THE LIQUID COMPANY

On-Chain Corporate Governance

Regulated Securities from Incorporation to IPO
on the Liquid EVM

Zach Kelling

Satschel, Inc.

Version 1.0 — April 2026

ERC-3643 Securities · MPC Custody · FHE Privacy
Reg D/S/A+/CF/IPO · Rule 144 · On-Chain Cap Table
Dividends · Corporate Actions · USDL Settlement

Contents

Abstract

We define the **Liquid Company**—a new corporate form whose entire equity structure exists as smart contracts on a regulated, EVM-compatible blockchain. A Liquid Company's cap table is the blockchain itself: token balances are ownership, transfer restrictions are code, and corporate actions execute as transactions. The standard spans the full lifecycle of a corporation from formation through seed financing (SAFEs, convertible notes), exempt offerings (Regulation D 506(b)/506(c), Regulation S, Regulation A+, Regulation CF), and ultimately an initial public offering, all without migrating off-chain. We specify the smart contract architecture—`SecurityToken`, `ComplianceRegistry`, `DividendDistributor`, `CorporateActions`, and `OrderBook`—deployed on Liquid EVM (chain ID 8675309). Transfer restrictions mandated by SEC Rule 144, Regulation D investor limits, blue sky laws, and restrictive legend requirements are enforced at the EVM level: non-compliant transfers revert. Dividends distribute on-chain in USDL (1:1 USD-backed stablecoin). Corporate actions—stock splits, reverse splits, mergers, conversions—execute atomically. Privacy is preserved through fully homomorphic encryption (FHE) for confidential holdings and multi-party computation (MPC) for threshold decryption of regulatory disclosures. The three-pillar infrastructure—Alternative Trading System (ATS) for order matching, Broker-Dealer (BD) for compliance and execution, Transfer Agent (TA) for cap table maintenance—maps one-to-one onto SEC and FINRA regulatory requirements. We compare the Liquid Company against traditional corporate structures across 20 dimensions and present a complete worked example: a company proceeding from Delaware incorporation to IPO entirely on-chain.

Introduction

Corporate formation in the United States has not changed in substance since the early twentieth century. A founder files a certificate of incorporation with a state secretary (typically Delaware), receives a paper charter, issues paper stock certificates, and maintains a paper cap table in a spreadsheet or a SaaS product that is, at bottom, a database with no cryptographic guarantees. Every subsequent equity event—a SAFE conversion, an option grant, a secondary sale, a dividend payment, a stock split—requires manual reconciliation between the company's records, the transfer agent's records, and the broker-dealer's records. Discrepancies are common. Resolution is slow. Costs are high.

The consequences compound at scale:

- **Private securities are illiquid.** Transfer restrictions under Rule 144 [12] require a 6–12 month holding period, but the actual impediment is operational: finding a buyer, verifying accreditation, executing legal paperwork, and updating the cap table takes 30–90 days.
- **Cap tables diverge from reality.** A 2023 Carta study found that 23% of cap tables contain errors discovered only during a financing round or M&A due diligence [20].
- **Corporate actions are expensive.** Stock splits require board approval, transfer agent coordination, broker notification, and shareholder communication—a process that takes weeks and costs $50,000–$200,000 in legal and administrative fees for a private company.
- **Regulatory compliance is retroactive.** Companies discover compliance failures (e.g., exceeding the 2,000-shareholder threshold under Section 12(g) [18], or violating Reg D investor limits) after the fact, when remediation is costly.

We propose the **Liquid Company**: a corporate entity whose equity structure is defined entirely by smart contracts on Liquid EVM. The cap table is not a database that mirrors the blockchain—

the cap table *is* the blockchain. Token balances are legal ownership. Transfer restrictions are enforced by code, not by legal opinions. Corporate actions are atomic transactions, not multi-week workflows.

This paper makes the following contributions:

1. A formal specification of the Liquid Company standard, covering formation, financing, governance, and dissolution.
2. A smart contract architecture for securities issuance, compliance, dividends, and corporate actions.
3. A regulatory mapping from each SEC, FINRA, and state blue sky requirement to its on-chain enforcement mechanism.
4. A privacy model using FHE for confidential holdings and MPC for threshold decryption.
5. A complete worked example of a company proceeding from incorporation to IPO.

The system builds on our prior work in on-chain settlement [1], the Liquid Securities token standard [2], regulated ATS architecture [3], MPC custody [4], FHE dark pools [5], USDL stablecoin design [6], and formal verification of exchange invariants [7].

The Problem: Paper-Based Corporations

Structural Deficiencies

The modern corporation rests on a stack of paper abstractions maintained by mutually distrusting intermediaries. A private company's equity lifecycle involves at minimum:

Event	Intermediaries	Typical Duration
Incorporation	State, law firm	1–5 days
SAFE issuance	Law firm, cap table SaaS	1–3 days
SAFE → equity conversion	Law firm, transfer agent, cap table SaaS	2–4 weeks
Reg D 506(c) offering	BD, law firm, KYC provider, escrow	3–6 months
Secondary sale (Rule 144)	BD, transfer agent, legal opinion, buyer's counsel	30–90 days
Dividend payment	Transfer agent, bank, tax advisor	2–6 weeks
Stock split	Board, transfer agent, BD notification, state filing	2–4 weeks
IPO	Underwriter, SEC review, FINRA, DTC, transfer agent	6–12 months

Each step introduces delay, cost, and the possibility of error. The root cause is that ownership records exist in multiple disconnected systems—the company's cap table, the transfer agent's ledger, the broker-dealer's books, and the DTC's records—with no single source of truth.

The Cap Table Problem

A cap table is a mapping from shareholders to shares. In a traditional company, this mapping exists in at least three places:

1. The company's internal records (typically Carta, Pulley, or a spreadsheet).
2. The transfer agent's shareholder register (the legal record of ownership).
3. Each broker-dealer's customer account records (for street-name holdings).

These three copies diverge. Option exercises are recorded in the company's system but take days to propagate to the transfer agent. Secondary sales settle in the broker's books but may

not update the company's cap table for weeks. The result: at any given moment, no single party has an accurate view of who owns what.

On-chain, the cap table is a `balanceOf` mapping. There is one copy. It updates atomically. Every participant reads the same state.

The Compliance Problem

SEC regulations impose transfer restrictions that must be enforced at the point of transfer. In a paper-based system, enforcement depends on legal opinions: a securities lawyer reviews the proposed transfer, confirms the holding period has elapsed, verifies accreditation, and issues a letter permitting the transfer agent to update the register.

