and use commitments and selective disclosure.

3.5 Scaling: billions of events, limited bandwidth, batching by design

At streaming scale, throughput and data volume dominate. Treating each play as a chain transaction turns the system into an expensive stunt.

3.5.1 What belongs on-chain at scale

- commitments to reports (hashes, Merkle roots)
- allocation state and version history

- aggregated settlement totals
- payment events (at the level of payout batches, not every play)

Per-play logs remain off-chain, with audit proofs available on demand.

3.5.2 Settlement patterns

- **Periodic funding, frequent payouts:** deposit daily/weekly; payouts run continuously or at short intervals.
- **Proof-gated settlement:** payouts release after report commitments and proofs arrive.
- **Hybrid:** baseline payouts flow; reconciliation adjusts later.

3.5.3 Permissioned vs public tradeoffs

Permissioned networks can push higher throughput because participants are known. Public chains are easier to verify broadly but add fee and throughput constraints. Many deployments split duties: permissioned for rights state and

report commitments, public chain for net settlement when needed.

3.5.4 Contract risk at scale

Scale increases the blast radius of bugs. Production hygiene includes:

- independent audits
- monitoring and alerting on anomalous flows
- emergency pause under multi-party control
- staged rollouts and replay testing against real report data

3.6 Oracles: the system is only as good as its inputs

Oracles bridge “what happened” to “what gets paid.” In royalties, the key oracle payload is usage and revenue reporting from platforms and intermediaries.

3.6.1 Failure modes

- wrong data (mistakes or fraud)
- missing fields (territory, usage class)
- delayed reports (breaks payout cadence)

- skewed aggregation rules (totals wrong even if raw logs exist)
- compromised signing keys

3.6.2 Practical mitigations

- commit every report (hash it, timestamp it)
- allow inclusion proofs for line items
- use multiple attestations where possible (platform + admin + society)
- keep audit rights and audit processes explicit

The goal isn't "no trust." It's making silent changes hard and legally risky.

3.7 Energy and cost: consensus choice and operating budget

Energy and cost show up as real operating expense, reputational risk, and sometimes regulatory pressure.

3.7.1 Energy profile

Proof-of-work systems consume far more energy than proof-of-stake and most permissioned consensus models. Ethereum's move to proof-of-stake is commonly described as cutting energy use by around 99.95%.

3.7.2 Cost drivers

Cost isn't only transaction fees:

- fees or infrastructure costs for the ledger
- storage for commitments and state
- compute for proofs and audits
- KYC, payouts, tax handling
- support and recovery operations

Batching keeps cost tied to reporting intervals rather than event counts.

3.7.3 Privacy tooling costs

Privacy systems add components and operational overhead (private payload distribution, certificate management, access policies). Many teams accept this because the alternative—publishing deal terms and personal data—is worse.

3.8 Integration with existing royalty systems: adoption means coexistence

A blockchain system that demands the industry restart from zero will fail. Integration is the adoption path.

3.8.1 Where integration happens

1. **Metadata supply chain**
   Ingest industry identifiers and catalog metadata; output validated state back to distributors, publishers, and DSPs.
2. **Usage reporting**
   Ingest standardized usage and revenue reports; post commitments and proofs.
3. **Settlement funding**
   Funds enter from DSPs, societies, and licensees; payouts follow governed split logic.
4. **Accounting and statements**
   Output statements creators and admins can use: line items, adjustments, audit trails.

3.8.2 Matching and reprocessing don’t disappear

Even modern centralized systems still match, fix links, and reprocess pending money as data improves. A ledger can reduce reconciliation work and tighten audit trails. It cannot remove the need for matching.

3.8.3 PRO/CMO interoperability patterns

- **Ledger as shared reference:** societies keep collecting and distributing under mandates, while using the ledger for ownership state and reciprocal routing checks.
- **Societies as nodes:** societies participate directly as network members, contributing attestations and receiving updates.

3.8.4 Migration and dual running

A safe rollout is almost always parallel operation:

- keep existing systems running
- run the new system for a limited slice of repertoire/territory/usage class
- compare statements, resolve gaps, harden dispute handling

- expand scope gradually

3.9 Risk register: what can still go wrong

3.9.1 Governance capture

Risk: dominant stakeholders control updates, censor claims, or set fees to their advantage.

