RGB Tech Internals, part I

General overview & tech introduction to RGB

LNP/BP Standards Association

Prepared & supervised by **Dr Maxim Orlovsky**, **Pandora Core AG**

Created with support from **Bitfinex & Fulgur Ventures**

What RGB is?

Smart contracts layer on top of Bitcoin & Lightning Network, with emphasise on privacy & scalability

“Mass market” definition

What RGB is?

Digital right ownership management system,
confidential & censorship-resistant
built on and for Bitcoin and Lightning Network

Legal definition

- Right ownership is assigned to Bitcoin transaction outputs
- Meaning that it is managed by Bitcoin transaction validation rules (Bitcoin script)
- Details on rights (state) and additional limits on rights management are managed by client-validated data

1. There always must be an owner

- Smart contract state is not a “public good” (Ethereum/“blockchain” approach);
  it must always have a well-defined ownership (private, multisig…).

- RGB defines ownership by binding/assigning state to Bitcoin transaction outputs with single-use seals: whoever controls the output owns the associated state

- I.e. RGB leverages Bitcoin script security model and all its technologies (Schnorr/Taproot etc).

2. State ownership != state validation

- Ownership defines WHO can change the state
- Validation rules (client-side validation) define HOW it may change

2. State ownership != state validation

- Ownership controlled by Bitcoin script, at Bitcoin blockchain level (non-Turing complete)
- Validation rules controlled by RGB Schema with Simplicity script (Turing-complete)

This allows to avoid mistake done by “blockchain smart contracts” (Ethereum/EOS/Polkadot etc): mixing of layers & Turing completeness into non-scalable blockchain layer

Also it makes possible for smart contracts to operate on top of Layer 2 solutions (Lightning Network)

What RGB is?

Distributed system of partially-replicated state machines without globally-known state, having nearly-synchronous state consistency property enforced by consensus protocol of underlying layer (single-use-seal medium).

Computer Science definition

RGB advantages as Client-validated system

- **Scalability** in terms of **speed**: use of Lightning network
- **Scalability** in terms of **storage** requirements: data are kept only by “owners”, not all nodes (=no global state)
- **Privacy**: no global state, so data leaks are much less common
- **Confidentiality**: nothing restricts from using cutting-edge cryptography & zero-knowledge proofs
- **Abstraction**: ready for future bitcoin upgrades – transactions do not keep any data

RGB System Components

Bitcoin Transaction Graph

- Graph reconstructed from PoW chain containing most of work
(“mined part” of Bitcoin blockchain)

- Most recent transactions from Lightning channel

- …other future graph sources verified by a local party
    - multi-party channels;
    - UTXO-based sidechains, including Liquid, for parties accepting them
    - ...

Client-validated Data

- Data with a certain pre-defined structure and validation rules linked to Bitcoin Transactions

Bitcoin transaction graph
(in blocks or LN channel)

Client-validated data
(in blocks or LN channel)

State assignment

State data

Unique TXO definition

Bitcoin transaction graph
(in blocks or LN channel)

Client-validated data
(in blocks or LN channel)

State assignment

Commitment

Some new data

State data

Unique TXO definition

RGB System Components

Bitcoin Transaction Graph

- Graph reconstructed from PoW chain containing most of work (“mined part” of Bitcoin blockchain)
- Most recent transactions from Lightning channel
- …other future graph sources verified by a local party
  - multi-party channels;
  - UTXO-based sidechains, including Liquid, for parties accepting them
  - ...

Single-use-seals

- Links client-validated data to Bitcoin transactions
- More than timestamps: ensures unique history of events
- Many-to-many linking

Client-validated Data

- Data with a certain pre-defined structure and validation rules linked to Bitcoin transactions

Single-use-seal is an agreement on future commitment
(where it will happen and which form it will take)

Single-use seal WTF

- A promise by Alice (public or private) to Bob
- to create a commitment to some message
- at the well-defined point (in time, space, or any other form of phase space)
- This point is named “seal” and the process of its definition via Alice’s promise is a “seal definition”
- When Alice creates a commitment to that message at that defined point, she *closes the seal over a message*, producing the *witness* (of the commitment)

