Hyperledger Fabric V2.0 Private Data and Decentralized Application Patterns

Dave Enyeart
Wenjian Qiao
Manish Sethi
Jason Yellick

Agenda

- **Hyperledger Fabric v1.x Refresher**
  - **Private Data Collections (私数据集合)**
  - **Endorsement (背书)**
  - **Commit (提交)**
  - **Reconciliation（和解）**
- **Hyperledger Fabric v2.0**
  - **Private data enhancements（私有数据增强）**
  - **Decentralized application patterns （去中心化的应用模式）**

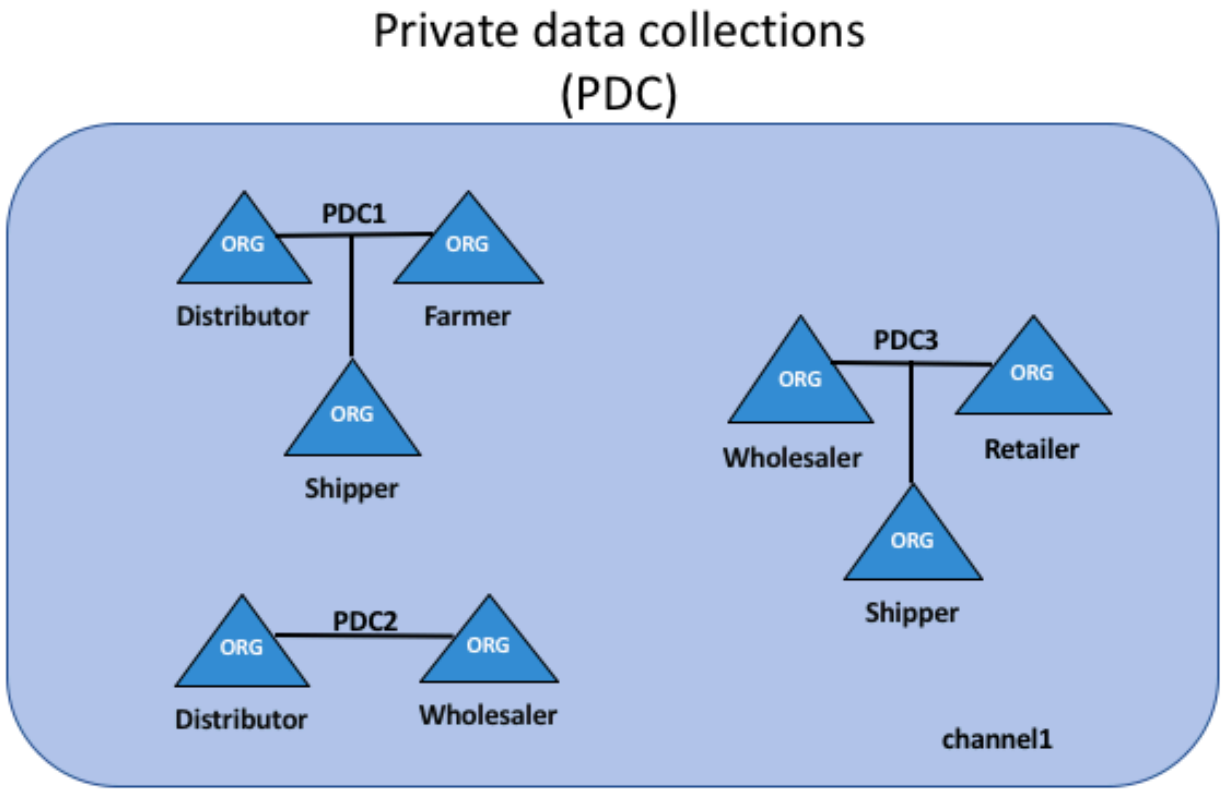
Private Data Collections (私有数据集合)

- On a channel, a group of organizations share private data but keep the data from other organizations (同一通道上多个组织之间共享数据, 但是对通道上其他组织保持数据私密)
- Defined during chaincode instantiation/upgrade in v1.x （在链码实例化/升级时定义）
- Policy defines which organizations' peers are allowed to persist the collection data （定义哪些组织的 Peer 节点被授权储存集合数据）

```
{
  "name": "collectionAB",
  "policy": "OR('Org1MSP.member', 'Org2MSP.member')",
  "requiredPeerCount": 1,
  "maxPeerCount": 2,
  "blockToLive":1000000,
  "memberOnlyRead": false,
  "memberOnlyWrite": false
}
```

Private Data Collections - Example

- 私有数据集合 PDC1： **分销商, 农民, 托运商**
- 私有数据集合 PDC2：**分销商, 批发商**
- 私有数据集合 PDC3：**批发商, 零售商, 托运商**

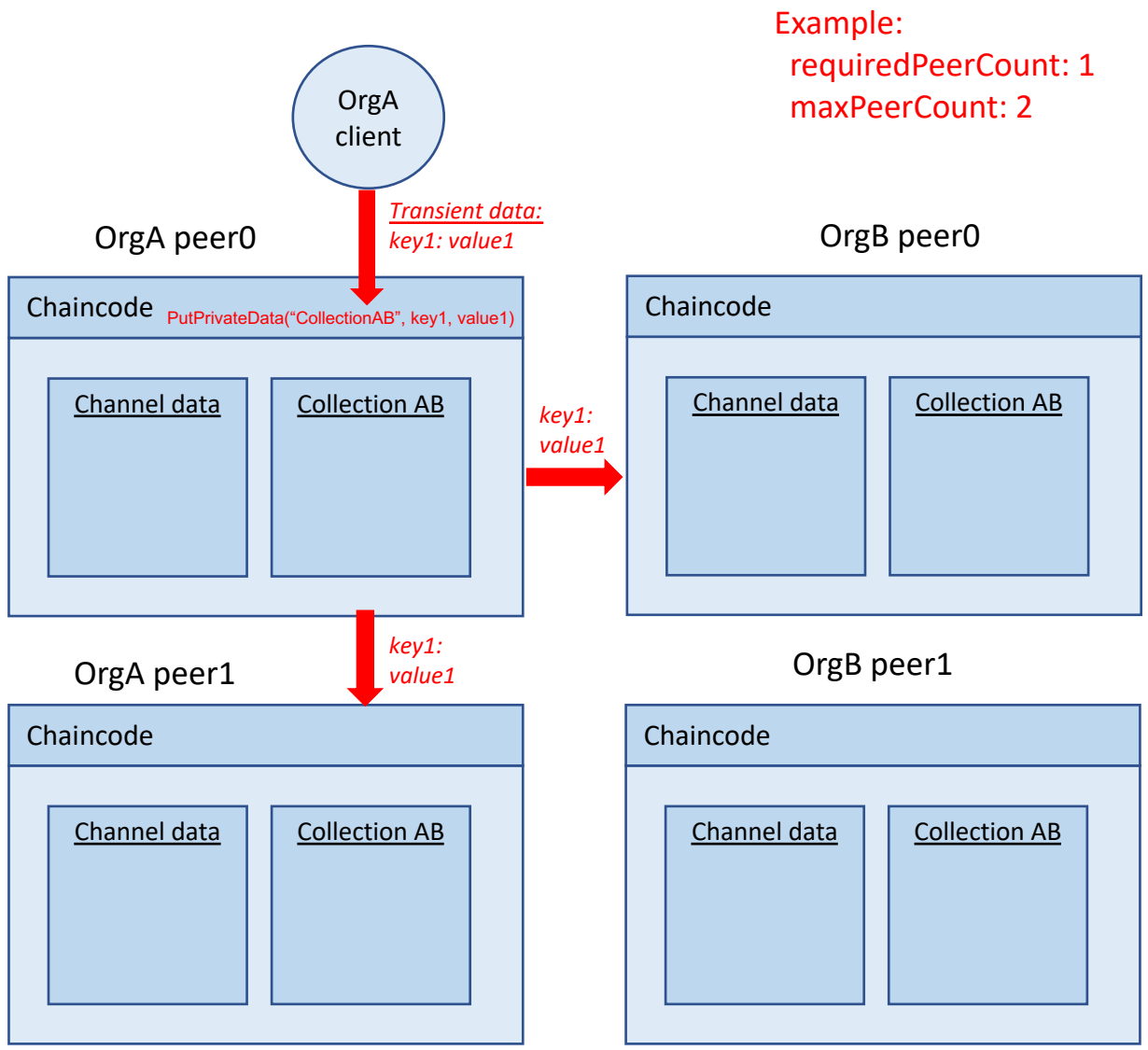


Private Data – Endorsement （背书）

- Private data is disseminated peer-to-peer by gossip to authorized org peers（背书节点分发私有数据到被授权的组织节点）
- Private data is initially persisted in peer's **transient store**, will move to collection state database upon commit （私有数据先存在临时存储，交易提交时再存到StateDB）
- **Transient data excluded from block transaction**. Only hash of private data included in block transaction （临时数据不会被包含在提交到排序服务的交易中，也就不会被包含在区块中）