This process is:

- **Slow**: 2–6 weeks per transfer.
- **Expensive**: $5,000–$25,000 in legal fees per transaction.
- **Error-prone**: the lawyer relies on representations from the seller, which may be inaccurate.
- **Retroactive**: violations are discovered after the transfer, when unwinding is difficult.

On-chain, compliance is checked before the transfer executes. If the holding period has not elapsed, the transaction reverts. If the buyer is not KYC-verified, the transaction reverts. If the investor limit would be exceeded, the transaction reverts. The legal opinion is replaced by a compliance module that runs in constant time.

The Liquid Company Standard

Definition 1 (Liquid Company)**.** *A* Liquid Company *is a legal entity whose equity instruments are represented exclusively as ERC-20 tokens on Liquid EVM (chain ID 8675309), where:*

1. *The on-chain token balances constitute the legal record of share ownership.*
2. *All transfer restrictions required by applicable securities law are enforced by smart contract compliance modules.*
3. *All corporate actions (dividends, splits, mergers, conversions) execute as on-chain transactions.*
4. *A SEC-registered Transfer Agent maintains the on-chain shareholder register.*
5. *Trading occurs through a SEC-registered Alternative Trading System.*
6. *Compliance verification is performed by a FINRA-registered Broker-Dealer.*

Legal Foundation

The Liquid Company standard does not require new legislation. It operates within existing corporate law:

- **Delaware General Corporation Law § 224** permits corporations to maintain stock ledgers "by any electronic means." An EVM address-to-balance mapping satisfies this requirement.
- **UCC Article 8** recognizes "uncertificated securities" held in book-entry form. ERC-20 tokens are book-entry securities where the "book" is a blockchain.
- **SEC Rule 17a-4** requires broker-dealers to preserve records in non-rewritable, non-erasable format. An append-only blockchain satisfies this by construction.

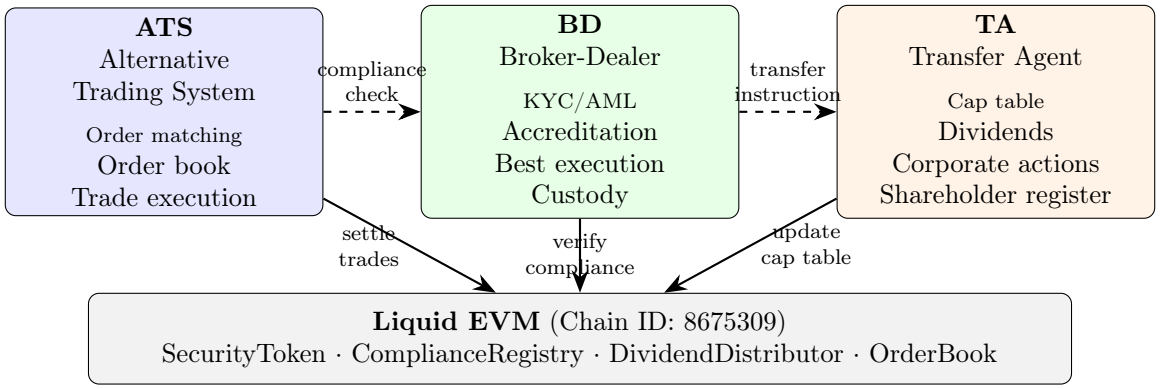
- **Wyoming Decentralized Autonomous Organization Supplement (2021)** explicitly permits blockchain-based corporate governance, providing a statutory framework for on-chain companies.

Design Principles

1. **The blockchain is the cap table.** There is no off-chain database that is "the real" cap table. Token balances are ownership. Period.
2. **Compliance is preventive, not reactive.** Transfer restrictions are checked before execution. Non-compliant transfers revert.
3. **Corporate actions are atomic.** A stock split updates all balances in a single transaction. There is no "processing period."
4. **Privacy by default.** FHE ciphertexts conceal holding amounts from all parties except the holder and authorized regulators.
5. **Regulatory equivalence.** Every SEC/FINRA requirement has a corresponding on-chain mechanism. The standard does not claim exemption from regulation—it claims superior enforcement of regulation.

Three-Pillar Architecture

The Liquid Company is operated by three SEC/FINRA-regulated entities, each responsible for a distinct function. This separation is not a design choice—it is mandated by regulation.



Alternative Trading System (ATS)

The ATS is registered with the SEC under Regulation ATS (17 CFR 242.300–303) [17]. Its on-chain responsibilities:

- Maintain the `OrderBook` contract for each listed SecurityToken.
- Match buy and sell orders using price-time priority.
- Settle matched trades atomically: transfer SecurityTokens from seller to buyer, USDL from buyer to seller.
- Report all trades to the compliance module for post-trade surveillance.

The ATS engine uses the Liquid DEX matching engine [9], which achieves 434 million operations per second on GPU-accelerated hardware, ensuring institutional-grade throughput.

Broker-Dealer (BD)

The BD is registered with FINRA and the SEC under Section 15(b) of the Securities Exchange Act of 1934. Its responsibilities:

- Perform KYC/AML verification on all investors before granting access to the `ComplianceRegistry`.
- Verify accreditation for Reg D offerings (income, net worth, or professional certification).
- Enforce best execution obligations (FINRA Rule 5310) through smart order routing across 16 execution venues [3].
- Maintain MPC custody wallets for each investor (2-of-3 threshold, non-custodial) [4].
- File regulatory reports: FINRA OATS, ATS-N, CAT (Consolidated Audit Trail).

Transfer Agent (TA)

The TA is registered with the SEC under Section 17A of the Securities Exchange Act of 1934. Its responsibilities:

- Deploy and administer `SecurityToken` contracts for each issuer.
- Maintain the `ComplianceRegistry`: add/remove verified investors, set lockup periods, enforce legends.
- Execute corporate actions: dividends, stock splits, reverse splits, mergers, conversions.
- Provide the official shareholder list for proxy voting and annual meetings.
- Issue tax documents (1099-DIV, 1099-B) based on on-chain transaction history.

Regulatory Separation

Entity	Regulator	On-Chain Role
ATS	SEC (Reg ATS)	`OrderBook` operator, trade settlement
BD	SEC + FINRA	`ComplianceRegistry` writer (KYC, accreditation), custody
TA	SEC (§ 17A)	`SecurityToken` admin, dividends, corporate actions

No single entity has unilateral control. The ATS cannot execute trades for unverified investors (the `ComplianceRegistry` rejects them). The BD cannot move shares (only the TA can mint/burn). The TA cannot trade (only the ATS can match orders). This separation is enforced by smart contract access control, not by policy.