What we need

- Assign state to UTXO: **single-use-seal definition**
  Bitcoin script of the UTXO will control the ownership of rights

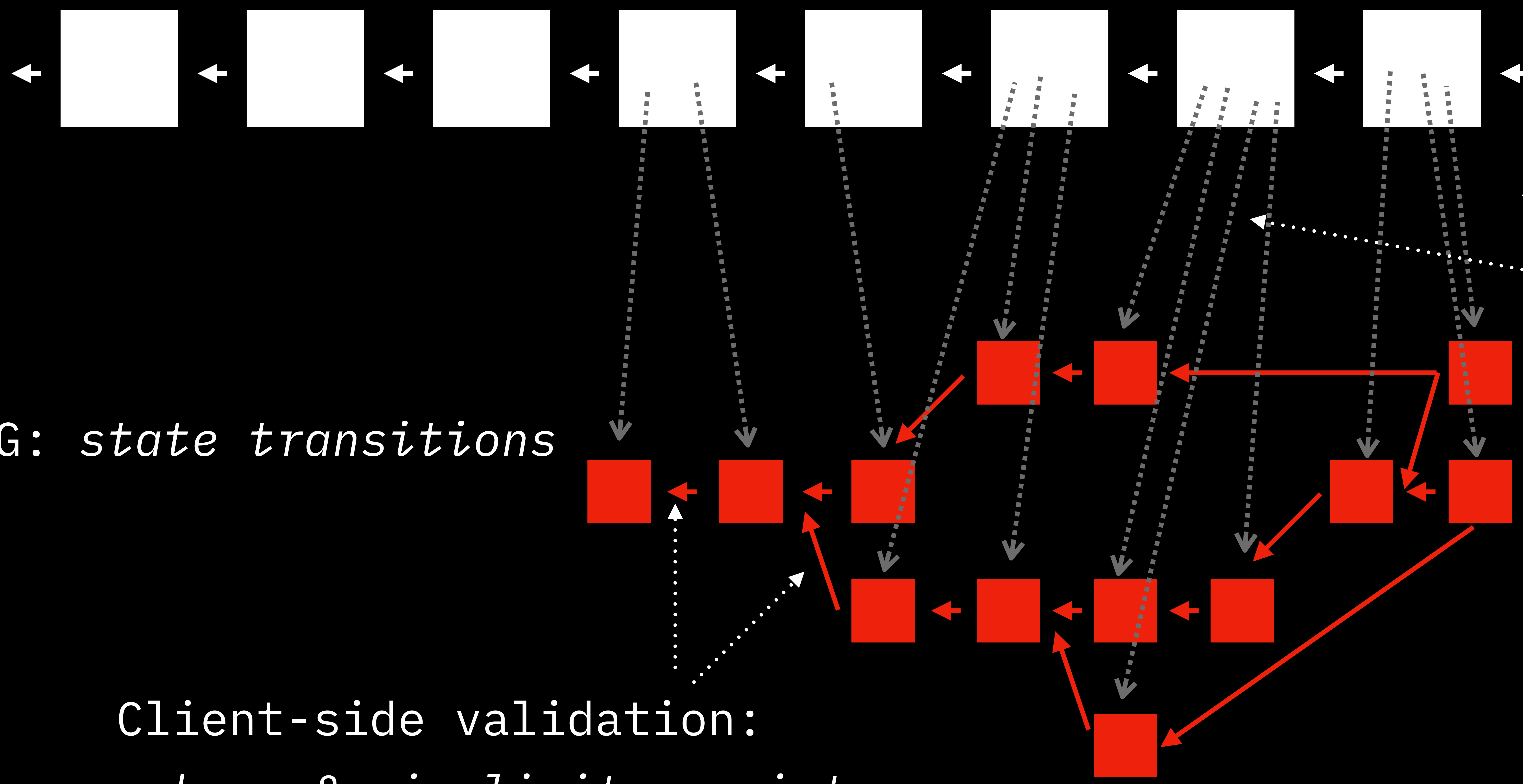
- Commit to new state data when TXO with assignment is spent:
  deterministic bitcoin commitment