Important collection dissemination properties:

- **requiredPeerCount**
- **maxPeerCount**

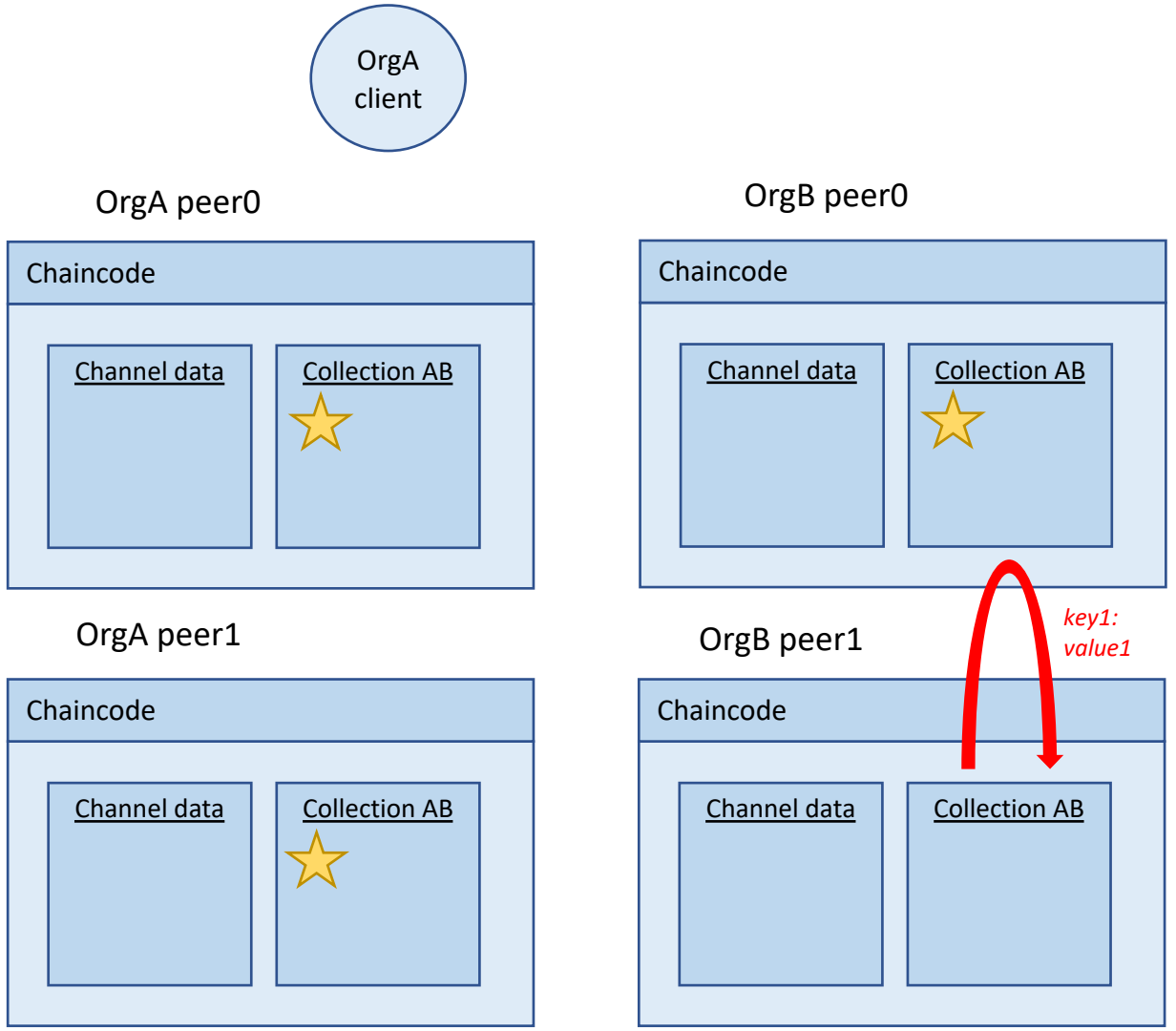


Private Data – Commit （提交）

- If transaction is validated, private data moved from transient store to collection state database （交易验证后私有数据和哈希值将被储存到state database）
- Unauthorized peers receive a hash of the private data in the block transaction （未被授权的节点只存储哈希值）
- Any authorized peer that doesn't yet have the private data, will attempt to pull the private data from other authorized peers （如果没有私有数据，被授权节点会试图从其它被授权节点获取）

Important core.yaml properties:

- peer.gossip.pvtData.**pullRetryThreshold** – Amount of time peer will attempt to pull private data before giving up.

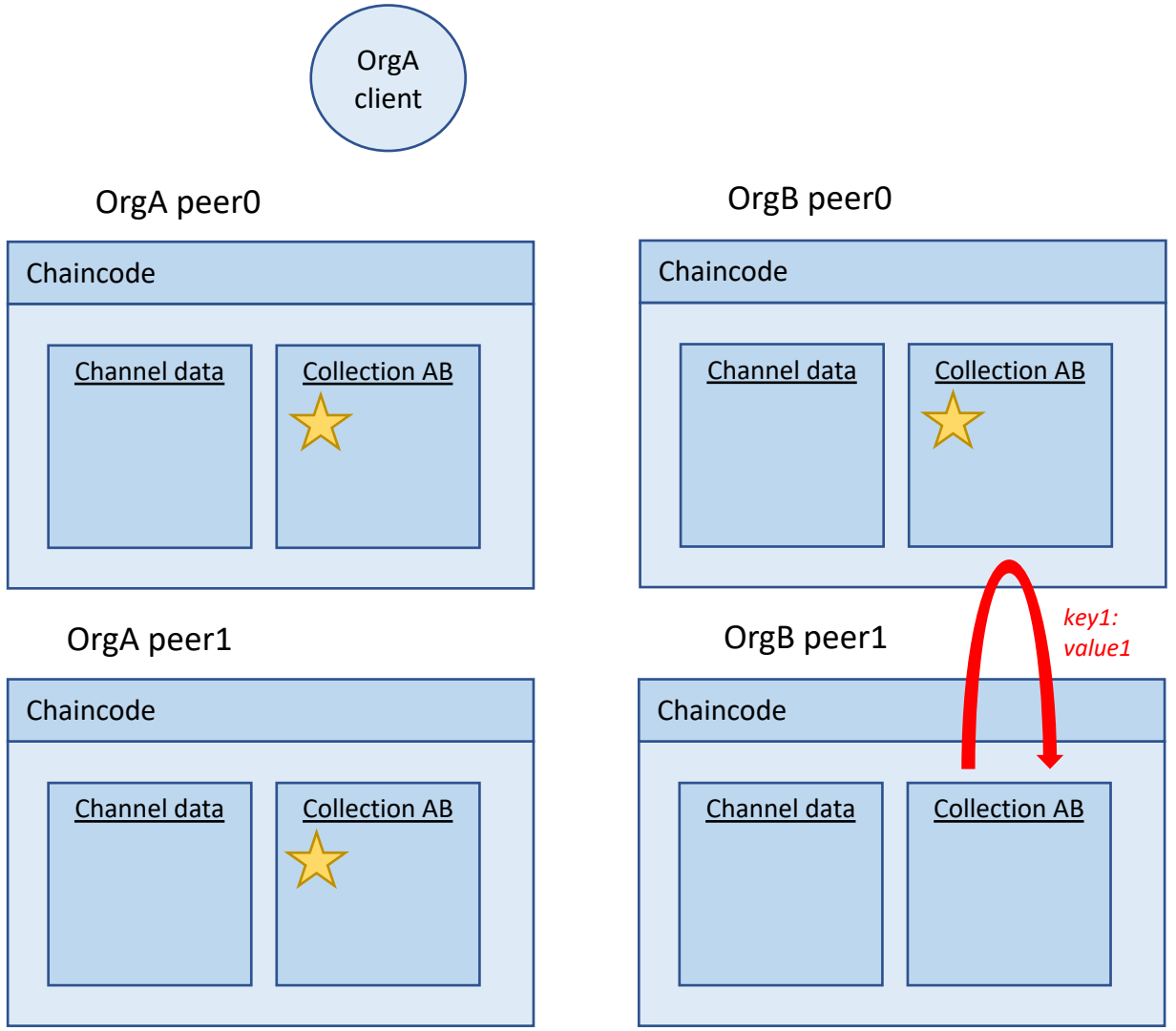


Private Data – Reconciliation（和解）

- If an authorized peer was not able to find the private data at commit time, it will attempt to reconcile (pull) later (如果被授权的节点没有得到私有数据，它将在提交后继续试图获得私有数据)

