Hashgraph

BEHIND THE LEDGER TECHNICAL SERIES: PART 1

# Transaction privacy in HashSphere: Atomic settlement without exposure

A technical paper for enterprise decision-makers on how HashSphere enables confidential value exchange across DvP, PvP, and private transfers.


Rohit Sinha
Head of Cryptography, Hashgraph

Pratay Mukherjee
Principal Researcher, Hashgraph

Amanda Clark
Senior Product Manager, Hashgraph

Kash Balhotra
Institutional Digital Assets Lead, Hashgraph

# Privacy is not a feature. It's actually four load-bearing pillars

HashSphere, a private network from Hashgraph powered by Hedera, is built on four compounding privacy properties that must coexist at every layer of the stack:

1. **Transaction confidentiality** — private by cryptographic enforcement, not policy
2. **Independent network design** — a sovereign perimeter, not a subnet on a public network
3. **Flexible validator and governance models** — not delegated to a third party
4. **Mathematically guaranteed fair ordering** — through the hashgraph consensus algorithm

Remove any one of these, and the guarantees of the others weaken. This paper focuses on the first and most foundational, transaction privacy, and how HashSphere enables atomic settlement across Delivery versus Payment (DvP), Payment versus Payment (PvP), and private transfers without exposing data to any party — including the network itself.

Subsequent papers in this series will address each of the remaining three pillars. With a final paper diving into how HashSphere uses a new cross-ledger protocol to enable interoperability between public and private networks, avoiding the "islands of trapped assets" problem inherent in closed network designs.

In summary, at Hashgraph we believe privacy and interoperability on a distributed ledger technology platform doesn't have to be a trade-off.

# The business problem

Regulated financial institutions operate under a set of obligations that make public, permissionless blockchain infrastructure fundamentally unsuitable for the settlement of real value. Regulators, market conduct rules, and data privacy frameworks, including Basel SCO60, MiFID II, Reg NMS, GDPR, GLBA, and the Bank Secrecy Act impose strict controls on who can see transaction data, when, and under what conditions. For example, the Bank Secrecy Act requires KYC and transaction monitoring. On a public, permissionless blockchain, transaction counterparties are pseudonymous wallet addresses, not verified legal identities.

On a public network like Hedera, transaction data is broadcast to every participant by design. Even where identities are anonymous, the transaction graph is visible, amounts are observable, and pattern analysis can reconstruct positions, strategies, and counterparty relationships that institutions are legally required to protect. Beyond compliance, there is a competitive dimension: a Tier-1 bank's FX flow, settlement timing, or treasury rebalancing activity is material non-public intelligence, and its exposure to peers on a shared network is not a theoretical risk. It is a direct liability.

A lot of research has been conducted on how best to introduce privacy into a public blockchain. What market participants actually need is not a privacy feature per se, they need to move value across counterparties, across asset classes, and across networks without exposing their institution's positions, strategies, or relationships to anyone who shouldn't see them.

A closed network without transaction-level confidentiality means every participant can still see every trade. A transaction privacy layer deployed on top of a shared or public base layer means the network perimeter is not truly controlled.

The right question is simpler: can Institution A and Institution B exchange value atomically, in both directions, without Institution C knowing what they traded, or how much?

## Segregated state models

In a conventional blockchain, a shared global state is maintained across all nodes, and the validity of every transaction is verifiable by every participant or node operator. Many private, permissioned blockchain networks abandon this model entirely. Instead of shared states, they use segregated states: each participant only ever sees the transactions in which they were directly involved. Privacy is achieved not through cryptography but through information withholding, you can't see what you were never sent.

The structural consequence of this design is centralization. Transactions are routed through a controller, which coordinates between participants, checks for double-spend, and sequences settlement. Critically, this coordinator can see all private transaction data. The network's privacy guarantee is not cryptographic, it is contractual and operational.

This breaks the core property that gives a blockchain its institutional value:

#### DECENTRALIZATION OF TRUST

On a true DLT, you do not need to trust any individual party. The math enforces correctness, as long as a sufficient fraction of nodes are honest. On some private, permissioned networks, that guarantee collapses. When a central operator routes, sequences, and validates transactions, the network's integrity is no longer cryptographic, it is contractual. If you already have a trusted intermediary at the center of your settlement infrastructure, you effectively have a database with extra steps, not a blockchain.

Finally, the segregated state approach means that participants only see their own transactions, but the controller sees them all.

## Glossary of terms

This paper uses the following atomic settlement terms which have been described in the glossary here.

- **DvP (Delivery versus Payment):** Settlement of two linked obligations: delivery of an asset and payment of cash. The transaction must be atomic — the asset is delivered only if cash pays, otherwise neither settles.
- **PvP (Payment versus Payment):** Atomic cash-for-cash settlement, often used in FX. Both currency legs settle together or not at all.
- **Atomic Settlement:** Both legs of every trade settle simultaneously, or neither does. This is not best-effort settlement, partial fill, or a sequence of dependent transactions that can fail mid-way. It is a binary, mathematically enforced commitment.
- **UUID (Universally Unique Identifier):** A transaction identifier agreed bilaterally between counterparties before trade. The atomic swap contract uses matching UUIDs to link and validate both legs of a transaction.

# HashSphere's solution: Privacy without compromise

HashSphere's approach inverts this entirely. Transaction privacy is enforced cryptographically through zkSNARKs and "encrypted" balance, so no party, including the network operator, can see transaction content. Transactions settle in under 3-5 seconds and are mathematically guaranteed with hashgraph consensus fair ordering.

The validator set is chosen by the stand-alone private network, which is not a subnet on a public network. There is no central routing party — the network chooses its own governance model. And the network is built on standard EVM infrastructure, compatible with the broader ecosystem of tooling, custody, and development talent.

Finally, by leveraging the Hashgraph-developed cross-ledger protocol (CLPR), interoperability exists across Spheres, the Hedera public network, and even other protocols.

## Addressing the three dimensions of transaction privacy

Transaction privacy has three distinct dimensions: **confidentiality, anonymity**, and **selective disclosure**. Most DLTs address one. Some address two. Very few address all three simultaneously, and fewer still do it without sacrificing cryptographic trust, introducing a centralized routing dependency, or creating a second system to operate.

Not every transaction requires the same level of privacy. HashSphere supports a spectrum across these core dimensions.