Mitigation: multi-stakeholder voting rules, published change processes, independent audits, appeal paths.

3.9.2 Bad data as permanent evidence

Risk: incorrect ownership or mappings get recorded and later used as leverage.

Mitigation: gated activation, multi-party co-signing, versioned corrections, dispute flags that stop payment.

3.9.3 Contract bugs and upgrade mistakes

Risk: funds routed wrong or locked.

Mitigation: audits, staged deployment, pause controls, conservative upgrade permissions.

3.9.4 Oracle manipulation and reporting fights

Risk: wrong usage data produces wrong payouts.

Mitigation: report commitments, inclusion proofs, multi-source attestations, audit rights.

3.9.5 Identity failure

Risk: compromised keys or bad bindings redirect money.

Mitigation: credential-based binding, recovery, managed custody option.

3.9.6 Compliance failure

Risk: payouts violate sanctions, AML rules, or local licensing constraints.

Mitigation: KYC gating, region-aware controls, Travel Rule-capable partners where required.

3.9.7 Privacy leakage

Risk: deal terms or personal data become public or inferable.

Mitigation: keep sensitive data off-chain; use commitments, restricted access, selective disclosure.

3.9.8 Cost overruns

Risk: the system costs more than incremental upgrades to existing databases.

Mitigation: batching, short settlement intervals without per-event writes, clear cost-sharing agreements.

3.10 Implementation roadmap: a pragmatic path

A credible deployment plan usually looks like this:

1. **Consortium formation and governance charter**
   Define membership, liability, cost sharing, rulebook, and minimal data model.
2. **Pilot repertoire and limited territories**
   Pick a slice with clean admin authority. Integrate reporting inputs. Run parallel statements.
3. **Dispute system hardening**
   Test counterclaims, escrow flows, evidence handling, and release rules.
4. **Expand to complex usage classes**
   Add pooled revenue categories, UGC matching attestations, and fraud controls.

5. **Cross-border and society interoperability**

   Add society attestations or nodes. Test reciprocal routing and consistency across territories.

6. **Mature operations**

   Regular audits, incident drills, compliance reporting, and continuous improvement of matching and onboarding coverage.

Conclusion

1. What the 2026 landscape demands from any new system

Chapter 1 shows a royalty economy with strong topline revenue and weak attribution plumbing. That combination creates a specific kind of frustration: money is in the system, but the path from use to pay is still hard to trace and slow to settle for many creators.

Several conditions define the 2026 baseline.

- **Rights are split and administered through different institutions.** A single play produces obligations across separate right bundles, and those obligations travel different paths.
- **Metadata remains incomplete at the moment it matters most.** Usage begins as soon as a track is released, while writer splits, contributor credits, and recording-to-work links are often unfinished.
- **Cross-border routing adds delay and data loss.** Every hop forces data conversion and introduces more places for shares and identifiers to drift.
- **UGC and short-form platforms introduce new reporting shapes.** Policy-driven monetization and

pooled payouts reduce the ability to trace a specific use to a line item.

- **AI increases volume and increases uncertainty.** The system must handle more tracks, more edge cases, more impersonation, and more spam.

This baseline matters because it rules out simple fixes. No single database, no single identifier, and no new payment rail can solve the entire stack. Any candidate system must focus on what it can change: state, evidence, and coordination.

2. What blockchain can add, when used with restraint

The model in Chapter 2 treats blockchain as an evidence layer and a settlement coordinator rather than a replacement for contracts and institutions. Under that view, blockchain can add value in six places.

2.1 A shared, append-only record of rights state

A rights registry is most useful when it records facts that multiple parties need to check: asset identity, ownership shares, admin roles, effective dates, territory limits, and dispute flags. The key point is append-only history.

Ownership changes are common. Errors are common. A system that cannot preserve history becomes a new reconciliation loop.

Append-only history does not produce truth by itself. It does produce a trail of claims, approvals, and corrections. That trail makes it harder to change the story quietly.