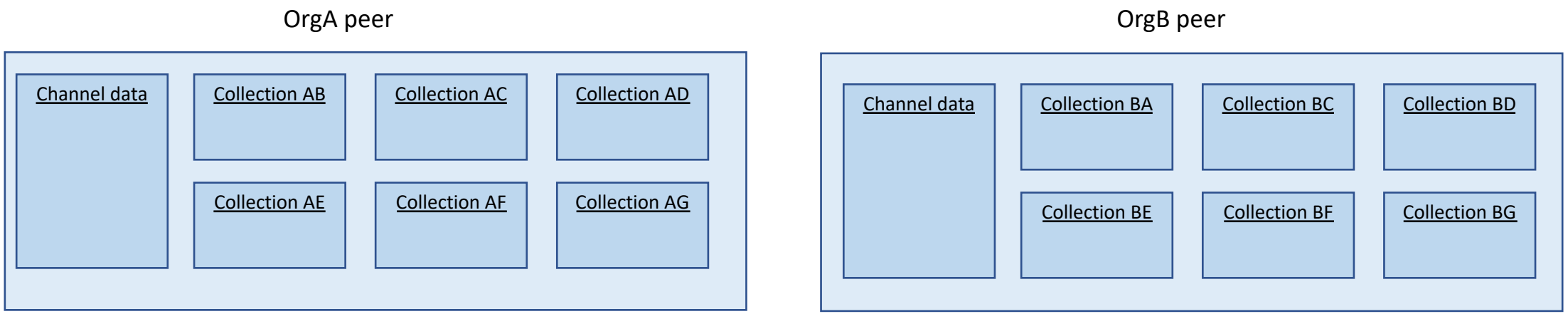
Important core.yaml properties:

- **peer.gossip.pvtData.reconciliationEnabled**
- **peer.gossip.pvtData.reconcileSleepInterval**



Private Data in v1.x

- Must define private data collections for all combinations of orgs that may want to transact (必须为所有可能交易私有数据的组织定义私有数据集合)
  - Even if it is for a single organization（即使是单独的组织）
  - Bilateral collections could be a pain point（定义双边集合是个痛点)

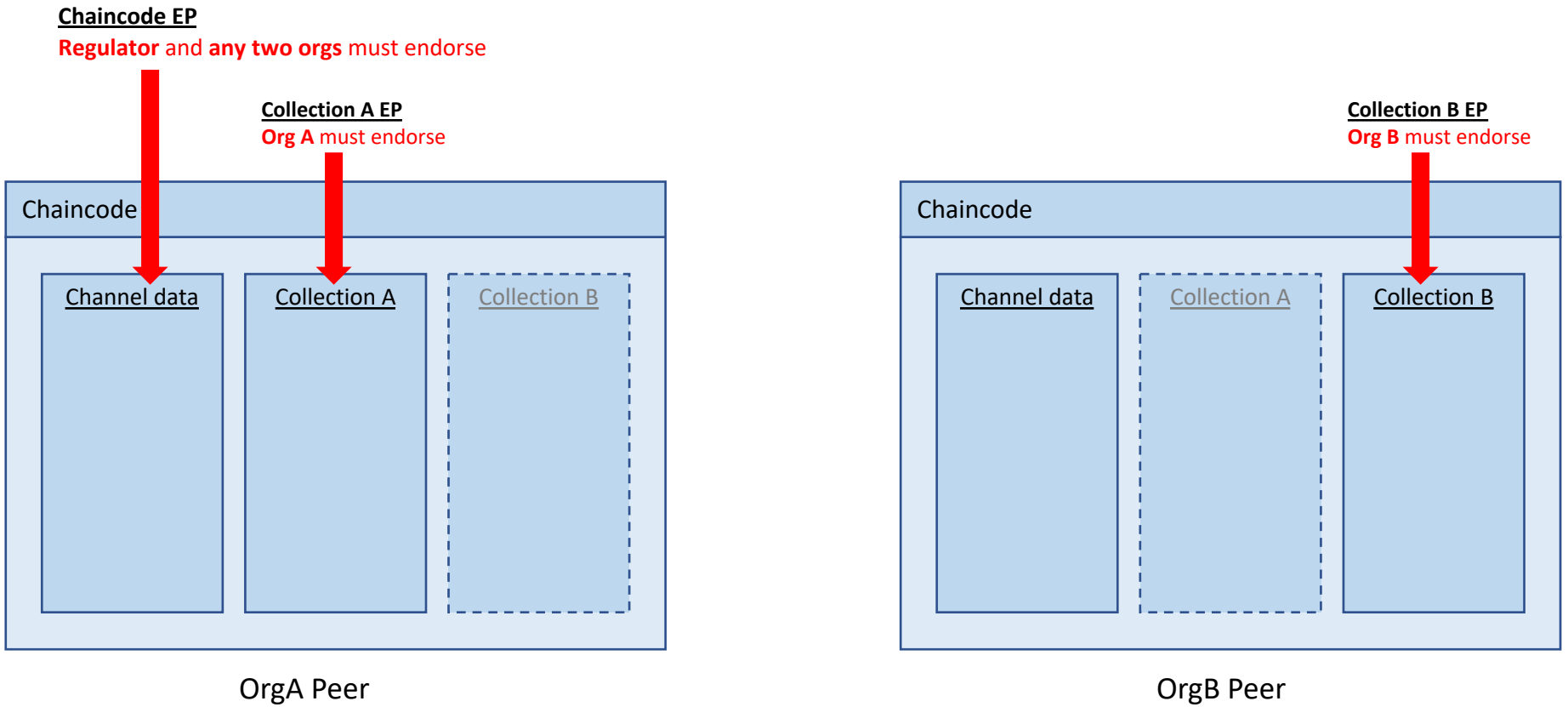


V2.0 Enhancements（V2.0增强）

- Endorsement policies on private data collections （集合级别的背书策略）
- Implicit organization-specific private data collections （每个组织隐式的私有数据集合）
- Share and verify private data (共享和验证私有数据)
  - Share private data across collections (跨集合共享私有数据)
    - each collection may include a single organization, or perhaps a single organization along with a regulator or auditor(每个集合可能包括单个组织，或者带有一个监管者或审计师的组织)
  - Decentralized application patterns （去中心化的应用模式）

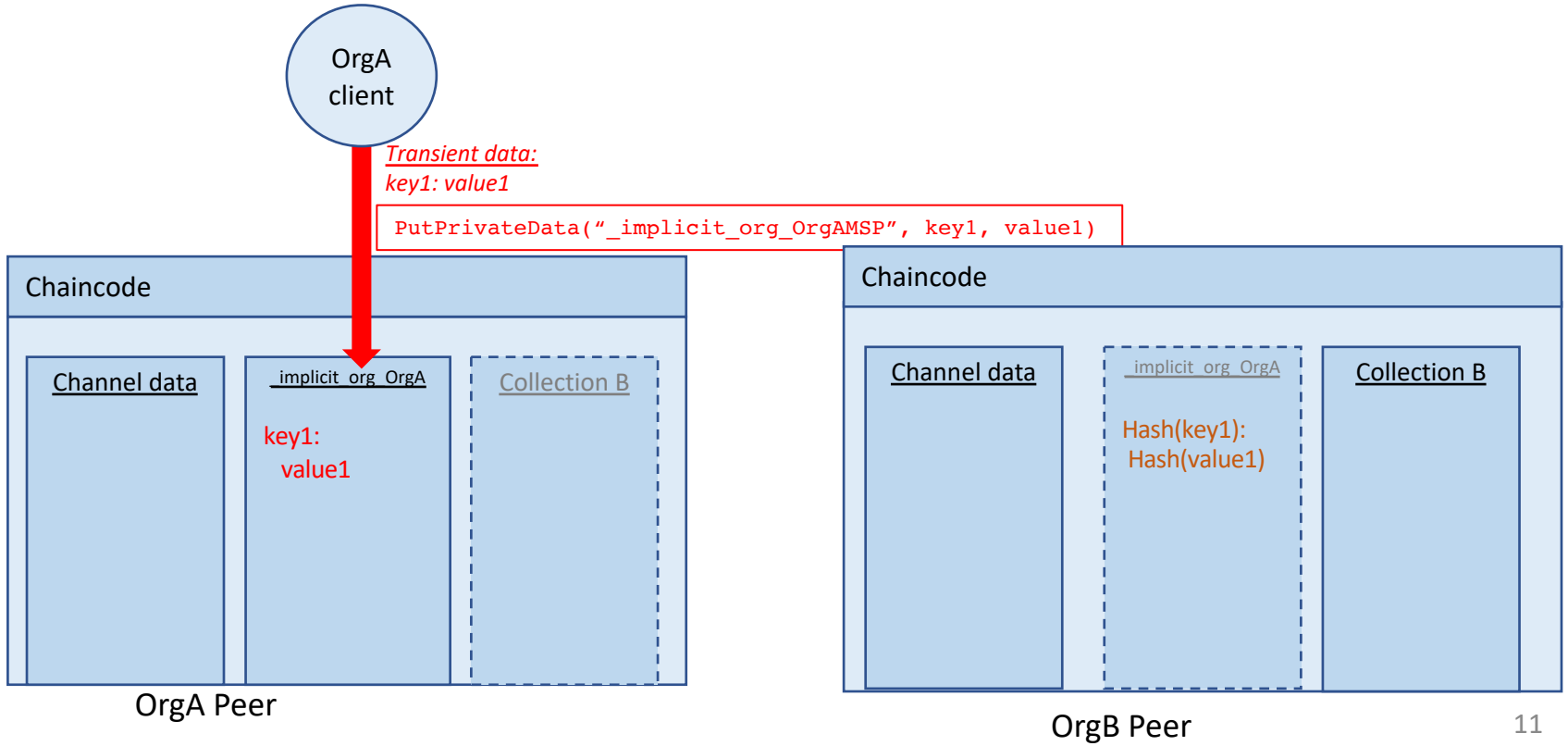
Endorsement policies on private data collections