- Have standards for **define, serialize & commit to a state** data and rules of its evolution

Sharded DAG on top of Bitcoin Blockchain

Bitcoin transaction graph in blockchain or state channel:

state ownership



Single-use seals:

state assignments

using

Deterministic

bitcoin

commitments:

anti-double

spend

DAG: *state transitions*

Client-side validation:

schema & simplicity scripts

What we need

- Assign state to UTXO: **single-use-seal definition**
  this UTXO Bitcoin script will control the rights ownership

- Commit to new state data when TXO with assignment is spent:
  deterministic bitcoin commitment

- Have standards for **define, serialize & commit to a state** data and rules of its evolution

- Merge multiple assignments and related state changes into a single operation: **multi-message commitments & anchors**

Bitcoin transaction graph
(in blocks or LN channel)

Client-validated data
(in blocks or LN channel)

State assignment

Commitment

Some new data

Some other state data

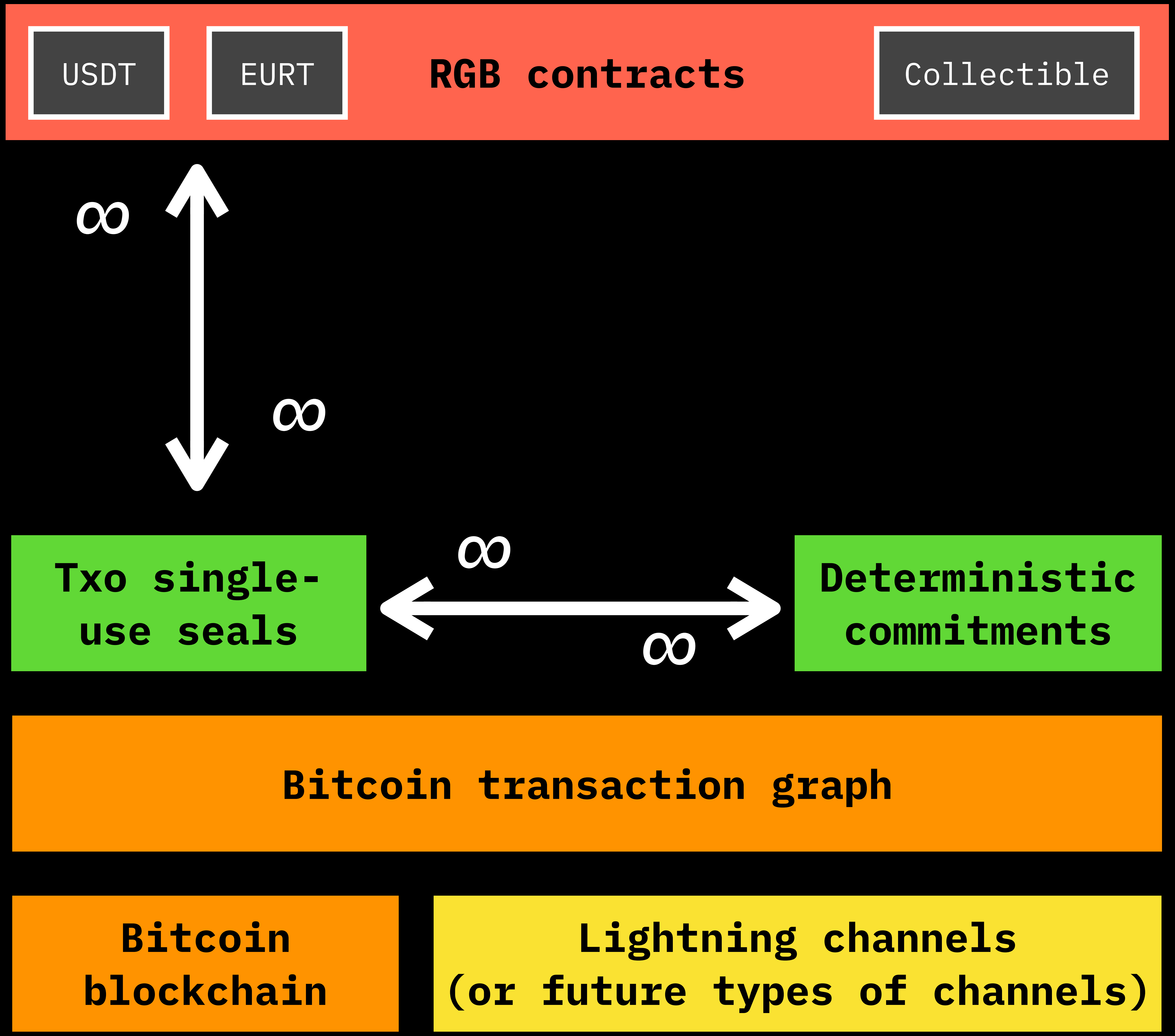
The same TXO definition

State data

Unique TXO definition

RGB & Bitcoin multidimensional relations

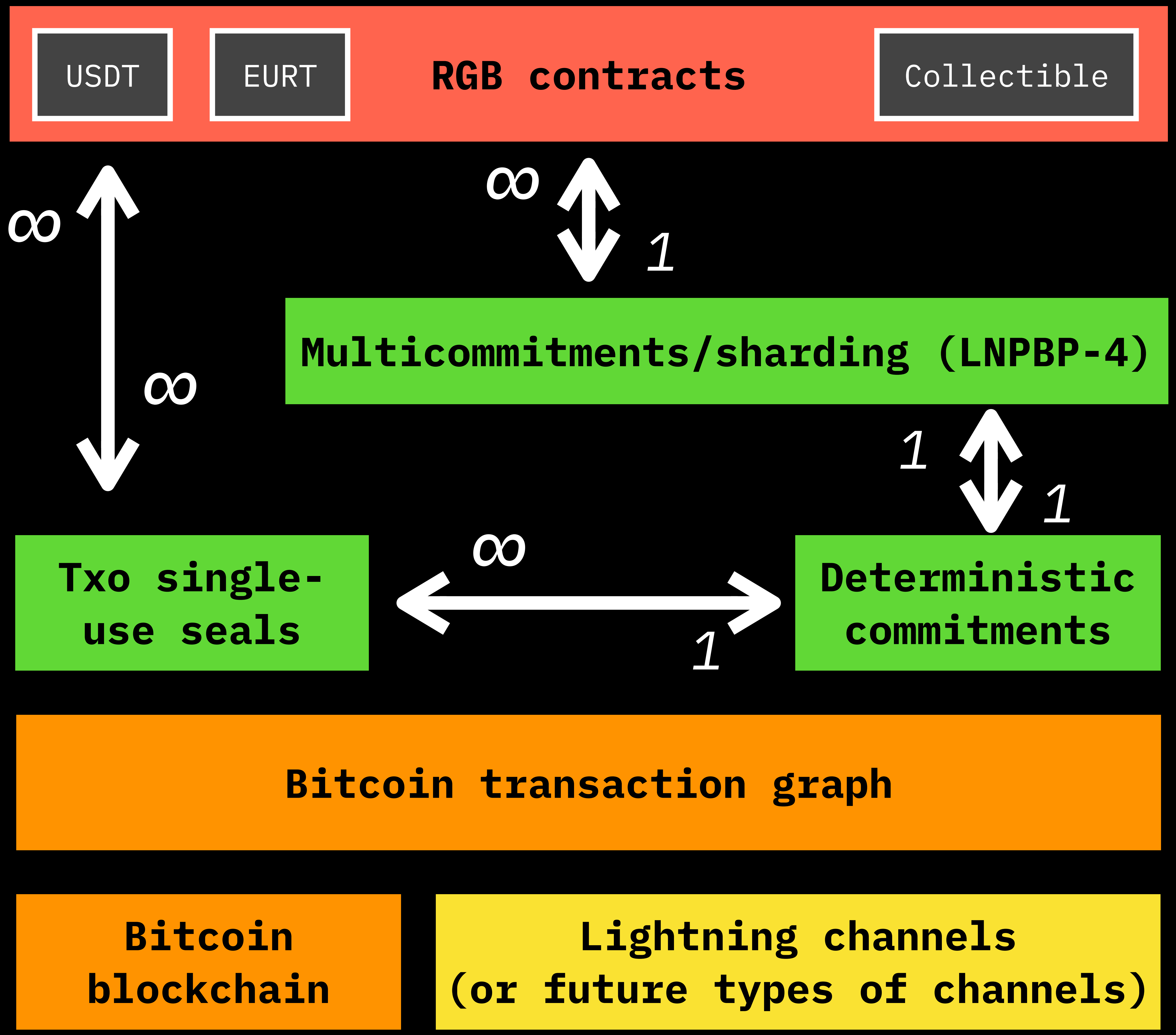
- There may be many RGB contracts issuing many different tokens, identity, collectibles…
- Each asset can be allocated to multiple transaction outputs owned by the same party
- Many different assets may be allocated to the same output
- Some asset may be allocated to the same output many times under different transfer operations...