Smart Contract Architecture

Contract Hierarchy

```
LiquidCompany (Governance Root)
  |-- SecurityToken[] One per share class
  | |-- ERC-20 (balanceOf, transfer, approve)
  | |-- ERC-3643 (T-REX compliance hooks)
  | |-- ERC-1404 (transfer restriction codes)
  | |-- ERC20Votes (governance delegation)
  |-- ComplianceRegistry Shared across all tokens
  | |-- IdentityRegistry KYC/AML identity claims
  | |-- JurisdictionModule Country-level restrictions
  | |-- AccreditationModule Investor qualification
  | |-- LockupModule Rule 144 holding periods
  | |-- MaxHoldersModule Reg D/CF investor limits
  | |-- LegendModule Restrictive legend tracking
```

```
  |-- DividendDistributor Per-token dividend payments
  |-- CorporateActions Splits, mergers, conversions
  |-- OrderBook Per-token trading venue
  |-- GovernanceRouter Proposal + voting
  |-- SAFEVault Pre-equity instruments
  |-- USDLSettlement USDL mint/burn for settlement
```

SecurityToken

Each share class of a Liquid Company is a distinct `SecurityToken` contract. The contract extends ERC-20 with compliance hooks from ERC-3643 and restriction codes from ERC-1404.

```
contract SecurityToken is ERC20, ERC3643, ERC1404, ERC20Votes {
    ComplianceRegistry public compliance;
    address public transferAgent; // Only TA can mint/burn

    modifier onlyTA() {
        require(msg.sender == transferAgent, "NOT_TA");
        _;
    }

    function transfer(address to, uint256 amount)
        public override returns (bool)
    {
        uint8 code = compliance.checkTransfer(msg.sender, to, amount);
        require(code == 0, compliance.messageForRestriction(code));
        return super.transfer(to, amount);
    }

    function mint(address to, uint256 amount) external onlyTA {
        uint8 code = compliance.checkMint(to, amount);
        require(code == 0, compliance.messageForRestriction(code));
        _mint(to, amount);
    }

    function burn(address from, uint256 amount) external onlyTA {
        _burn(from, amount);
    }

    function forcedTransfer(
        address from, address to, uint256 amount
    ) external onlyTA {
        // Court orders, escheatment, regulatory seizure
        _transfer(from, to, amount);
    }
}
```

ComplianceRegistry

The `ComplianceRegistry` is a modular compliance engine shared across all SecurityTokens of a Liquid Company. Each module implements a single check. Modules are composable: a transfer must pass all active modules.

```
contract ComplianceRegistry {
    IComplianceModule[] public modules;

    function checkTransfer(
```

```
        address from, address to, uint256 amount
    ) external view returns (uint8) {
        for (uint i = 0; i < modules.length; i++) {
            uint8 code = modules[i].checkTransfer(from, to, amount);
            if (code != 0) return code;
        }
        return 0; // All checks passed
    }
}
```

The compliance modules are specified in detail in the Liquid Securities Standard [2].

DividendDistributor

Dividends are paid in USDL [6]. The distributor uses a snapshot mechanism: the TA takes a snapshot of token balances at the record date, then distributes pro-rata.

```
contract DividendDistributor {
    struct Dividend {
        uint256 snapshotId; // ERC20Snapshot at record date
        uint256 totalAmount; // Total USDL to distribute
        uint256 totalShares; // Total shares at snapshot
        mapping(address => bool) claimed;
    }

    function declareDividend(
        address token, uint256 amount, uint256 recordDate
    ) external onlyTA {
        uint256 snapId = SecurityToken(token).snapshot();
        // Transfer USDL from issuer to this contract
        USDL.transferFrom(msg.sender, address(this), amount);
        // Record dividend
        dividends[token].push(Dividend(snapId, amount, ...));
    }

    function claimDividend(address token, uint256 index)
        external
    {
        Dividend storage d = dividends[token][index];
        require(!d.claimed[msg.sender], "ALREADY_CLAIMED");
        uint256 shares = SecurityToken(token)
            .balanceOfAt(msg.sender, d.snapshotId);
        uint256 payout = (shares * d.totalAmount) / d.totalShares;
        d.claimed[msg.sender] = true;
        USDL.transfer(msg.sender, payout);
    }
}
```

CorporateActions

The `CorporateActions` contract executes stock splits, reverse splits, mergers, and share class conversions. Each action is atomic.

```
contract CorporateActions {
    function stockSplit(
        address token, uint256 numerator, uint256 denominator
    ) external onlyTA {
```

```
        // For a 2:1 split: numerator=2, denominator=1
        address[] memory holders = getHolders(token);
        for (uint i = 0; i < holders.length; i++) {
            uint256 current = SecurityToken(token)
                .balanceOf(holders[i]);
            uint256 newBal = (current * numerator) / denominator;
            uint256 delta = newBal - current;
            if (delta > 0) {
                SecurityToken(token).mint(holders[i], delta);
            }
        }
    }

    function merger(
        address sourceToken, // Acquired company
        address targetToken, // Acquiring company
        uint256 ratio_num, // Conversion ratio
        uint256 ratio_den
    ) external onlyTA {
        address[] memory holders = getHolders(sourceToken);
        for (uint i = 0; i < holders.length; i++) {
            uint256 srcBal = SecurityToken(sourceToken)
                .balanceOf(holders[i]);
            uint256 tgtAmount = (srcBal * ratio_num) / ratio_den;
            SecurityToken(sourceToken).burn(holders[i], srcBal);
            SecurityToken(targetToken).mint(holders[i], tgtAmount);
        }
    }
}
```

OrderBook

The on-chain `OrderBook` provides a CLOB (Central Limit Order Book) for each SecurityToken. In practice, the matching engine runs off-chain for performance (the Liquid DEX engine [9]), and the `OrderBook` contract serves as the settlement layer.

```
contract OrderBook {
    function settleTrade(
        address buyer,
        address seller,
        address token,
        uint256 shares,
        uint256 usdlAmount
    ) external onlyATS {
        // Compliance check
        uint8 code = SecurityToken(token).compliance()
            .checkTransfer(seller, buyer, shares);
        require(code == 0, "COMPLIANCE_FAIL");
        // Atomic DVP: shares and USDL move simultaneously
        SecurityToken(token)
            .transferFrom(seller, buyer, shares);
        USDL.transferFrom(buyer, seller, usdlAmount);
        emit TradeSettled(buyer, seller, token, shares, usdlAmount);
    }
}
```

Settlement is delivery-versus-payment (DVP): shares and USDL move atomically in a single

transaction. There is no counterparty risk. There is no T+1 or T+2 delay.