- Overrides chaincode-level endorsement policy for any data written to the collection (覆盖链码级别的背书策略)



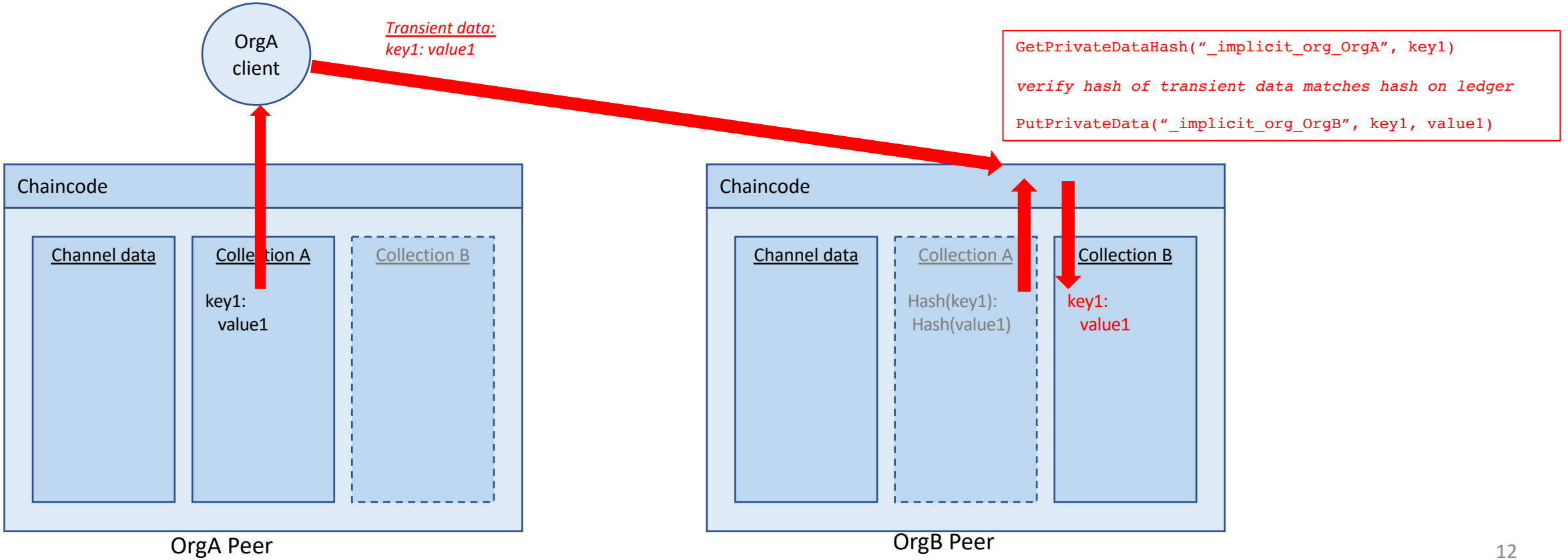
Implicit Organization Specific Private Data Collections

- Available for each org in the channel, _implicit_org_<MSPID>, no need to define a collection （不需要定义集合）
- Only peers from the organization can read/write the private data（只有组织内的peers可以读写私有数据）
- Implicit endorsement policy matching the organization （隐式背书策略匹配相应的组织）
  - Since writes are guaranteed to be endorsed by organization's own trusted peer, it can be used to indicate organization's agreement in decentralized applications



Share and Verify Private Data

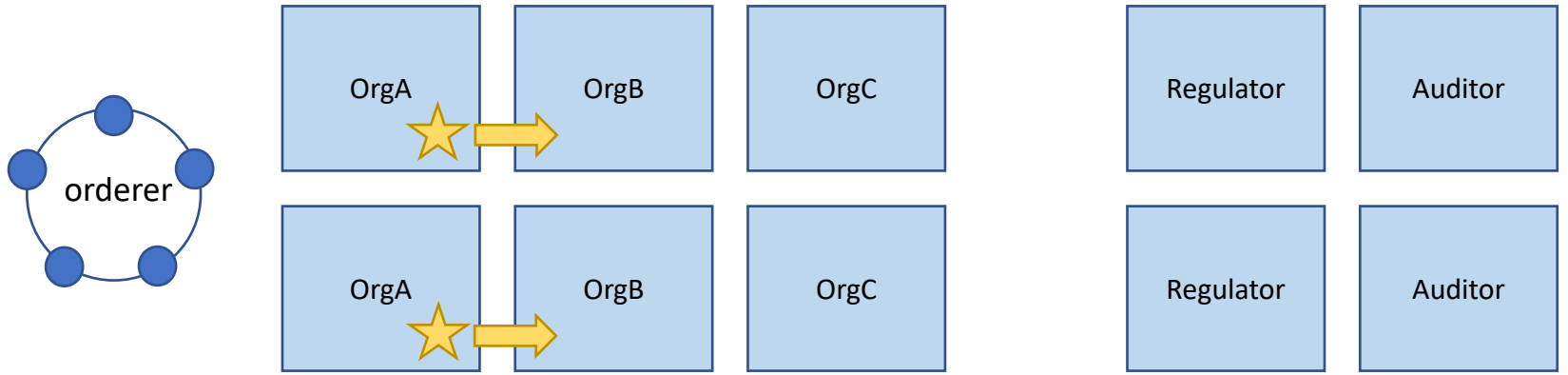
- Share private data on a need-to-know basis (在需要知道的基础上共享私有数据)
- `GetPrivateDataHash()` chaincode API enables non-member to verify the shared private data against hashes stored on the ledger （非集合成员通过储存在账本中的哈希值验证私有数据）



Example: Decentralized application for asset transfer

Scenario: Asset Transfer

- Transfer an asset from OrgA to OrgB (从OrgA到OrgB转移资产）
- Both parties must come to agreement on a price, with proof of their agreement recorded on the ledger prior to the asset transfer (双方必须同意价格，并提供储存在账本上的证明）
- Keep asset and transaction details private between the transactors（保持资产和交易信息是交易双方的隐私）
- Share private data on a need-to-know basis with regulator, auditor, or potential future transactors, such that they can verify the private data and transaction records（在需要知道的基础上，可以和监管者, 审核员，其他交易者分享私有数据）