### Confidentiality

**Confidentiality** conceals the details of the transaction payload — the amounts, asset types, and resulting balances remain confidential. This applies to treasury rebalancing, payroll, and intercompany settlements where the participants may be known but the amounts must not be. An employer paying employees on a shared network is one example; the identities are not secret, but the salary amounts are.

Even when counterparty identities are obscured, transaction amounts, asset types, and resulting balances can reveal competitive intelligence that institutions cannot afford to expose.

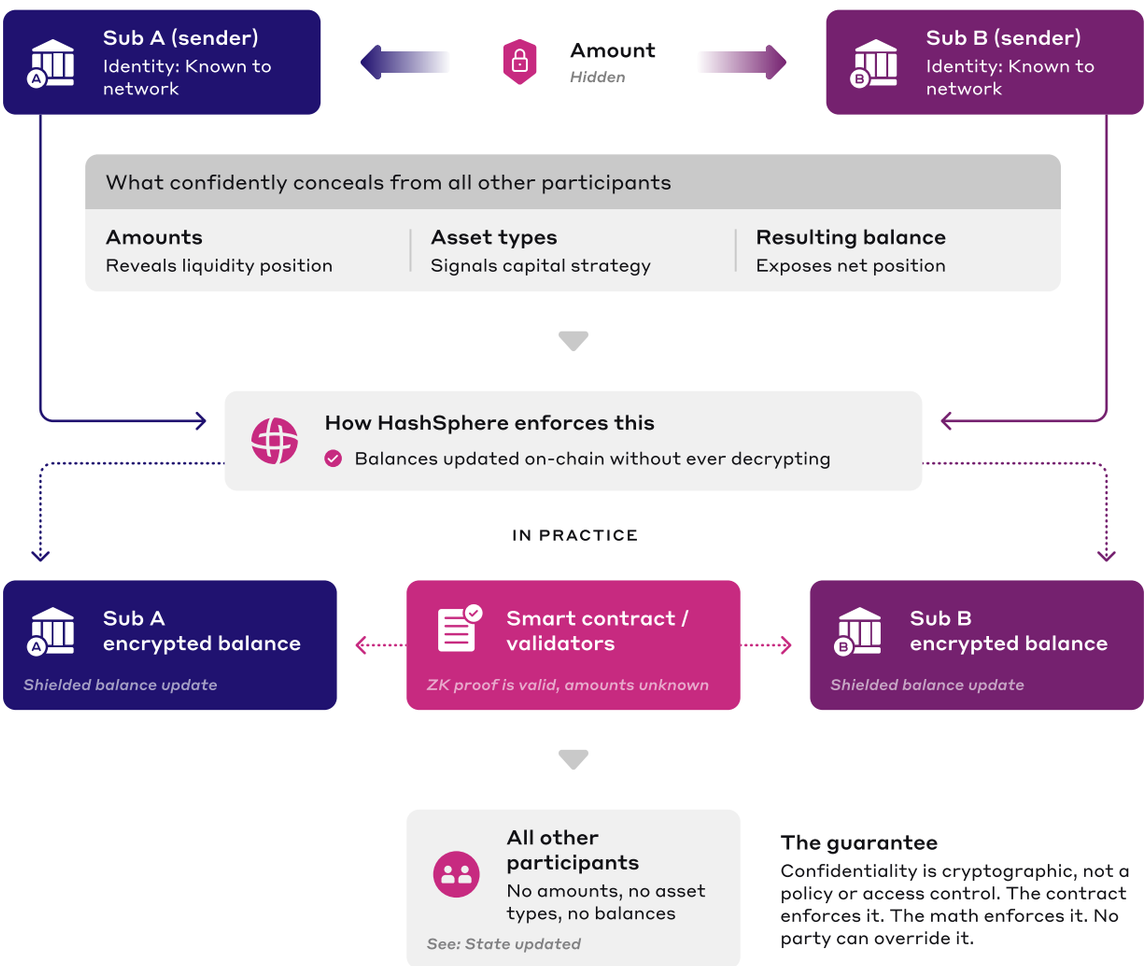
- Treasury rebalancing flows reveal liquidity positions.
- Payroll amounts expose commercial terms.
- Intercompany settlement sizes signal internal capital allocation strategies.

**Bob and Alice in practice:** A corporate treasury is moving stablecoins between subsidiaries on HashSphere. Both subsidiaries, Bob and Alice, are known participants on the network, but the amounts, asset types, and resulting balances must not be. No information about the transaction should be accessible to any third party present on the same network. Any leakage (however small) could allow an uninvolved participant to infer liquidity stress, capital reallocation, or upcoming market activity.

### HOW HASHSPHERE ACHIEVES THIS

**Private balance:** Encrypted balances are updated on-chain without the contract, the validator, or any other participant ever seeing the plaintext values. Bank A's balance decreases by 100 USDC. Bank B's balance increases by 100 USDC. The math is correct. The numbers are never exposed.
The contract enforces this property, it is not a policy or an access control, it is cryptographic.

### CONFIDENTIALITY — WHAT WAS TRANSACTED?

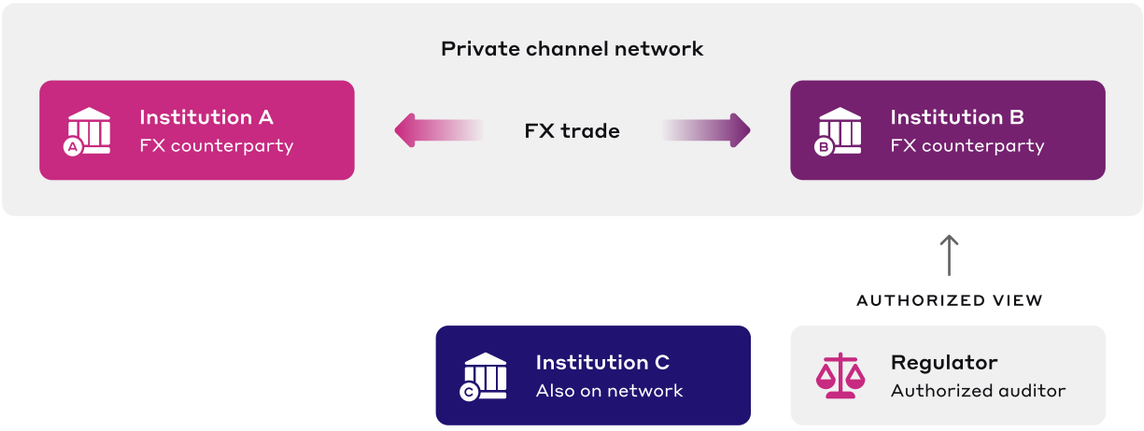


## Anonymity

**Anonymity** conceals who is participating in a transaction. This may be needed for cross-border payments, liquidity management, and price discovery scenarios where the participants themselves, not just the amounts, represent sensitive intelligence. A Tier-1 bank bidding in an intraday liquidity auction does not want competitors mapping its activity.