SAFEVault

Pre-equity instruments (SAFEs, convertible notes) are represented as on-chain contracts that automatically convert to SecurityTokens upon a qualifying event.

```
contract SAFEVault {
    struct SAFE {
        address investor;
        uint256 amount; // USDL invested
        uint256 valuationCap; // Cap for conversion
        uint256 discount; // Discount rate (basis points)
        bool converted;
    }

    function convert(
        address token, uint256 pricePerShare
    ) external onlyTA {
        for (uint i = 0; i < safes.length; i++) {
            SAFE storage s = safes[i];
            if (s.converted) continue;
            uint256 effectivePrice = min(
                pricePerShare * (10000 - s.discount) / 10000,
                s.valuationCap > 0
                    ? s.valuationCap / totalShares
                    : pricePerShare
            );
            uint256 shares = s.amount / effectivePrice;
            SecurityToken(token).mint(s.investor, shares);
            s.converted = true;
        }
    }
}
```

Liquid Company Lifecycle

Phase 1: Formation

1. Incorporate in Delaware (or Wyoming for DAO Supplement).
2. Amend certificate of incorporation to specify that the "stock ledger" is maintained on Liquid EVM per DGCL § 224.
3. Deploy the `LiquidCompany` governance root contract.
4. Deploy a `SecurityToken` for common stock (e.g., "ACME-A" with 10,000,000 authorized shares).
5. Configure the `ComplianceRegistry` with initial modules (jurisdiction, accreditation).
6. Mint founder shares to founder MPC wallets.
7. Appoint the TA as the `transferAgent` on the SecurityToken contract.

On-chain, this is seven transactions. Elapsed time: under 60 seconds. Cost: approximately $50 in gas at current Liquid EVM gas prices (subsidized for formation).

Phase 2: Seed Round (SAFEs and Convertible Notes)

Founders issue SAFEs or convertible notes to angel investors:

1. Investor completes KYC through the BD; BD writes the investor's identity to the `ComplianceRegistry`.
2. Investor deposits USD via ACH/wire; USDL is minted to their MPC wallet.
3. Investor sends USDL to the `SAFEVault` contract, receiving a SAFE receipt NFT.
4. Upon a qualifying event (priced round), the TA calls `SAFEVault.convert()`, which mints SecurityTokens to all SAFE holders at the appropriate price.

No legal opinion is required for the conversion. The SAFEVault contract encodes the valuation cap, discount, and conversion mechanics. The TA verifies that the priced round is a qualifying event and triggers the conversion.

Phase 3: Series A (Regulation D 506(c))

1. Deploy a new `SecurityToken` for Series A preferred stock ("ACME-PA").
2. Configure the `AccreditationModule` to require accredited investor status for all holders.
3. Configure the `MaxHoldersModule` (no limit for 506(c), but useful for 506(b) offerings).
4. Configure the `LockupModule` with a 12-month Rule 144 holding period.
5. The BD verifies accreditation for each investor and writes claims to the `IdentityRegistry`.
6. Investors deposit USDL; the TA mints Series A tokens to verified investors.
7. File Form D with the SEC (off-chain, but data is sourced from on-chain records).
8. SAFEVault converts outstanding SAFEs at the Series A price.

Phase 4: Series B and Growth

Subsequent rounds follow the same pattern. Each round may use a different exemption:

Exemption	**Key Constraint**	**On-Chain Enforcement**
Reg D 506(b)	$\leq$35 non-accredited	`MaxHoldersModule` counter
Reg D 506(c)	All accredited, verified	`AccreditationModule` check
Reg S	Non-US persons only	`JurisdictionModule`: country $\neq$ US
Reg A+ (Tier 2)	\$75M max offering	Cumulative amount tracker in module
Reg CF	\$5M max, investment limits	Per-investor limit module

The `ComplianceRegistry` is reconfigured for each offering by adding or removing modules. Old modules remain in effect for previously issued securities. A single company can have multiple active offerings under different exemptions simultaneously.

Phase 5: Secondary Trading

Once securities are issued, holders can trade on the ATS:

1. Seller places a sell order on the `OrderBook` (via the ATS interface).
2. Buyer places a buy order.
3. The Liquid DEX matching engine matches orders by price-time priority.
4. Settlement: `OrderBook.settleTrade()` executes DVP atomically.
5. The `LockupModule` checks Rule 144 holding periods; if the seller's tokens are still restricted, the trade reverts.

Secondary trading provides liquidity that does not exist for traditional private securities. The restriction is regulatory (holding periods, accreditation), not operational.

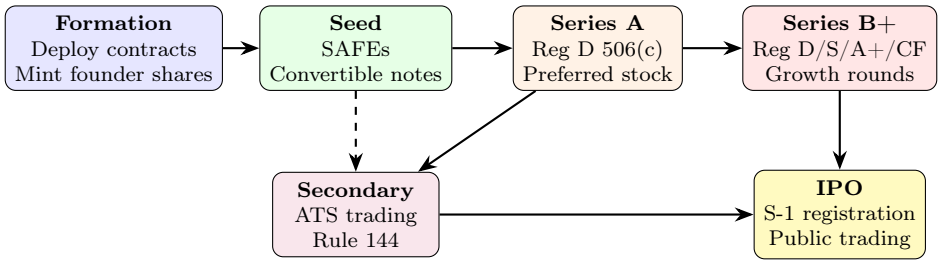
Phase 6: IPO

The Liquid Company can go public without migrating to a different infrastructure:

1. File S-1 registration statement with the SEC.
2. Upon effectiveness, update the `ComplianceRegistry` to remove the `AccreditationModule` from the common stock SecurityToken (public shares do not require accreditation).
3. The `LockupModule` applies a 180-day lockup for insider shares (standard IPO lockup).
4. List the SecurityToken on the ATS for public trading.
5. USDL settlement provides T+0 (instant) settlement, versus T+1 for traditional exchanges.
6. DTC bridge: for investors who prefer traditional brokerage accounts, a DTC-eligible master omnibus account holds SecurityTokens and mirrors them as DTC-eligible shares.

The on-chain infrastructure does not change. The only change is the compliance configuration: from "accredited investors only" to "any KYC-verified person."

Lifecycle Diagram



Regulatory Mapping

Every SEC and FINRA requirement maps to a specific on-chain mechanism. The Liquid Company does not claim exemption from regulation. It claims *superior enforcement* of regulation.