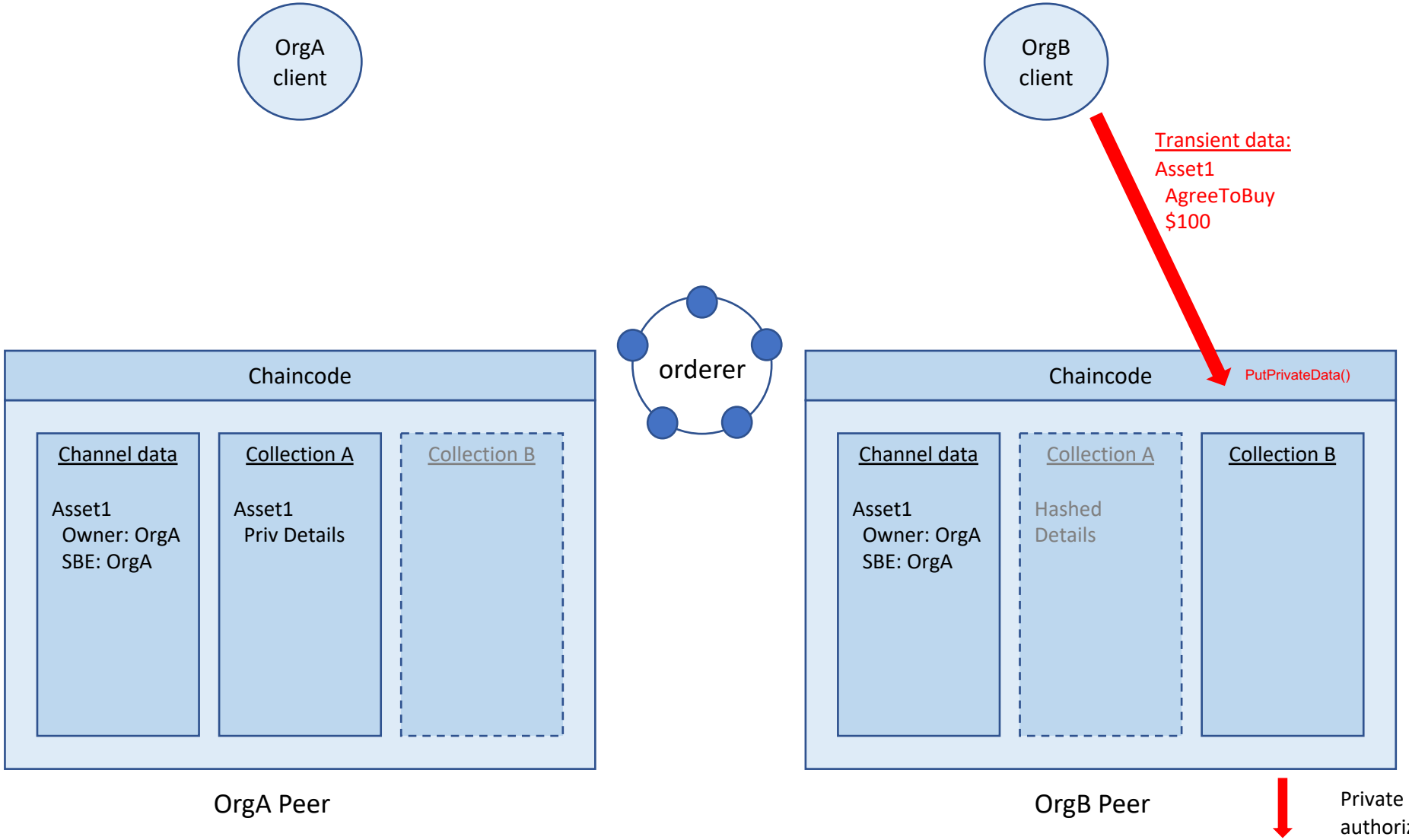


Consider a network (single channel) with multiple organizations, a regulator, an auditor, and an ordering service. Each member has two peers for HA and data redundancy.

Asset Transfer: 3 Transactions

- Pre-condition（前提）：
  - OrgA owns an asset (private data stored in ledger) (OrgA拥有资产并且储存在账本里)
  - OrgA and OrgB have agreed on the price for asset transfer (OrgA和OrgB同意资产转移的价格)
- Transaction 1:
  - OrgB indicates agreement to buy (OrgB 表明同意买资产)
- Transaction 2:
  - OrgA indicates agreement to sell (OrgA 表明同意卖资产)
- Transaction 3:
  - OrgA initiates asset transfer （OrgA发起资产转移）

Transaction1: OrgB indicates agreement to buy – endorsement



Transaction1 - endorsement

OrgA owns private data Asset1

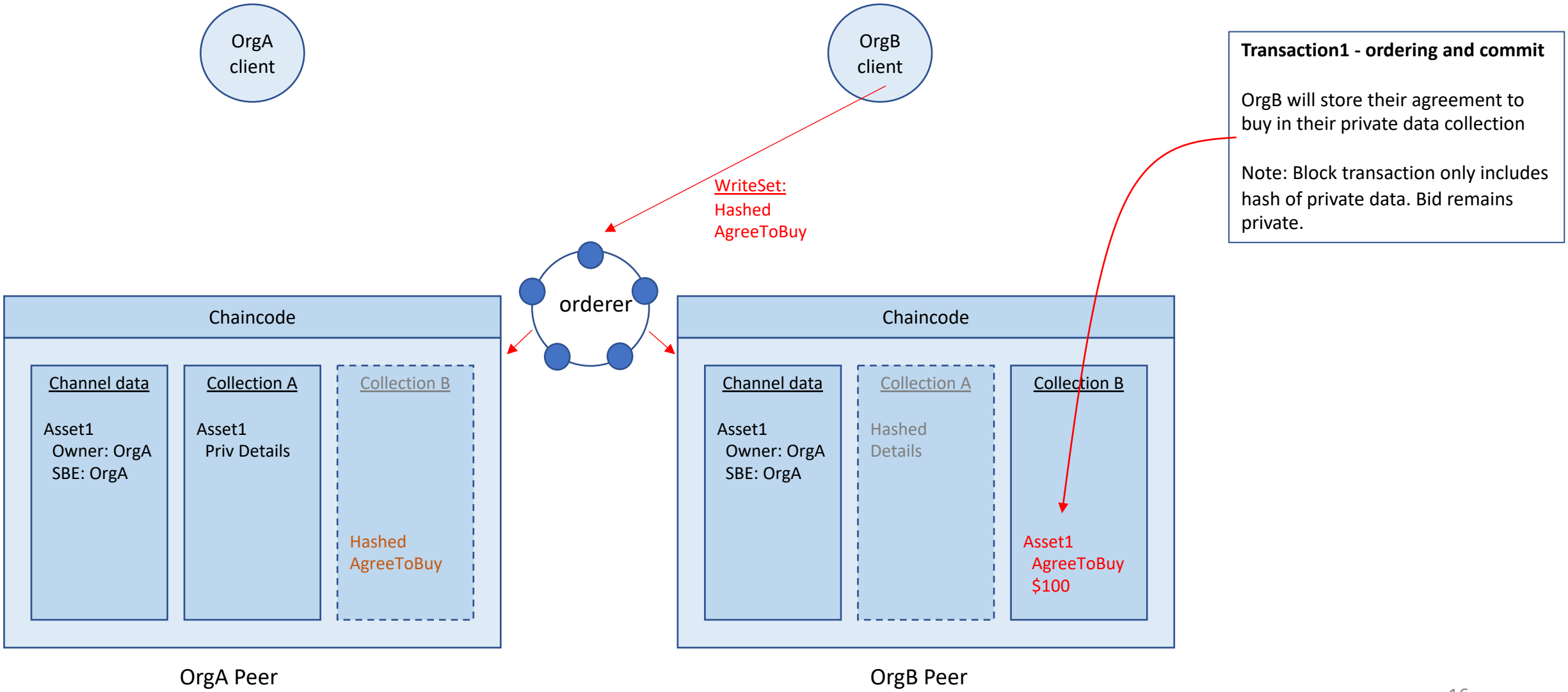
OrgB invokes chaincode to indicate agreement to buy for $100.

Chaincode verifies request is coming from an OrgB client.