2.2 Split logic that is consistent across payers and payees

Royalties involve arithmetic. The same split should produce the same output regardless of which back office is running it. When split logic is expressed as code plus parameters, parties can test it, audit it, and replay it. This helps most where rules are deterministic: fixed shares, recoupment waterfalls defined in a template, time-based changes, and dispute freezes.

This does not remove bespoke deals. It creates pressure to express bespoke deals in a way that can be computed.

2.3 More frequent distribution after funds enter the system

Most mainstream platform revenue is still collected in fiat and processed in reporting cycles. A blockchain layer cannot

change that unless platforms move to on-chain settlement. What it can do is shorten distribution once the funds are funded into the settlement layer.

The practical route is batching: commit to usage data off-chain, post commitments on-chain, and distribute in frequent intervals without trying to record every play as a chain transaction.

2.4 Better evidence for reporting integrity

Usage reporting disputes often turn on a simple question: did a report change after the fact? Cryptographic commitments help here. If a platform posts a hash or Merkle root for a report, later changes become detectable. A rights holder can verify inclusion of line items without needing the platform to expose everyone else's data.

This does not guarantee that the platform's report is correct. It does make quiet edits harder.

2.5 Identity-bound payout endpoints

Royalty systems pay legal persons, not anonymous wallets. Binding payout endpoints to verified identities reduces

misdirected payments and makes compliance more workable. It also enables role claims: who is allowed to update splits, who can file claims, and who can act for an estate.

Identity work is uncomfortable for many crypto projects. It is unavoidable for royalties.

2.6 Privacy through restricted data sharing and selective disclosure

Royalty systems contain sensitive information: deal terms, contributor identities, tax details, and in some cases user-level logs. Publishing this to a public ledger is not viable. A production design must use restricted payloads and commitment-based proofs.

Permissioned networks can limit who sees what. Public settlement can still work if sensitive data stays off-chain and parties use commitments and selective disclosure.

3. What blockchain does not fix

Chapter 3 is blunt about limits.

3.1 Bad metadata remains the root cause

A ledger does not create missing writer splits. It does not resolve a sample dispute. It does not identify session musicians who were never credited. It records what someone claimed and what others approved. That can improve accountability, but it does not create correct data.

Therefore, any serious system must invest in upstream workflows: split capture at creation, contributor credit capture, and mapping discipline between recordings and works.

3.2 Oracles remain the weak point

Blockchains cannot observe streams, matches, and plays. Usage data must be introduced. That introduces a trust and security surface: signing keys, aggregation logic, and report delays.

The right way to talk about oracles in royalties is not “trustless.” It is “auditable.” Commitments, inclusion proofs, multi-party attestations, and audit rights reduce the chance of undetected manipulation.

3.3 Governance does not disappear

A shared registry needs rules. Those rules need a way to change. That is governance. The choice is not "governance or no governance," but whether governance is explicit and checkable or informal and dominated by whoever controls the infrastructure.

3.4 Tokenization creates regulatory and consumer risk

Tokenizing royalty participation can be useful as a way to represent economic interests and route cash flows, but it invites financial regulation. It also creates consumer harm risk if marketed irresponsibly.

A royalty system can still use blockchain without making economic interests freely tradeable. Tokenization is optional. Rights state and settlement are not.

3.5 Faster payouts do not automatically mean fairer payouts

More frequent distribution is attractive. It can also move money quickly based on wrong inputs. If the system pays fast and corrects later, creators can face clawbacks and confusion.

If the system holds money until everything is final, creators face delays.

There is no free choice here. The best systems make the tradeoff explicit: baseline payouts for undisputed shares, escrow for disputed shares, and clear adjustment rules.

4. Production design principles that survive contact with the industry

Across Chapters 2 and 3, several design principles show up repeatedly. These are the principles that make the difference between a demo and a system that can be run for years.

4.1 Separate registry, rights logic, and settlement

Projects fail when they treat “rights” as a single blob. Keep layers distinct.

- Registry: identity, asset links, shares, roles, effective dates, disputes.
- Rights logic: contract templates, parameters, state transitions such as recoupment.

- Settlement: funding, distribution cadence, payout endpoints, tax and compliance hooks.

This separation also makes change safer. You can correct registry state without redeploying settlement logic. You can upgrade a template without rewriting identity.