USDT
EURT
RGB contracts
Collectible
∞
∞
Txo single-use seals
∞
∞
Deterministic commitments
Bitcoin transaction graph
Bitcoin blockchain
Lightning channels (or future types of channels)

RGB & Bitcoin multidimensional relations

- Contract sharding:
  - Isolates histories of different contract without the risk of double-spending
- Requires introduction of
  - “Anchors”, linking many transitions to the same single commitment, closing some set of seals over multiple messages
  - “Stash”: a combination of all contracts with their histories and inter-contract anchors kept by an owner (wallet)


USDT
EURT
RGB contracts
Collectible
∞
∞
∞
1
Multicommitments/sharding (LNPBP-4)
1
1
∞
Txo single-use seals
1
Deterministic commitments
Bitcoin transaction graph
Bitcoin blockchain
Lightning channels (or future types of channels)

Putting layers together

Code layers

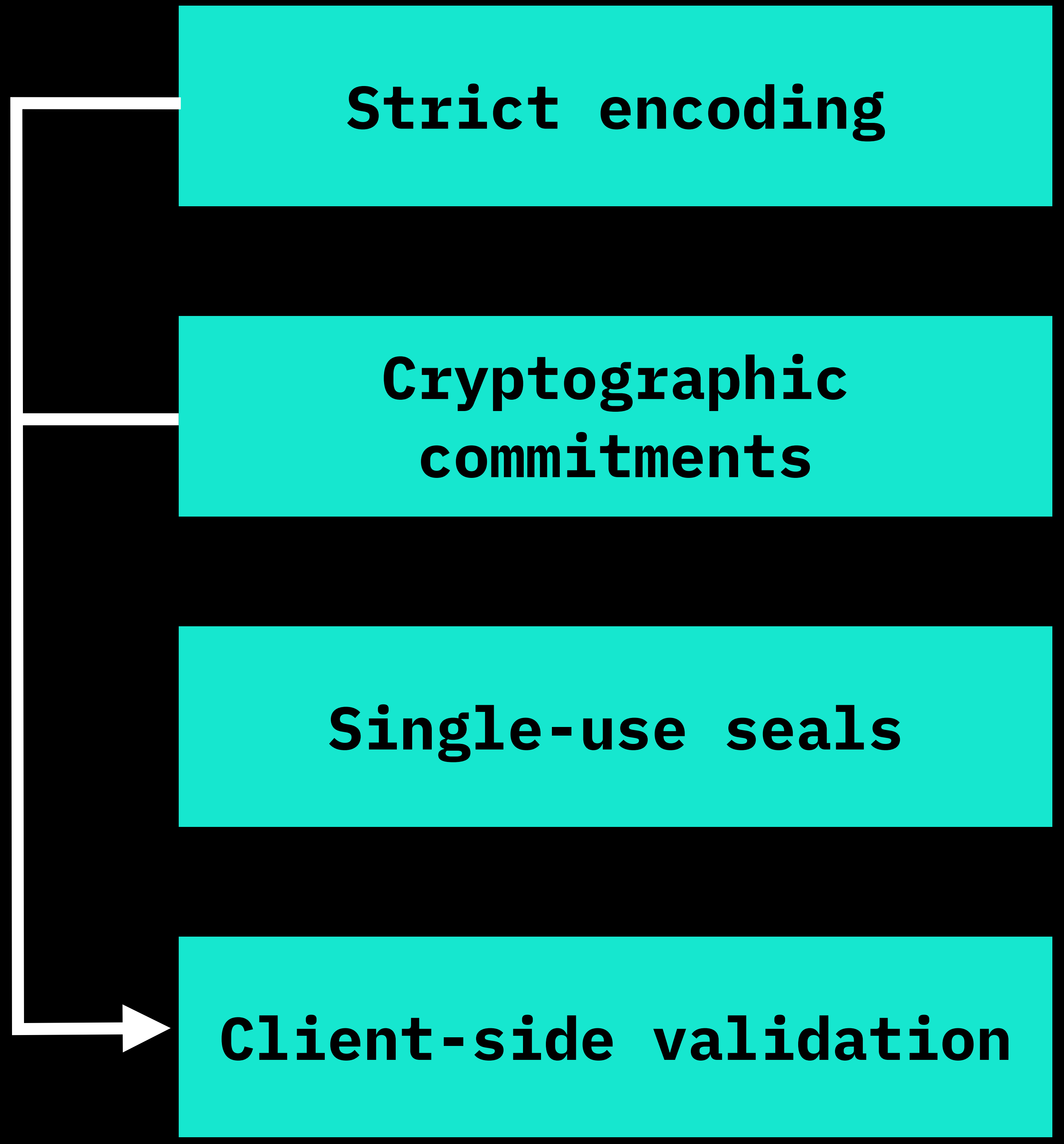
- LNP/BP Core Library
  - Generic paradigms for building layers
  - Extensions to Bitcoin protocol
  - Extensions to Lightning protocol
  - RGB core data structures
- RGB Node
  - Schema-specific functionality
  - Integration API with Bitcoin network & Lightning Network
  - No private key management or direct Bitcoin/LN integration!
  - Wallet API