**Bob and Alice in practice:** Institution A and Institution B are executing a cross-border FX trade on HashSphere. Both are on the same network and want to transact privately. Neither wants Institution C — also on the network — to know that they are trading with each other, let alone when and how frequently. However, they do want to include an authorized regulator/auditor.

### HASHSPHERE NETWORK




### HOW HASHSPHERE ACHIEVES THIS

A form of cryptographic proof that combines ‘encryption’ with zero-knowledge verification; allowing one party to prove a statement is true without revealing any of the underlying information. Each party generates a proof that attests to the validity of their transaction: that they have sufficient funds, the parameters are correct, and value is conserved. The contract verifies this proof, confirming correctness, without ever seeing who sent, who received, or how much was transferred. Once funds are on-ramped, amounts and balances are cryptographically hidden from all uninvolved parties, however, full counterparty anonymity requires deposit shuffling, which unlinks deposits from private transactions. The size of the anonymity set is configurable, with throughput traded against the number of accounts included in each shuffle. Only the transacting parties, and an authorized regulator if designated, can see any information about the transaction.

## Selective disclosure

Cryptographically enforced, auditor-accessible read access. Any solution that cannot give a designated regulator access to specific transaction data on demand is not enterprise-ready. Critically, this access must be cryptographically enforced, not policy-enforced.

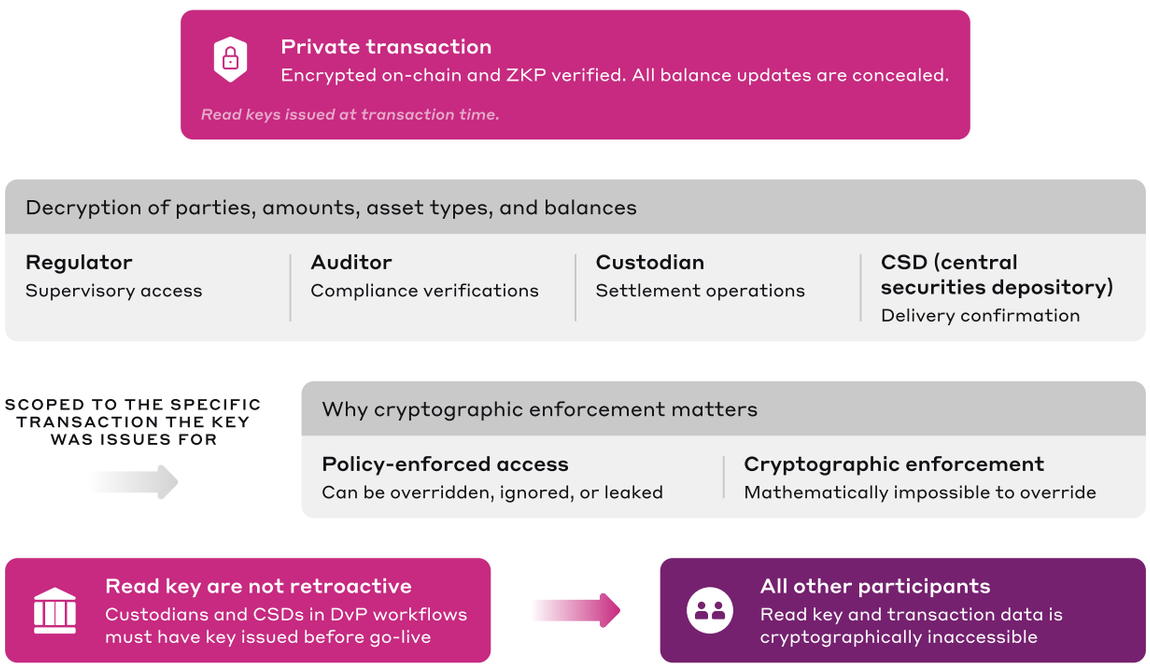
**Bob and Alice in practice:** Bank A and Bank B have completed a bond settlement on HashSphere. The transaction is private; amounts, identities, and asset types are cryptographically hidden from uninvolved parties. When a regulator opens a supervisory review three months later, there is no need to involve the operator or rely on a contractual obligation to produce records. The regulator was issued a read key at settlement, scoped to that transaction. They decrypt and verify the full record independently. Bank C sees nothing. The operator sees nothing. Only parties issued read keys, including the custodian, the CSD, and the regulator, can access the data, and only for the transactions they were authorized to see.

### HOW HASHSPHERE ACHIEVES THIS

HashSphere implements selective disclosure through cryptographic verification rather than a separate access layer. When a regulator or auditor requests proof of a specific transaction, the bank shares the relevant transaction details, which the auditor can verify directly against the on-chain record. Access is scoped to the specific transaction disclosed, nothing more is revealed. Institutions, particularly custodians and CSDs in DvP workflows, should plan disclosure procedures before going live.

### SELECTIVE DISCLOSURE

Cryptographically enforced read access for designated parties. Not optional. Not policy.