SEC Requirements

Requirement	Citation	On-Chain Enforcement
Transfer restrictions	Rule 144 [12]	`LockupModule`: 6/12-month hold, volume limits, manner of sale
Accredited investor verification	Reg D 506(c) [13]	`AccreditationModule`: claim from trusted KYC issuer
Non-accredited investor limits	Reg D 506(b) [13]	`MaxHoldersModule`: counter ≤ 35
Non-US persons only	Reg S [14]	`JurisdictionModule`: country $\neq$ US
Offering amount limits	Reg A+ [15], Reg CF	Cumulative tracker in compliance module
Per-investor limits	Reg CF [16]	Per-investor amount cap in compliance module
Shareholder threshold	Section 12(g) [18]	`MaxHoldersModule`: alert at 1,900; block at 2,000

Record keeping	Rule 17a-4	Blockchain: immutable, append-only, timestamped
Anti-fraud	Rule 10b-5	ATS surveillance: wash trading detection, pre-trade checks
Beneficial ownership	Schedule 13D/13G	Automatic filing trigger at 5% threshold
Insider reporting	Section 16	Automatic Form 4 generation on insider trades

FINRA Requirements

Requirement	Citation	On-Chain Enforcement
KYC/AML	FINRA Rule 2090	`IdentityRegistry`: verified claims required for all participants
Suitability	FINRA Rule 2111	BD checks before writing to `ComplianceRegistry`
Best execution	FINRA Rule 5310	Smart order routing across 16 venues [3]
Order handling	FINRA Rule 5310	Price-time priority in matching engine
ATS-N reporting	Reg ATS [17]	On-chain trade data exported to FINRA format
CAT reporting	FINRA Rule 6830	Every order/trade has unique ID traceable on-chain

State Blue Sky Laws

State securities laws ("blue sky laws") impose additional requirements, typically notice filings and fees. The `JurisdictionModule` can be configured per state:

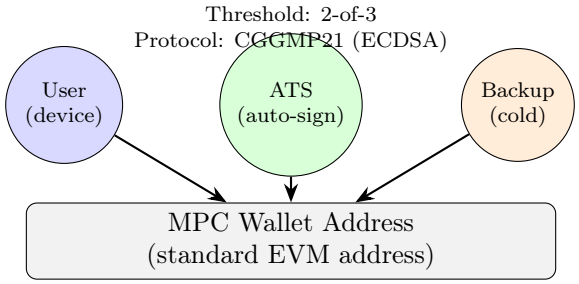
```
// Block transfers to residents of states where no
// notice filing has been made
mapping(bytes2 => bool) public stateCleared; // "CA", "NY", etc.

function checkTransfer(address, address to, uint256)
    external view returns (uint8)
{
    bytes2 state = identityRegistry.getState(to);
    if (!stateCleared[state]) return 20; // STATE_NOT_CLEARED
    return 0;
}
```

Non-Custodial Custody and Privacy

MPC Wallets

Every investor in a Liquid Company holds securities in a 2-of-3 MPC wallet, as described in our MPC custody paper [4]:



The ATS operator holds exactly one shard and therefore cannot sign unilaterally. This satisfies the non-custodial requirement: the operator never has the ability to move user funds. Passkey binding (FIDO2/WebAuthn) ensures the user's shard is hardware-bound.

FHE for Confidential Holdings

Public cap tables create a privacy problem: competitors, employees, and the general public can see exactly how much equity each address holds. Fully homomorphic encryption (FHE) solves this, as described in our FHE dark pools paper [5].

The FHE privacy model for the Liquid Company:

1. **Encrypted balances.** Token balances are stored as FHE ciphertexts on-chain. The `balanceOf` function returns a ciphertext that only the holder can decrypt.
2. **Encrypted transfers.** Transfer amounts are encrypted. The compliance module operates on encrypted values using homomorphic comparison (e.g., "is the encrypted balance $\geq$ the encrypted transfer amount?").
3. **Threshold decryption for regulators.** The SEC or FINRA can request decryption of specific balances via a threshold decryption protocol (3-of-5 MPC among authorized regulatory nodes). Individual regulators cannot decrypt unilaterally.
4. **Audit-compatible.** The TA can generate a zero-knowledge proof that the total supply equals the sum of all encrypted balances, without revealing individual holdings.

Property 1 (Privacy)**.** *For any observer $\mathcal{O}$ who is not (a) the holder, (b) the TA, or (c) a quorum of ≥ 3 authorized regulatory nodes, the encrypted balance $Enc(b_i)$ is computationally indistinguishable from a random ciphertext, under the RLWE assumption.*

Post-Quantum Security

Liquid EVM includes 19 precompiles, of which 6 are NIST post-quantum standards [1]:

Precompile	Standard	Use Case
ML-DSA	FIPS 204	Post-quantum digital signatures
SLH-DSA	FIPS 205	Stateless hash-based signatures
ML-KEM	FIPS 203	Key encapsulation (FHE key distribution)
FROST	Draft	Threshold EdDSA signatures
CGGMP21	Academic	Threshold ECDSA signatures
Ringtail	Lux	Lattice-based threshold signatures

A Liquid Company's securities are protected against quantum attacks from day one.

USDL: The Settlement Currency

All transactions in a Liquid Company settle in USDL, the on-chain dollar of Liquid EVM [6].

- **1:1 USD backed.** Each USDL is collateralized by $1 in a segregated bank account.
- **Mint on deposit.** Investor wires USD to the regulated custodian; USDL is minted to their MPC wallet.
- **Burn on withdrawal.** Investor redeems USDL; the custodian wires USD to their bank account; USDL is burned.
- **Atomic DVP.** Trade settlement transfers SecurityTokens and USDL in a single transaction. No counterparty risk.

USDL is the *only* dollar on Liquid EVM. There is no USDC, USDT, or LUSD. One dollar, one representation.

Comparison: Traditional vs. Liquid Company

Dimension	**Traditional Company**	**Liquid Company**
Cap table	Spreadsheet / SaaS	On-chain (`balanceOf`)
Source of truth	TA ledger (paper/DB)	Blockchain (single copy)
Share transfer speed	30–90 days	<60 seconds
Transfer cost	$5K–$25K legal fees	Gas fee (<$1)
Compliance enforcement	Legal opinion (retroactive)	Smart contract (preventive)
Dividend payment	2–6 weeks	1 transaction
Stock split	2–4 weeks, $50K–$200K	1 transaction, <$10 gas
Investor verification	Manual, per-transaction	On-chain claim, one-time
Rule 144 enforcement	Legal opinion per sale	`LockupModule` auto-check
Accreditation check	BD verifies, paper trail	`AccreditationModule` claim
Holder count tracking	Manual reconciliation	`MaxHoldersModule` counter
Audit trail	Scattered across parties	Blockchain (immutable, public)
Settlement	T+1 (public) / T+30–90 (private)	T+0 (atomic DVP)
Counterparty risk	Yes (settlement gap)	No (atomic)
Privacy	Opaque (too opaque)	FHE (selective disclosure)
Custody	Custodial (BD holds)	Non-custodial (MPC 2-of-3)
Post-quantum security	None	6 PQC precompiles
IPO migration	New infrastructure	Same contracts, new config
24/7 trading	No	Yes
Cross-border settlement	Days (wire transfer)	Seconds (USDL)

Worked Example: AcmeTech Inc.