4.2 Make “active state” gated by authority

Letting anyone write claims is fine at the edge. Letting claims control payments is not fine.

A practical rule is:

- anyone can submit a claim
- a claim becomes active only after recognized authority signs it
- disputes freeze only the affected share, not the whole asset

Authority can be multi-sig by admins, co-signing by publishers above a threshold, or endorsements by societies where mandates apply.

4.3 Treat disputes as first-class state

Disputes are not bugs. They are a normal operating condition.

A production system needs:

- a dispute state machine
- share-level escrow
- evidence handling with commitments
- decision recording and appeal windows
- clear retroactive adjustment logic

If the system can't do this, it will either lock too much money or pay the wrong party.

4.4 Keep sensitive data off public ledgers

Do not write personal data and deal terms to a public ledger. Do not write user-level listening logs. Use commitments and restricted access. Use selective disclosure when verification is needed.

This is not a minor detail. It determines whether serious participants will even consider joining.

4.5 Design for batching and replay

At scale, the unit of settlement is a report period, not a play.

- batch events off-chain
- commit reports on-chain
- distribute based on the committed state
- support replay testing and audits

Replay testing is essential. You need to be able to re-run the same report through the system and get the same output.

4.6 Build key recovery and estate handling

Creators lose phones. Keys get compromised. Estates inherit rights. A payout system that cannot handle recovery and succession is not ready.

Make recovery normal:

- managed custody option
- multi-party recovery for self-custody
- clear procedures for estates, trustees, and company role changes

4.7 Interop is a requirement, not a feature

A blockchain system must accept the world it is joining.

- it must ingest existing identifiers and metadata
- it must ingest existing usage reports
- it must output statements and adjustments in formats current systems can use

If it cannot do that, it becomes a parallel world with no adoption path.

5. A practical deployment path

Chapter 3 argues for phased rollout because royalty systems cannot be "moved fast and broken." Breaking a payout pipeline means creators do not get paid.

A realistic path looks like this.

5.1 Start with governance and a minimal data model

Before writing code, define:

- who participates
- who pays for the network
- who can write rights state

- what “active” means
- which identifiers are required
- how corrections and disputes work

Then write the minimum registry schema that supports a pilot.

5.2 Pilot with a narrow slice that has clean authority

Pick a repertoire slice with clear admin authority and fewer chain-of-title issues. Run the system in parallel with existing statements. Compare outputs. Fix mismatches. Harden tooling.

5.3 Add disputes and escrow before scaling

A pilot that pays everyone based on assumed clean splits is a pilot that has avoided the real work. Add controlled counterclaims, escrow flows, and evidence handling while the scope is still small.

5.4 Expand to harder usage classes

Once the core loop works, expand into:

- pooled short-form reporting

- UGC policy-driven monetization
- fraud and spam controls

This is where the system's audit trail and proof methods matter most.

5.5 Add cross-border and society integration

If the goal is broad impact, the system must handle cross-border shares and society mandates. That may involve societies as participants, or it may involve attestations and shared registry state that societies can accept.

6. What "success" looks like in measurable terms

It is easy to claim success in vague language. A production system needs metrics.

A practical scorecard includes:

- **match rate at first pass** for works and recordings
- **time to first payment** after usage occurs
- **share of usage in escrow** and average time to resolve

- **rate of corrections** and how often corrections change payments
- **audit cycle time** (how long it takes a rights holder to verify a statement)
- **operational incidents** (key compromise, report delays, contract pauses)
- **cost per dollar distributed** including compliance and support overhead

These measures allow a clear comparison against the baseline system.

7. The central takeaway

Blockchain is a tool for shared state and shared evidence. In royalties, that is valuable only when the system is designed around authority, disputes, privacy, and interop.

A serious royalty system is not a chain with music on it. It is a set of boring agreements, strict data rules, careful identity handling, dispute escrow, verifiable reporting commitments, and settlement that does not collapse under volume.

If those parts are in place, a blockchain layer can reduce reconciliation loops, shorten distribution cycles after funding, and give creators a clearer path to audit what happened. If those parts are missing, the project will recreate the same delays and arguments in a new interface.

The work ahead is less about technology choice and more about willingness to agree on shared rules—and to run them for years.