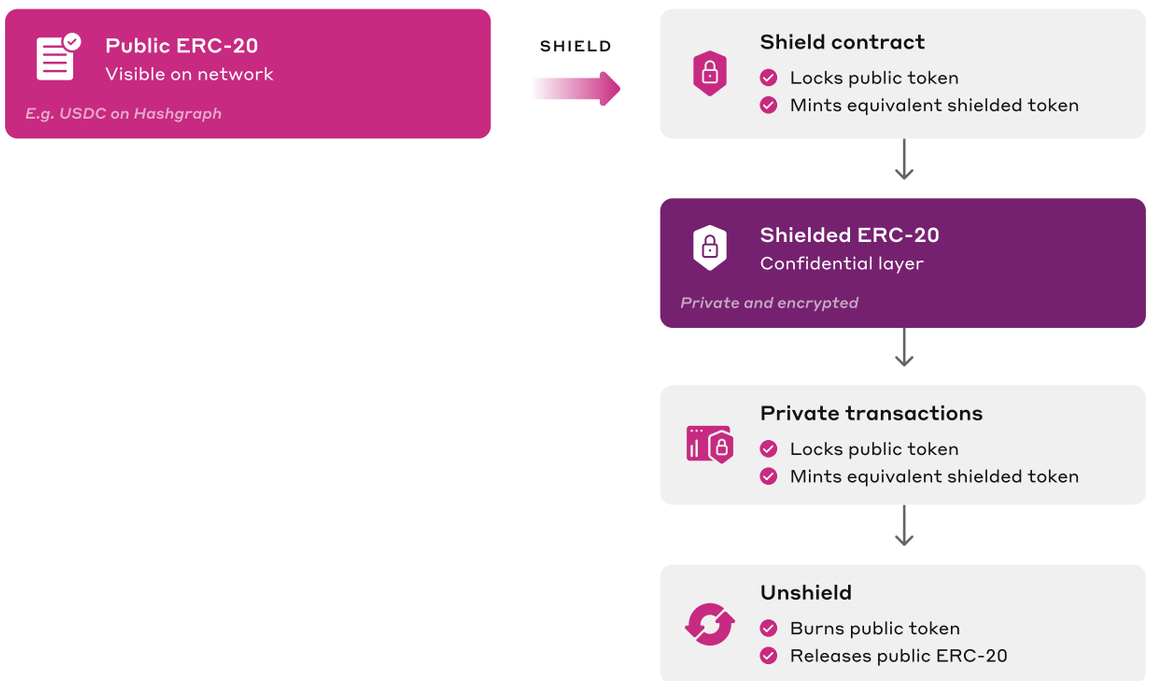
# HashSphere privacy uses ERC-20 token standard

HashSphere's architecture delivers against all three privacy dimensions simultaneously, using the ERC-20 token standard as the foundation and cryptographic mechanisms operating in concert.

## ERC-20 shield and unshield contracts

ERC-20 Shield and Unshield Contracts are the entry and exit point of the HashSphere privacy layer. A public ERC-20 token is locked in the Shield contract and an equivalent shielded ERC-20 token is minted in its place. When the institution exits the privacy layer, the shielded token is burned and the original public token is released.

**ERC-20 SHIELD AND UNSHIELD — PRIVACY LAYER ENTRY AND EXIT**

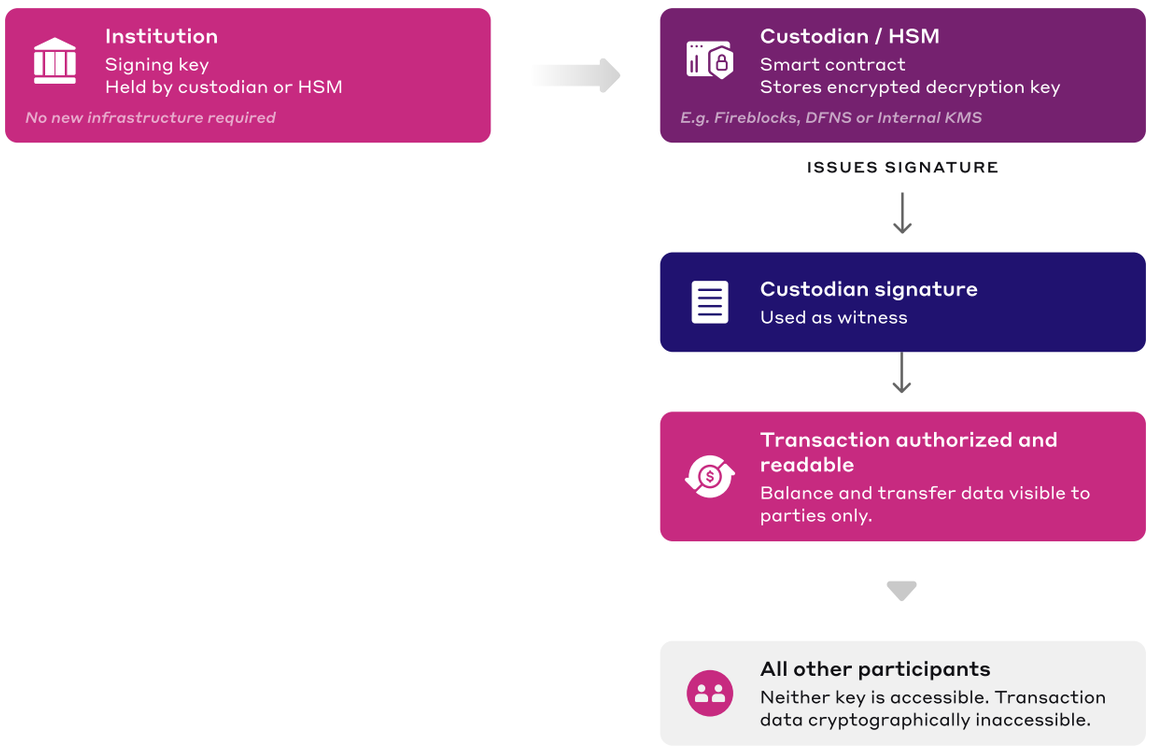



The token standard does not change, the visibility of what happens to it does. This is a well-understood, auditable mechanism built on standard token infrastructure. The shielding and unshielding steps themselves are not private; they are visible on-chain. All subsequent transactions within the privacy layer are not.