We trace the complete lifecycle of a hypothetical company, AcmeTech Inc., from incorporation to IPO on Liquid EVM.

Day 1: Incorporation

Alice and Bob incorporate AcmeTech in Delaware. They amend the certificate of incorporation to specify Liquid EVM as the stock ledger per DGCL § 224.

```
Transactions:
  1. Deploy LiquidCompany("AcmeTech Inc.")
  2. Deploy SecurityToken("ACME", "AcmeTech Common", 10_000_000)
  3. Deploy ComplianceRegistry([JurisdictionModule, LockupModule])
  4. Mint 5,000,000 ACME to Alice (50%)
  5. Mint 5,000,000 ACME to Bob (50%)
  6. Set transferAgent = Satschel TA address

Gas cost: ~$30
Time: 42 seconds (7 blocks)
```

Month 3: Seed Round ($2M SAFE)

AcmeTech raises $2M from 8 angel investors via post-money SAFEs with a $10M cap and 20% discount.

```
Transactions:
  1. Deploy SAFEVault(cap=10_000_000, discount=2000)
  2. For each investor:
     a. BD writes KYC claim to IdentityRegistry
     b. Investor deposits USD -> USDL mint
     c. Investor sends USDL to SAFEVault
     d. SAFEVault emits SAFEIssued(investor, amount)

Total USDL deposited: 2,000,000
Time per investor: ~90 seconds
```

Month 14: Series A ($15M, Reg D 506(c))

Venture capital firm leads a $15M Series A at $50M pre-money valuation ($5.00/share).

```
Transactions:
  1. Deploy SecurityToken("ACME-PA", "AcmeTech Series A Preferred",
                          3_000_000)
  2. Configure AccreditationModule (all holders must be accredited)
  3. Configure LockupModule (12-month Rule 144 hold)
  4. For each investor:
     a. BD verifies accreditation, writes claim
     b. Investor deposits USDL
     c. TA mints ACME-PA tokens
  5. SAFEVault.convert(ACME, pricePerShare=5_000_000_000_000_000_000)
     -> 8 SAFE holders receive ACME common at effective price
  6. File Form D (off-chain, data sourced from on-chain)

SAFEs convert at: min($5.00 * 0.80, $10M/10M shares) = $1.00/share
SAFE holders receive: 2,000,000 ACME common shares
```

```
Total common outstanding: 12,000,000
```

Month 24: Secondary Trading

AcmeTech enables secondary trading of common stock on the ATS.

```
Prerequisite: 12-month Rule 144 holding period elapsed for
              seed SAFE conversions.

Trading:
  1. Seller places ask: 10,000 ACME @ $7.50
  2. Buyer places bid: 10,000 ACME @ $7.50
  3. Liquid DEX matches at $7.50
  4. OrderBook.settleTrade():
     - 10,000 ACME: seller -> buyer
     - 75,000 USDL: buyer -> seller
     - LockupModule confirms: holding period elapsed
     - AccreditationModule confirms: buyer is accredited
  5. Trade complete. Time: 6 seconds.
```

Month 36: Series B ($50M, Reg D 506(c) + Reg S)

AcmeTech raises $50M with $250M pre-money. US investors under Reg D 506(c), non-US investors under Reg S.

```
Transactions:
  1. Deploy SecurityToken("ACME-PB", "AcmeTech Series B Preferred",
                          2_000_000)
  2. ComplianceRegistry for ACME-PB:
     - US investors: AccreditationModule (506(c))
     - Non-US investors: JurisdictionModule (country != "US")
     - All: LockupModule (12-month hold)
  3. Mint ACME-PB to verified investors
  4. Stock split: 5:1 on ACME common
     CorporateActions.stockSplit(ACME, 5, 1)
     -> All common balances multiply by 5
     -> Total common: 60,000,000 shares
```

Month 48: Dividend

AcmeTech declares a $0.10/share dividend on common stock.

```
Transactions:
  1. TA calls DividendDistributor.declareDividend(
       ACME, 6_000_000 USDL, recordDate)
     -> Snapshot taken of all ACME balances
  2. Each holder calls claimDividend()
     -> Pro-rata USDL transferred from distributor

Total distributed: 6,000,000 USDL ($0.10 x 60,000,000 shares)
Gas per claim: ~$0.02
```

Month 60: IPO

AcmeTech files S-1 with the SEC. Upon effectiveness:

```
Transactions:
  1. Remove AccreditationModule from ACME common
     -> Any KYC-verified person can now hold ACME
  2. Configure LockupModule: 180-day insider lockup
     for Alice, Bob, and all pre-IPO investors
  3. TA mints 10,000,000 new ACME shares (IPO tranche)
  4. Underwriter distributes to IPO investors via ATS
  5. Public trading begins on ATS

Settlement: T+0 (atomic DVP in USDL)
Trading hours: 24/7/365
```

The same contracts, the same chain, the same wallets. No migration. No new infrastructure. The only change: a compliance module was removed.

On-Chain Governance

Voting

SecurityTokens extend ERC20Votes, providing checkpointed voting power with delegation. Governance proposals are submitted to the `GovernanceRouter`:

```
contract GovernanceRouter {
    struct Proposal {
        bytes32 id;
        address proposer;
        bytes[] actions; // Encoded function calls
        uint256 snapshotBlock; // Voting power snapshot
        uint256 deadline;
        uint256 forVotes;
        uint256 againstVotes;
        bool executed;
    }

    function propose(bytes[] calldata actions)
        external returns (bytes32)
    {
        require(
            getVotes(msg.sender) >= proposalThreshold(),
            "BELOW_THRESHOLD"
        );
        // Create proposal with current block as snapshot
    }

    function castVote(bytes32 proposalId, bool support)
        external
    {
        uint256 weight = getPastVotes(
            msg.sender,
            proposals[proposalId].snapshotBlock
        );
        // Record vote weighted by shares at snapshot
    }
}
```

Proxy Voting

For regulated securities, proxy voting follows SEC Rule 14a. The TA distributes proxy materials via on-chain events, and shareholders vote through the `GovernanceRouter`. Broker-held shares (street name) are voted by the BD on instruction from the beneficial owner.

Board Elections

Board seats are filled by shareholder vote. Cumulative voting, plurality voting, and majority voting are configurable per `GovernanceRouter` deployment.