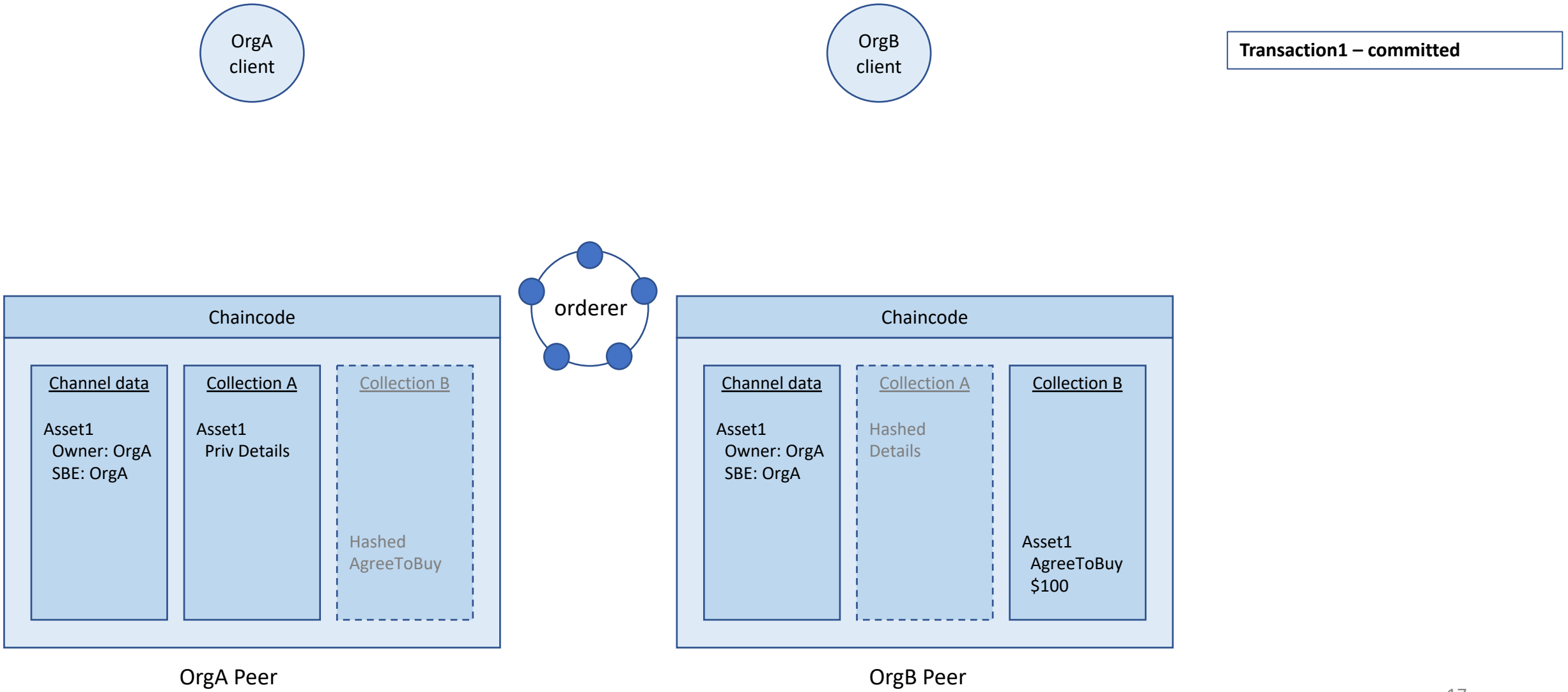
Only an OrgB peer needs to endorse, since only CollectionB is updated.

Note: private data is passed to chaincode in transient field and disseminated to other authorized OrgB peers for redundancy.

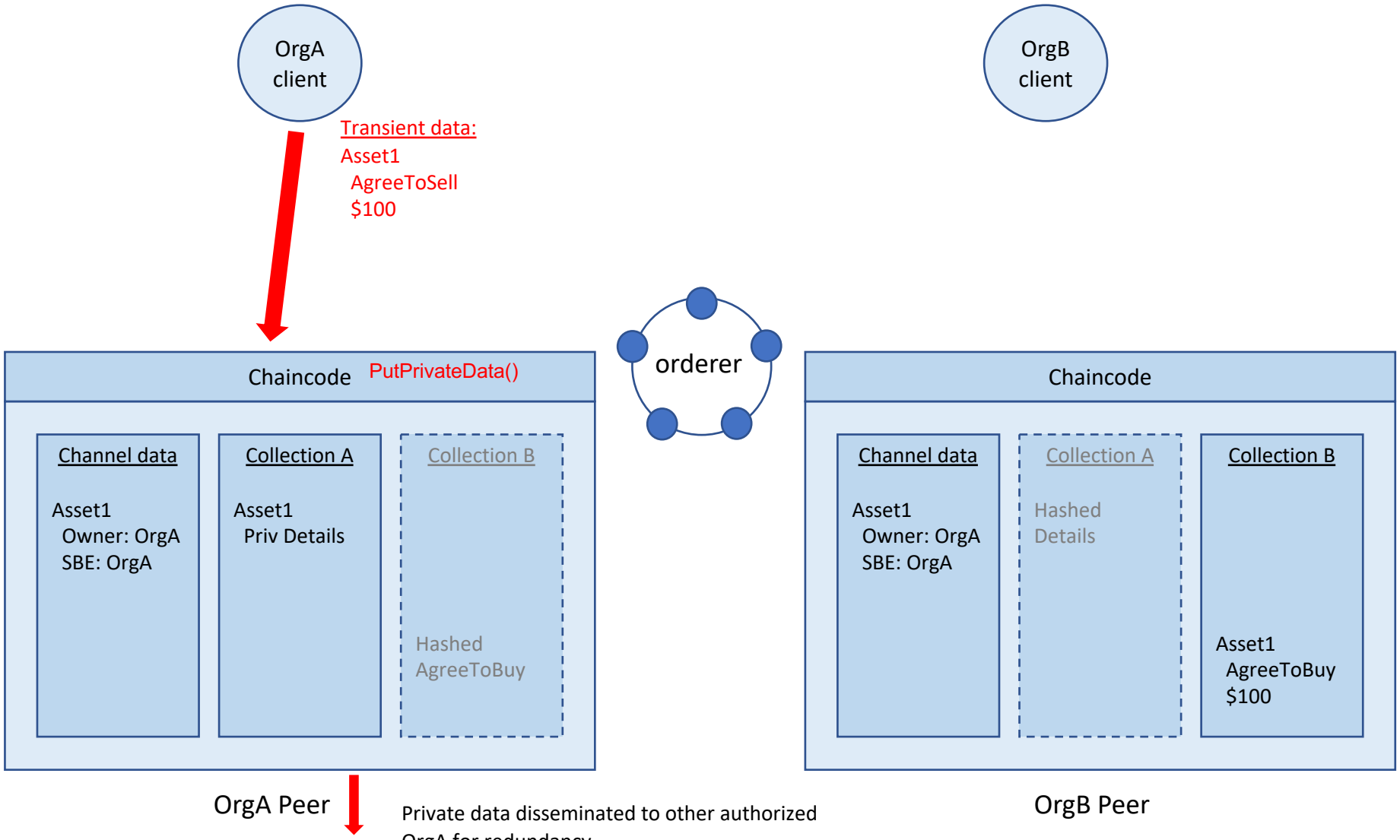
Transaction1: OrgB indicates agreement to buy – commit



Transaction1: OrgB indicates agreement to buy - committed



Transaction2: OrgA indicates agreement to sell – endorsement



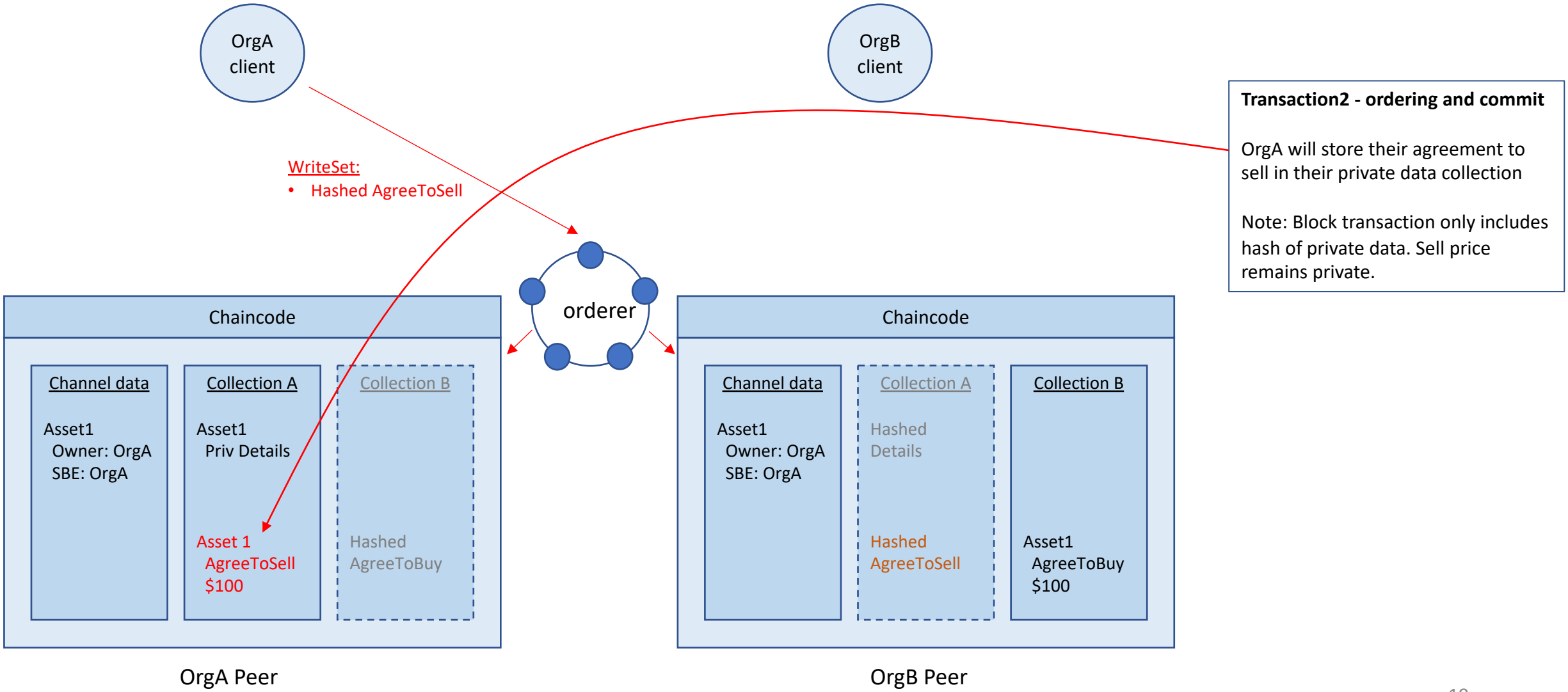
Private data disseminated to other authorized OrgA for redundancy.