## Institution key

Each institution on HashSphere holds a signing key:

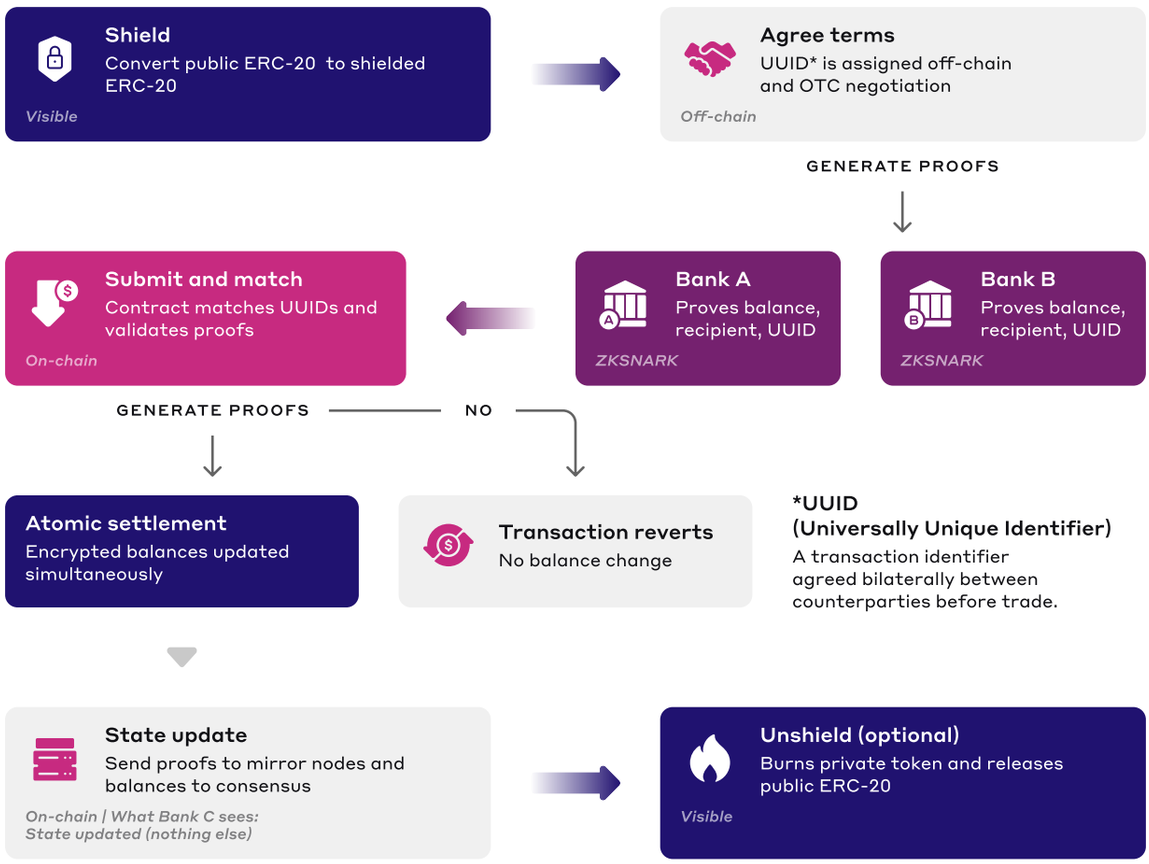
- A signing key, managed through the institution's existing custodian infrastructure or Hardware Security Module (HSM). No new key infrastructure is required.



# Transaction flow

A private atomic swap on HashSphere moves through six distinct stages, from shielding assets at the entry point to optional return to the public layer. Throughout, zero-knowledge proofs guarantee transaction validity without exposing any underlying values, counterparties, amounts, and asset types remain invisible to all uninvolved participants, including others on the same network.

TRANSACTION FLOW




 |  

# Real institutional use cases

## DvP: Tokenized bond settlement

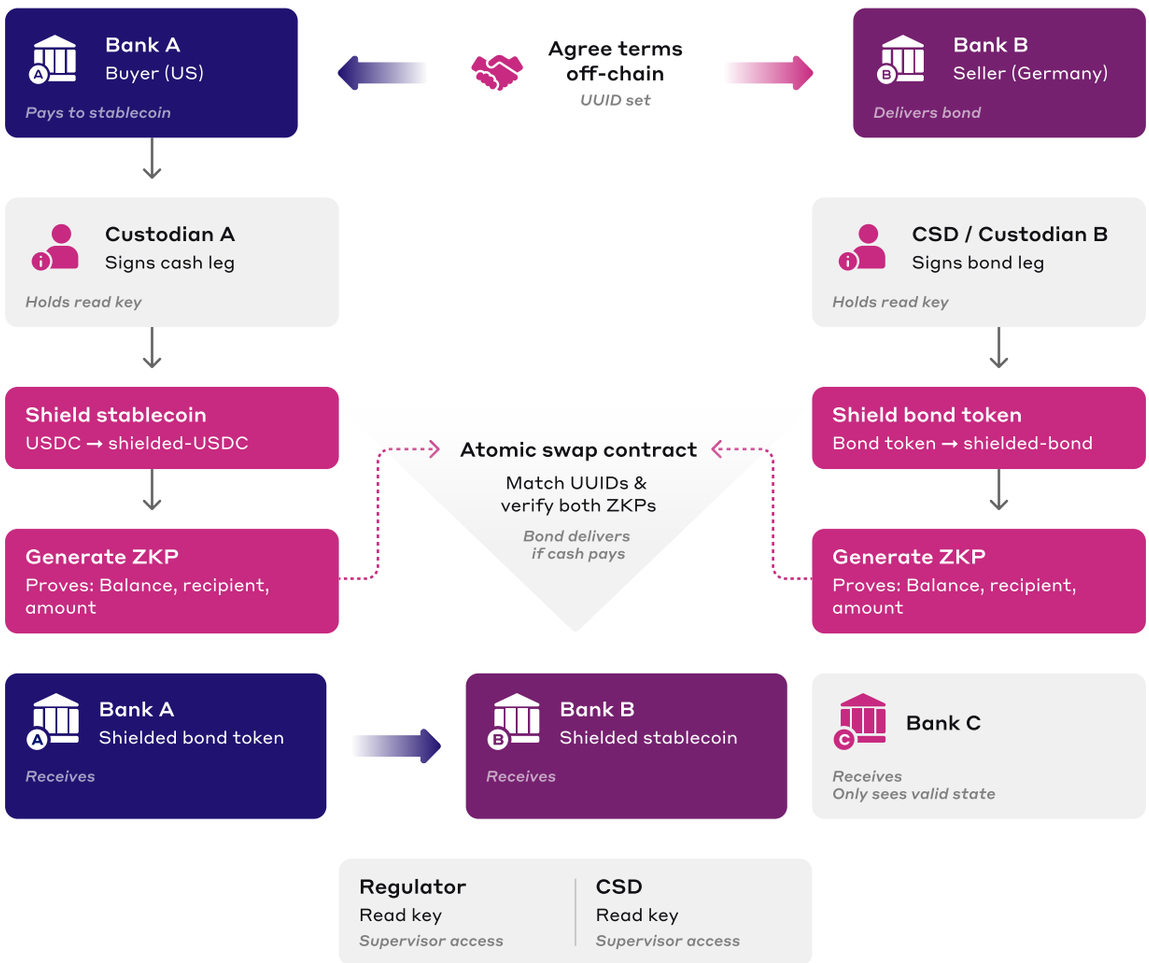
Bank A (US) purchases a tokenized bond from Bank B (Germany). The cash leg is settled in stablecoins on HashSphere. Bank C, also on the network, is uninvolved.