From paradigms through bitcoin to RGB

Paradigms

Bitcoin tx graph

RGB



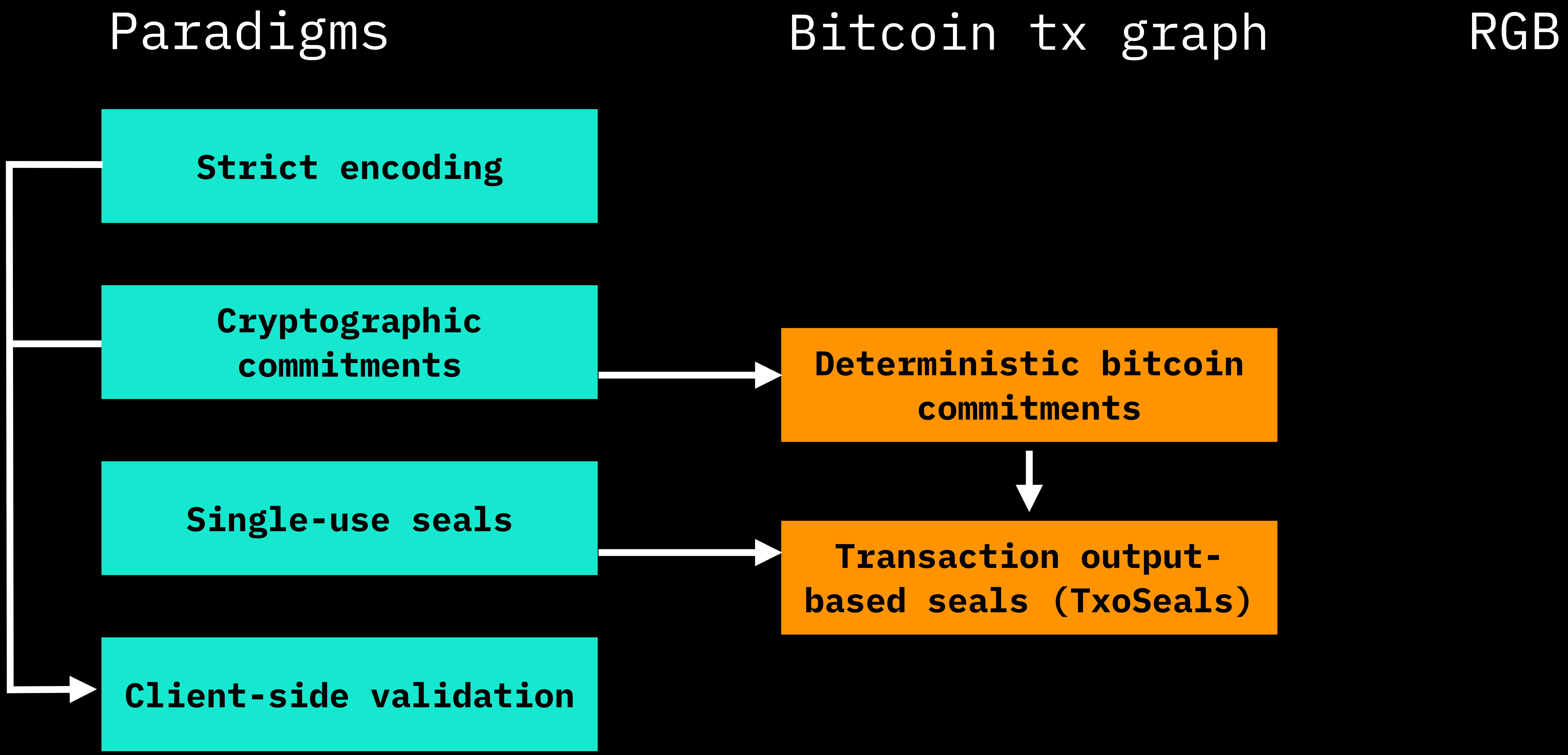
Strict encoding

Cryptographic commitments

Single-use seals