Transaction2 - endorsement

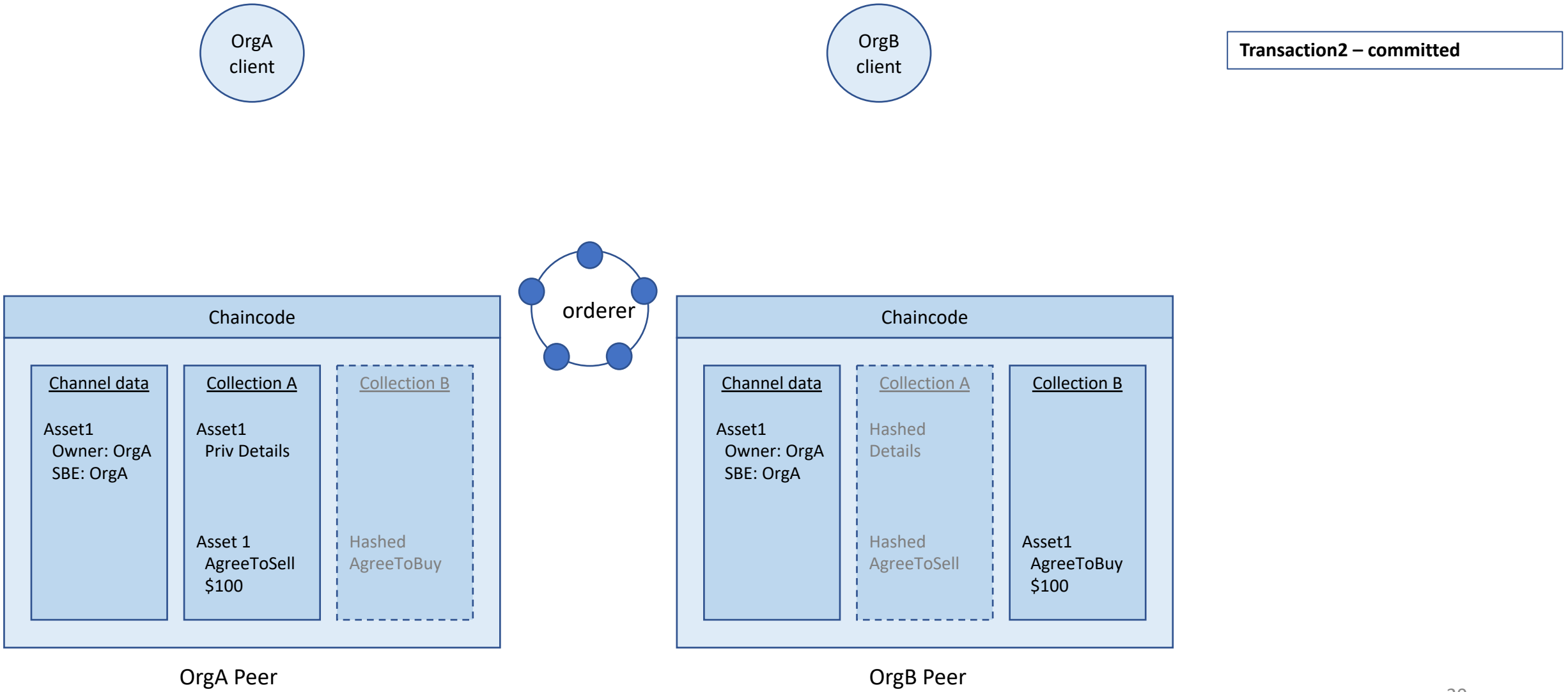
OrgA invokes chaincode to indicate agreement to sell for $100.

Chaincode verifies request is coming from an OrgA client.

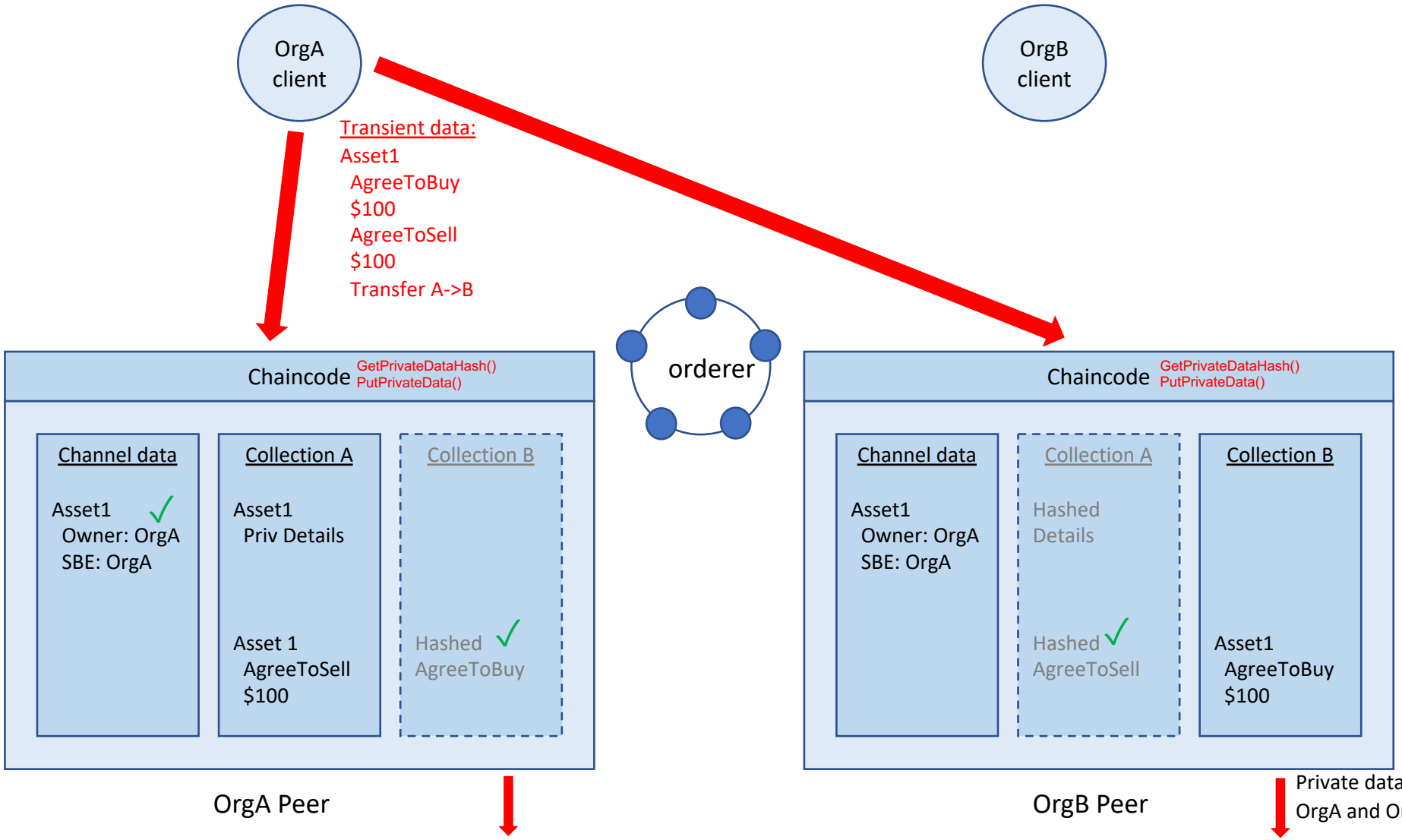
Transaction2: OrgA indicates agreement to sell – commit



Transaction2: OrgA indicates agreement to sell - committed



Transaction3: OrgA initiates asset transfer – endorsement



Transaction3 - endorsement

OrgA invokes chaincode to transfer asset, the agreed to terms are passed as input in transient field.