The transaction is atomic: the bond delivers only if cash pays, and the cash pays only if the bond delivers. Bank C sees that a valid private transaction occurred. It cannot see the parties, the price, or the size — protecting both sides from front-running, competitive intelligence leakage, or signaling.

Both institutions use their existing custody infrastructure for signing. The asset custodian, central securities depository, and regulator can each independently verify specific transaction details when required; through cryptographic verification, not policy-dependent access controls. No other participant sees anything.

### DVP TRANSACTION FLOW

Tokenized bond settlement



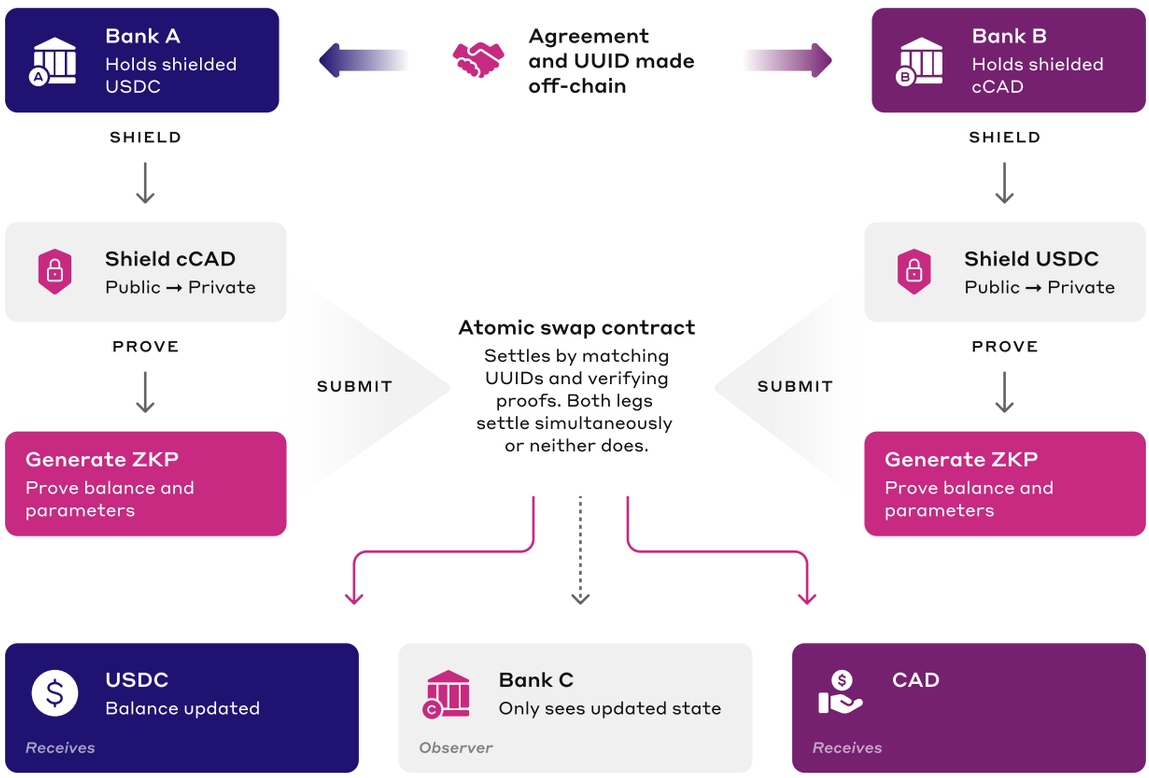
## PvP: Cross-border FX

Bank A and Bank B agree to exchange USDC for cCAD. The exchange rate is agreed bilaterally off-chain via an OTC mechanism. Both banks have already shielded their respective currencies.

Execution on-chain is atomic: both currency legs settle simultaneously or neither does. There is no correspondent bank in the middle. No bridge. No shared validator with a potential conflict of interest in the ordering of settlement. The transaction is private, fairly ordered, and final.