Client-side validation

From paradigms through bitcoin to RGB



Paradigms

Bitcoin tx graph

RGB

Strict encoding

Cryptographic commitments

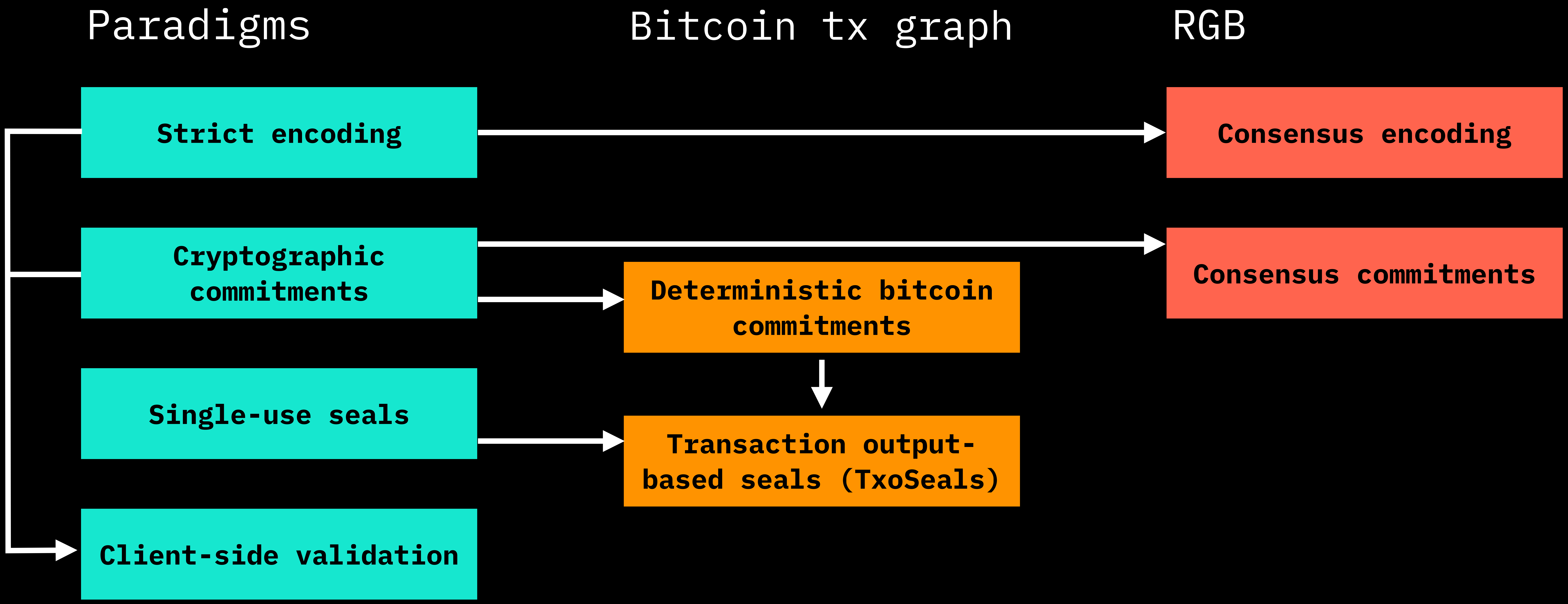
Single-use seals