Corporate Actions in Detail

Stock Splits and Reverse Splits

A k:1 stock split multiplies every holder's balance by k and divides the par value by k. The total market capitalization is unchanged. On-chain, this is a single transaction that iterates over all holders.

Property 2 (Split Invariant)**.** *For a k:1 split, $\forall$ holders h: $balance_{after}(h) = k \cdot balance_{before}(h)$, and $totalSupply_{after} = k \cdot totalSupply_{before}$.*

Reverse splits (1:k) reduce balances. Fractional shares are settled in USDL at the current market price.

Mergers and Acquisitions

When Company A acquires Company B at a ratio of r A-shares per B-share:

1. Shareholder vote on both sides (via `GovernanceRouter`).
2. Upon approval, `CorporateActions.merger()` burns all B-tokens and mints A-tokens at ratio r.
3. B's `SecurityToken` is permanently frozen (no further transfers).
4. Fractional shares are settled in USDL.

Conversions

Preferred stock converts to common stock per the terms encoded in the SecurityToken contract (conversion ratio, anti-dilution adjustments). Conversion can be:

- **Optional**: holder calls `convert()` at any time.
- **Mandatory**: TA triggers conversion upon a qualifying event (IPO, acquisition).
- **Automatic**: smart contract triggers conversion when conditions are met (e.g., stock price exceeds threshold for 20 consecutive trading days).

Buybacks and Tender Offers

The company deploys a `TenderOffer` contract specifying price, quantity, and duration. Shareholders tender by approving the contract to transfer their tokens. At expiration, the contract settles pro-rata if oversubscribed, returning USDL to the company and tokens to treasury.

Economic Analysis

Cost Comparison

Event	Traditional	Liquid Company
Incorporation + stock issuance	$5,000–$15,000	<$100
SAFE round (8 investors)	$10,000–$30,000	<$50
Series A (Reg D 506(c))	$100,000–$250,000	<$500
Secondary sale (per transaction)	$5,000–$25,000	<$1
Dividend payment (1,000 holders)	$15,000–$50,000	<$20
Stock split	$50,000–$200,000	<$10
IPO (excluding underwriting)	$1M–$5M admin/legal	<$1,000

The cost reduction is 3–5 orders of magnitude for most corporate actions. The primary remaining cost is regulatory (SEC filing fees, FINRA fees, auditor fees), which applies equally to both models.

Time Comparison

Event	Traditional	Liquid Company
Share transfer	30–90 days	<60 seconds
SAFE conversion	2–4 weeks	1 transaction
Dividend distribution	2–6 weeks	1 transaction
Stock split	2–4 weeks	1 transaction
Cap table reconciliation	Days–weeks	Real-time (no reconciliation)

Liquidity Premium

Private securities trade at a 20–40% illiquidity discount relative to comparable public securities [21]. The Liquid Company standard eliminates the operational component of this discount (transfer friction, settlement delay, counterparty risk). We estimate a residual discount of 5–10%, attributable solely to the smaller investor base and information asymmetry of private markets.

Security Analysis

Threat Model

We consider the following adversaries:

1. **Malicious insider**: An employee of the ATS, BD, or TA attempts unauthorized actions.
2. **Compromised shard**: An attacker obtains one MPC key shard.
3. **Regulatory coercion**: A government actor demands access to encrypted holdings.
4. **Quantum adversary**: An attacker with a cryptographically relevant quantum computer.

Mitigations

Threat	Mitigation
Malicious insider	Three-pillar separation: no single entity can trade, verify compliance, and update the cap table. Smart contract access control enforces separation.
Compromised shard	2-of-3 MPC: one shard is insufficient to sign. CGGMP21 key is never reconstructed [4].
Regulatory coercion	3-of-5 threshold decryption for FHE balances. No single regulator can decrypt. Compliant with lawful disclosure orders via quorum.
Quantum adversary	ML-DSA, SLH-DSA, ML-KEM precompiles for post-quantum signatures and key encapsulation [1].

Formal Verification

The core invariants of the Liquid Company standard—supply conservation, compliance gating, DVP atomicity, split correctness—have been formally verified in Lean 4, as described in our formal verification paper [7]. Key theorems:

Theorem 1 (Supply Conservation). *For any sequence of* `mint`*,* `burn`*, and* `transfer` *operations,* $\texttt{totalSupply} = \sum_{a \in Addresses} \texttt{balanceOf}(a)$.

Theorem 2 (Compliance Gating). *A* `transfer(from, to, amount)` *succeeds if and only if* $\forall m \in \texttt{modules} : m.\texttt{checkTransfer}(\texttt{from}, \texttt{to}, \texttt{amount}) = 0$.

Theorem 3 (DVP Atomicity). *In* `settleTrade(buyer, seller, token, shares, usdl)`*, either both the SecurityToken transfer and the USDL transfer succeed, or neither does.*

Related Work

Tokenized Securities Platforms

Securitize operates a SEC-registered transfer agent and ATS for digital securities. Securitize uses the DS Protocol, a proprietary token standard. Unlike the Liquid Company, Securitize maintains an off-chain cap table that is "mirrored" on-chain; the blockchain is not the source of truth.

tZERO (Overstock subsidiary) operates an ATS for digital securities on Ethereum. tZERO uses a centralized matching engine with Ethereum settlement. The key difference: tZERO's securities trade on public Ethereum, sharing block space with DeFi and NFTs. Liquid EVM is purpose-built for regulated securities.

Polymath / Polymesh built a purpose-built blockchain for securities (Polymesh). Polymesh uses a permissioned validator set and native compliance modules. However, Polymesh is not EVM-compatible, limiting tooling and developer access.

On-Chain Corporate Governance

Aragon provides on-chain governance tools for DAOs. Aragon's Governor contracts support proposal/vote/execute workflows. The Liquid Company's `GovernanceRouter` is structurally similar but adds compliance modules: only verified shareholders can vote, and voting power is proportional to registered shares (not freely transferable governance tokens).

Compound Governor (Bravo/Charlie) provides governance for DeFi protocols. The Liquid Company adapts this pattern for regulated securities, adding snapshot-based voting that respects transfer restrictions and SEC proxy rules.

Privacy-Preserving Finance

Penumbra provides private staking and DEX trading using zero-knowledge proofs. The Liquid Company uses FHE rather than ZKPs because FHE supports computation on encrypted data (required for compliance checks on encrypted balances), while ZKPs prove statements about private data without enabling computation.

Zama fhEVM provides FHE-native EVM execution. Liquid EVM's FHE precompiles are informed by fhEVM's approach but are implemented as native precompiles rather than a modified EVM, providing better performance and auditability.