### PVP TRANSACTION FLOW

USDC / cCAD atomic swap



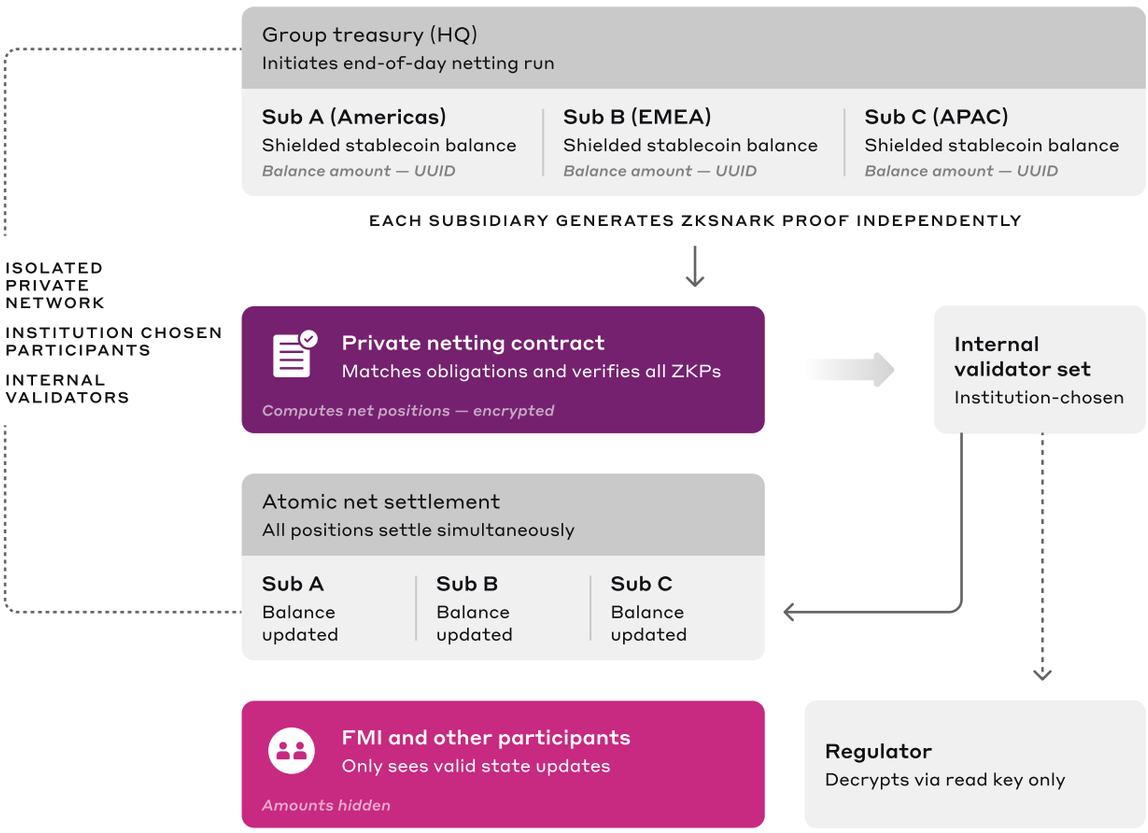
## Treasury management and intercompany settlement

A multinational corporation settles intercompany balances across subsidiaries. The financial market infrastructure provider (FMI) and other network participants must not see the amounts. The network is isolated to the institution's chosen participants. Validators are internal. Settlement is deterministic and auditable. The regulator has cryptographically enforced read access through a pre-issued read key.

At scale, this scenario requires approximately 18–20 TPS for an end-of-day netting of 100,000 transfers over a 90-minute window. HashSphere supports up to 10,000 TPS, and for confidential transactions within the network, a latency overhead of no more than two seconds is added above that baseline consensus time, inclusive of ZK proof generation.

### TREASURY MANAGEMENT AND INTERCOMPANY SETTLEMENT

Isolated private network of institution: Chosen participants and internal validators.



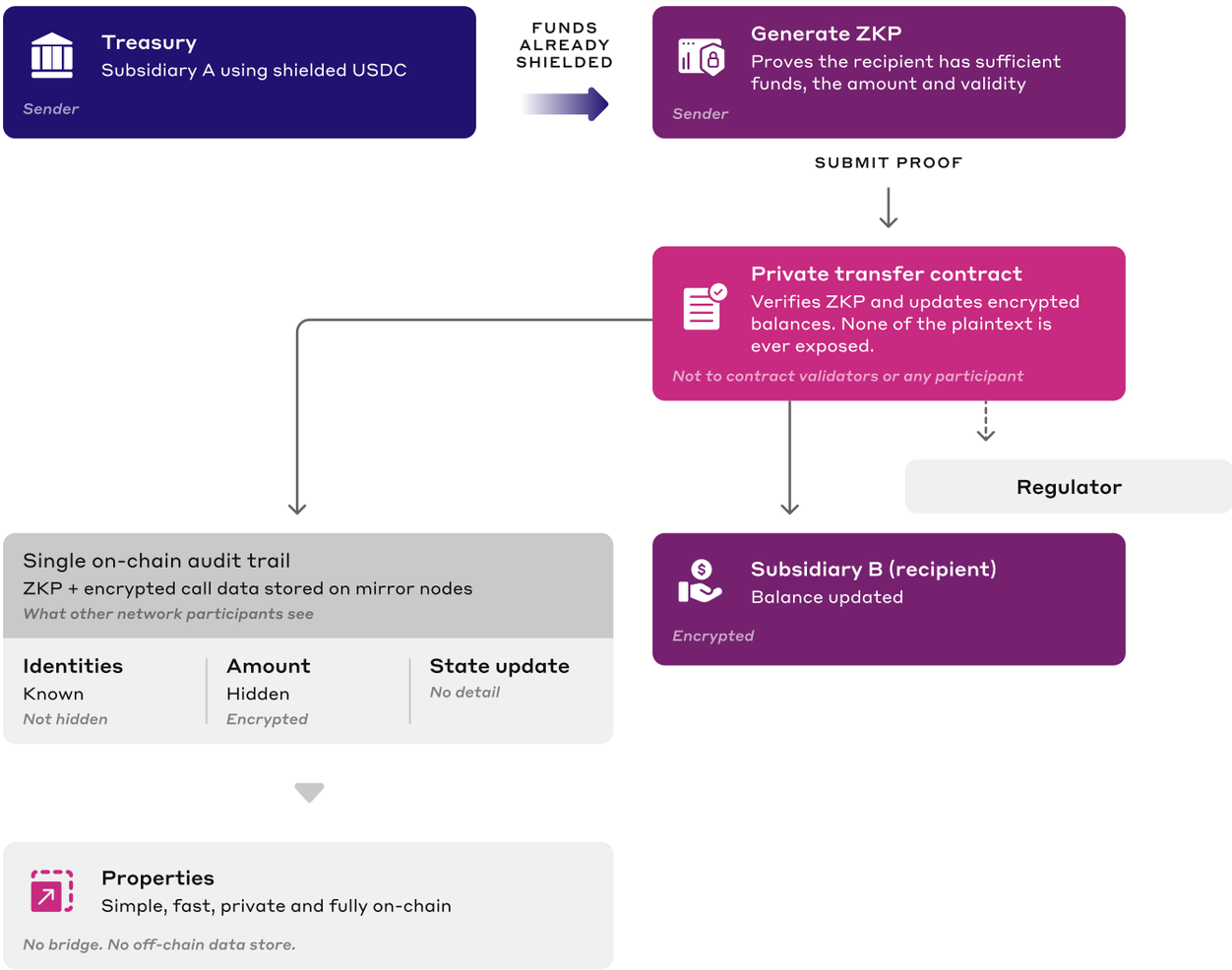
# Private transfer

A corporate treasury moves stablecoins between subsidiary accounts. The participants are known to each other. The amounts are not visible to anyone else on the network.