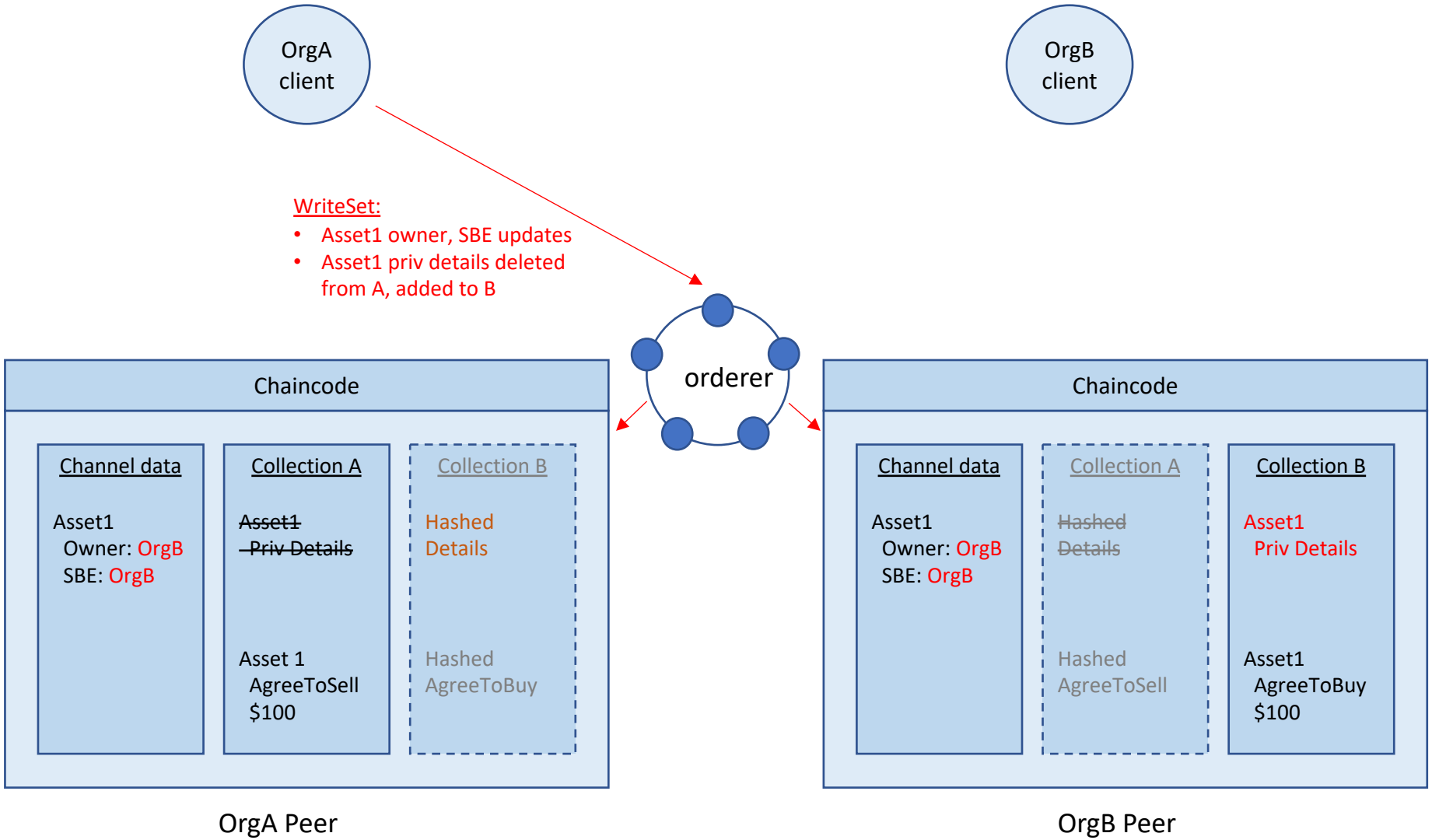
Chaincode verifies owning org is making the request.

Chaincode verifies that both parties have agreed on the passed price – calls GetPrivateDataHash() on each each collection.

Chaincode data updates:
-Asset1 Priv Details deleted from ColA
-Asset1 Priv Details added to ColB
-Asset1 owner and SBE updated to OrgB

OrgA must endorse due to SBE and Collection A updates.
OrgB must endorse due to the CollectionB updates.

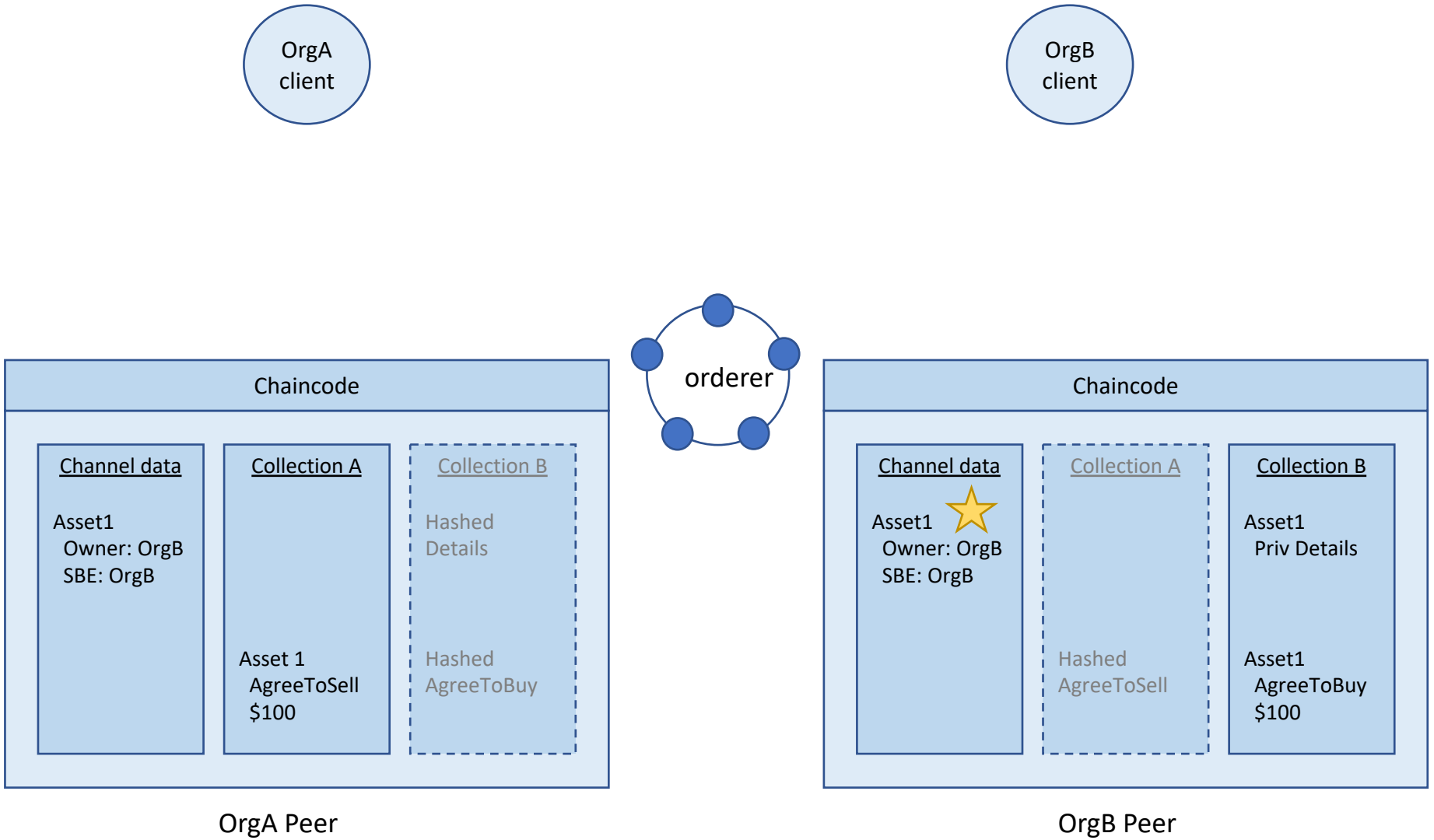
Transaction3: OrgA initiates asset transfer – commit



Transaction3 - ordering and commit

Note: Block transaction only includes hash of private data. Private details and price remains private.

Transaction3: OrgA initiates asset transfer - committed



Transaction3 – committed

Asset transferred.

Many variations of these patterns can be applied in chaincode

Examples…

- Regulator or other intermediary must endorse the asset transfer (transaction 3)
- Regulator or auditor optionally included in private data dissemination policy
  - e.g. private data collection for each buyer-regulator and seller-regulator combination
- Make agreement to buy known to seller by also writing in seller's private data collection
  - e.g. to support auction scenarios with multiple bidders, where seller queries for highest bidder
- Key-level ACLs for sharing specific private data from your own collection with specific clients
- Verify a payment record on ledger for payment versus delivery requirements
- Verify if a bank has submitted a letter of credit record to the ledger
- Transact through custodial organizations instead of every transactor hosting a peer
- Etc, etc