Limitations and Future Work

Current Limitations

1. **Gas costs for large cap tables.** A stock split that iterates over 100,000 holders requires a high gas limit. Mitigation: batch processing across multiple transactions, or a Merkle-based claim pattern (similar to airdrops).
2. **FHE performance.** Fully homomorphic encryption is computationally expensive. Current FHE comparison operations take 50–200ms on Liquid EVM's precompiles. This is acceptable for compliance checks but not for high-frequency trading.
3. **Legal uncertainty.** While DGCL § 224 and UCC Article 8 support on-chain stock ledgers, no court has ruled on the legal status of EVM token balances as definitive ownership records. The first contested transfer will set precedent.
4. **DTC interoperability.** Institutional investors who require DTC-eligible securities need a bridge (omnibus account). This bridge reintroduces a trusted intermediary for that subset of holders.

Future Work

1. **Cross-chain mergers.** Two Liquid Companies on different EVM chains executing a merger via a compliance-preserving bridge [10].
2. **Automated regulatory filings.** On-chain events trigger automatic generation and submission of SEC forms (Form D, S-1 amendments, 13D/13G, Form 4).
3. **DAO-to-corporation conversion.** A DAO on public Ethereum converting its governance token to a Liquid Company SecurityToken, gaining regulatory compliance while preserving decentralized governance.
4. **International expansion.** Adapting the compliance module system for non-US jurisdictions: FCA (UK), BaFin (Germany), MAS (Singapore), JFSA (Japan).

Conclusion

The Liquid Company standard eliminates the paper-based infrastructure that has defined corporate equity management for over a century. By making the blockchain the cap table—not a mirror of the cap table, but the cap table itself—we achieve:

- **Single source of truth**: one copy of ownership records, updated atomically.
- **Preventive compliance**: transfer restrictions enforced before execution, not after.
- **Atomic corporate actions**: dividends, splits, and mergers in a single transaction.
- **T+0 settlement**: no counterparty risk, no settlement gap.

- **Privacy**: FHE for confidential holdings, MPC for threshold access.
- **Full lifecycle support**: from incorporation to IPO on the same infrastructure.

The system does not require new laws. It operates within existing SEC, FINRA, and state corporate law frameworks. It does not claim exemption from regulation. It claims—and delivers—superior enforcement of regulation through code.

The three-pillar architecture (ATS, BD, TA) maps one-to-one onto the regulatory structure that governs securities markets. Smart contract access control enforces the separation that regulation mandates. The result is a corporate form that is more liquid, more transparent to authorized parties, more private from unauthorized parties, and less expensive to operate than any traditional alternative.

A company incorporated today as a Liquid Company on Liquid EVM can raise its seed round next month, complete a Series A next year, trade on a secondary market the year after, and go public without ever changing its infrastructure, migrating its cap table, or reconciling its records. The equity is the token. The cap table is the chain. The company is liquid.

References

[1] Z. Kelling. Liquid EVM: On-chain securities settlement with post-quantum MPC custody. *Satschel, Inc. / Liquidity.io*, April 2026.

[2] Z. Kelling. Liquid Securities: Compliant digital securities on EVM with transfer restrictions and corporate actions. *Satschel, Inc. / Liquidity.io*, April 2026.

[3] Z. Kelling. Liquidity.io: A regulated alternative trading system for digital securities on EVM. *Satschel, Inc. / Liquidity.io*, April 2026.

[4] Z. Kelling. Non-custodial MPC custody for regulated securities: Passkey-bound threshold signatures on Liquid Threshold. *Satschel, Inc. / Liquidity.io*, April 2026.

[5] Z. Kelling. FHE dark pools: Encrypted order matching with threshold decryption and compliance. *Satschel, Inc. / Liquidity.io*, January 2025.

[6] Z. Kelling. USDL: Algorithmic USD stablecoin for securities settlement. *Satschel, Inc. / Liquidity.io*, November 2025.

[7] Z. Kelling. Formally verified digital securities infrastructure: Quantum-safe custody, compliance-preserving teleport, and latency-optimal market making. *Satschel, Inc. / Liquidity.io*, December 2025.

[8] Z. Kelling. LIQUID: Governance token economics for a regulated digital securities exchange. *Satschel, Inc. / Liquidity.io*, April 2026.

[9] Z. Kelling. Liquid DEX: High-performance on-chain order matching for regulated securities. *Satschel, Inc. / Liquidity.io*, April 2026.

[10] Z. Kelling. Liquid Bridge: Compliance-preserving cross-chain securities transfer. *Satschel, Inc. / Liquidity.io*, April 2026.

[11] Z. Kelling. Liquid Consensus: Quasar BFT for regulated settlement chains. *Satschel, Inc. / Liquidity.io*, April 2026.

[12] Securities and Exchange Commission. Rule 144: Selling restricted and control securities. 17 CFR 230.144, 1972 (amended 2008).

[13] Securities and Exchange Commission. Regulation D: Rules governing the limited offer and sale of securities. 17 CFR 230.500–508, 1982 (amended 2020).

[14] Securities and Exchange Commission. Regulation S: Rules governing offers and sales made outside the United States. 17 CFR 230.901–905, 1990.

[15] Securities and Exchange Commission. Regulation A: Conditional small issues exemption. 17 CFR 230.251–263, 2015 (amended 2020).

[16] Securities and Exchange Commission. Regulation Crowdfunding. 17 CFR 227.100–503, 2016 (amended 2021).

[17] Securities and Exchange Commission. Regulation ATS: 17 CFR 242.300–303. *Federal Register*, 1998.

[18] Securities and Exchange Commission. Section 12(g): Registration requirement for securities. Securities Exchange Act of 1934, § 12(g).

[19] J. Music, T. Musik, and R. Musik. ERC-3643: T-REX — Token for Regulated Exchanges. *Ethereum Improvement Proposals*, 2021.

[20] Carta, Inc. 2023 State of Private Markets Report. https://carta.com/blog/state-of-private-markets-2023/, 2023.

[21] M. Bajaj, D. J. Denis, S. P. Ferris, and A. Sarin. Firm value and marketability discounts. *Journal of Corporation Law*, 27:89–115, 2001.

[22] Delaware General Corporation Law. § 224: Form of records. Title 8, Chapter 1, Subchapter VII.

[23] Uniform Commercial Code. Article 8: Investment Securities. American Law Institute / Uniform Law Commission, 1994 (revised).

[24] State of Wyoming. Decentralized Autonomous Organization Supplement. Wyoming Statute § 17-31-101 through 17-31-116, 2021.

The Liquid Company standard is built on open source contracts from the Lux Network ecosystem. All upstream Lux repositories are open source (BSL-1.1, MIT, Apache-2.0).

Lux Platform Specs: https://github.com/luxfi/lps