The transaction produces a single, complete audit trail. Simple, fast, private, and fully on-chain.

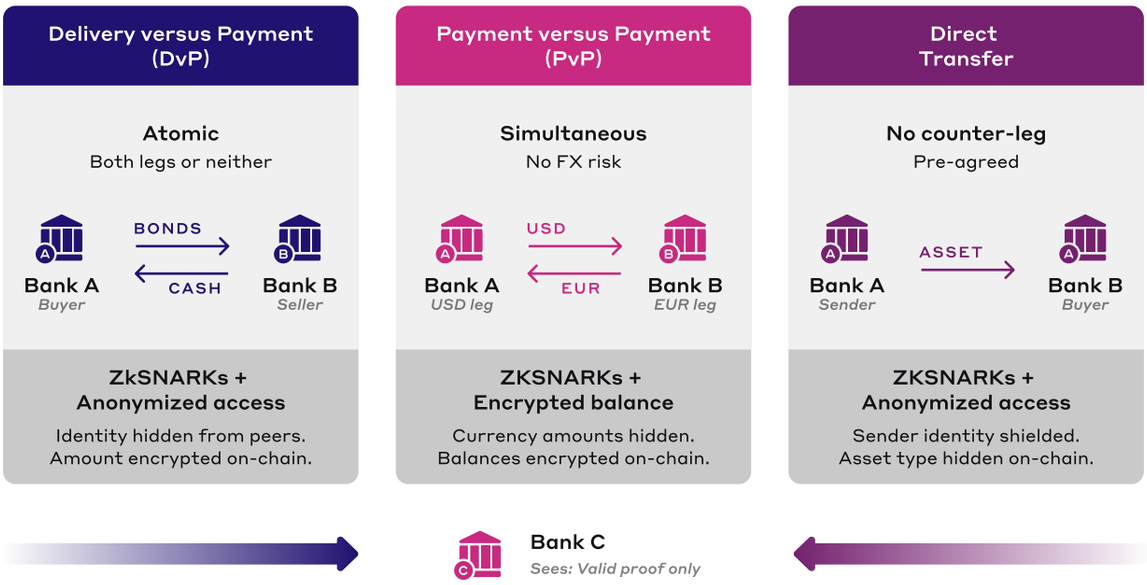
### PRIVATE TRANSFER TRANSACTION FLOW

Within a private network where participants are known, but amounts are hidden from all others.



# Private transfer continued

## TRANSACTION PRIVACY APPROACH ACROSS ALL SETTLEMENT TYPES




In-network privacy layer for all participants

**ZK-Proofs**
Prove transactions validity without revealing data.

**Anonymized access**
Obscure sender among network participants.

**ZkSNARKs**
Succinct proofs, zero knowledge.

**Private balance**
Compute on encrypted balances directly.

# What is and is not visible

**A note on metadata:** The submitter of a transaction is observable at the consensus protocol level, node operators may be able to see in log files that a transaction originated from a particular institution at a particular time. They cannot see the recipient, the amount, or the resulting balances. HashSphere has the option to add an anonymizing relay, submitting all transactions on behalf of participants to obscure origination, if customer requirements demand it.

### WHAT EACH PARTY CAN SEE

| PARTY | IDENTITY | AMOUNTS | BALANCES | ASSET TYPE |
|---|---|---|---|---|
| **Counterparties** | ✓ | ✓ | — | ✓ |
| **Asset custodians**<br>Via read key | ✓ | ✓ | ✓ | ✓ |
| **Regulator / auditor**<br>Via read key | ✓ | ✓ | ✓ | ✓ |
| **Other participants**<br>Bank C, uninvolved | — | — | — | — |
| **Consensus validators**<br>Encrypted state only | — | — | — | — |
| **The smart contract**<br>ZkSNARK proofs only | — | — | — | — |

✓ *Visible* — *Not visible. Cryptographically enforced.*

# Conclusion

The regulated institutions that will define the next era of financial infrastructure are not looking for a privacy feature. They are looking for a system in which value can move, across counterparties, asset classes, and jurisdictions, without exposing the positions, strategies, or relationships that make that value meaningful in the first place.

That is a harder problem than it sounds. Most approaches in the market today solve part of it. Some hide transaction content from participants which puts a centralized operator in the middle that can see everything. Some get the cryptography right but cannot yet run at production throughput. Every partial solution introduces a dependency on a trusted party, a performance ceiling, or a constraint that makes it unsuitable for the institutions where the stakes are highest.

HashSphere's approach is different in one fundamental respect: privacy is not achieved by withholding information from some parties and routing it through others. It is achieved by making the underlying data mathematically inaccessible to anyone who is not a designated party to the transaction, including the network itself.

The zkSNARK proves validity and updates balances without ever decrypting them. The atomic swap contract settles both legs simultaneously or neither. Regulators receive cryptographic proof, not a policy promise, scoped to exactly what needs to be disclosed, and nothing more.

The four use cases in this paper, DvP bond settlement, cross-border PvP FX, intercompany treasury netting, and private transfer,  are not hypotheticals. They represent the actual settlement workflows of Tier-1 banks, multinationals, and asset managers that cannot operate on infrastructure where their counterparties, competitors, and infrastructure providers can see their flows. HashSphere makes those workflows possible today, on a production network, without requiring institutions to rebuild their custody infrastructure, adopt a proprietary smart contract language, or trust a third party not to look.

Privacy in this context is not a feature to be toggled on. It is the foundation on which institutional adoption of tokenized assets depends. HashSphere is powered by a DLT that is built on that foundation.

**NEXT IN THE BEHIND THE LEDGER SERIES**

Network Isolation and the Sovereign Perimeter, how HashSphere's permissioned design differs from subnets on public infrastructure, and why that distinction matters for institutional deployment.

Hashgraph

# About Hashgraph

Hashgraph is an enterprise software firm powering the next generation of connected digital markets. The company develops solutions for financial services that leverage hashgraph technology and provides ongoing support to the Hedera public network. Through CLPR (cross-ledger interoperability), private network infrastructure, and tokenization platforms, Hashgraph enables institutions to issue, trade, and manage assets seamlessly across systems, helping to drive a more connected digital economy.

For more information, visit hashgraph.com