Client-side validation

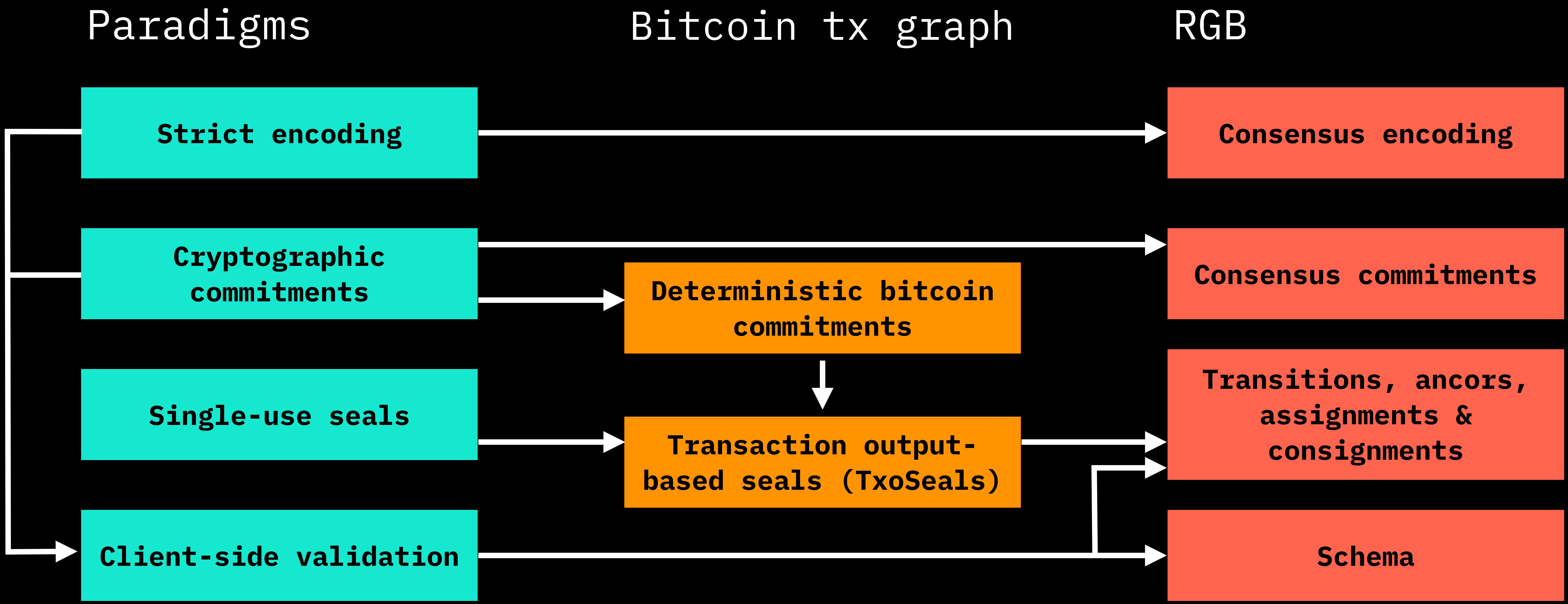
Deterministic bitcoin commitments

Transaction output-based seals (TxoSeals)

From paradigms through bitcoin to RGB

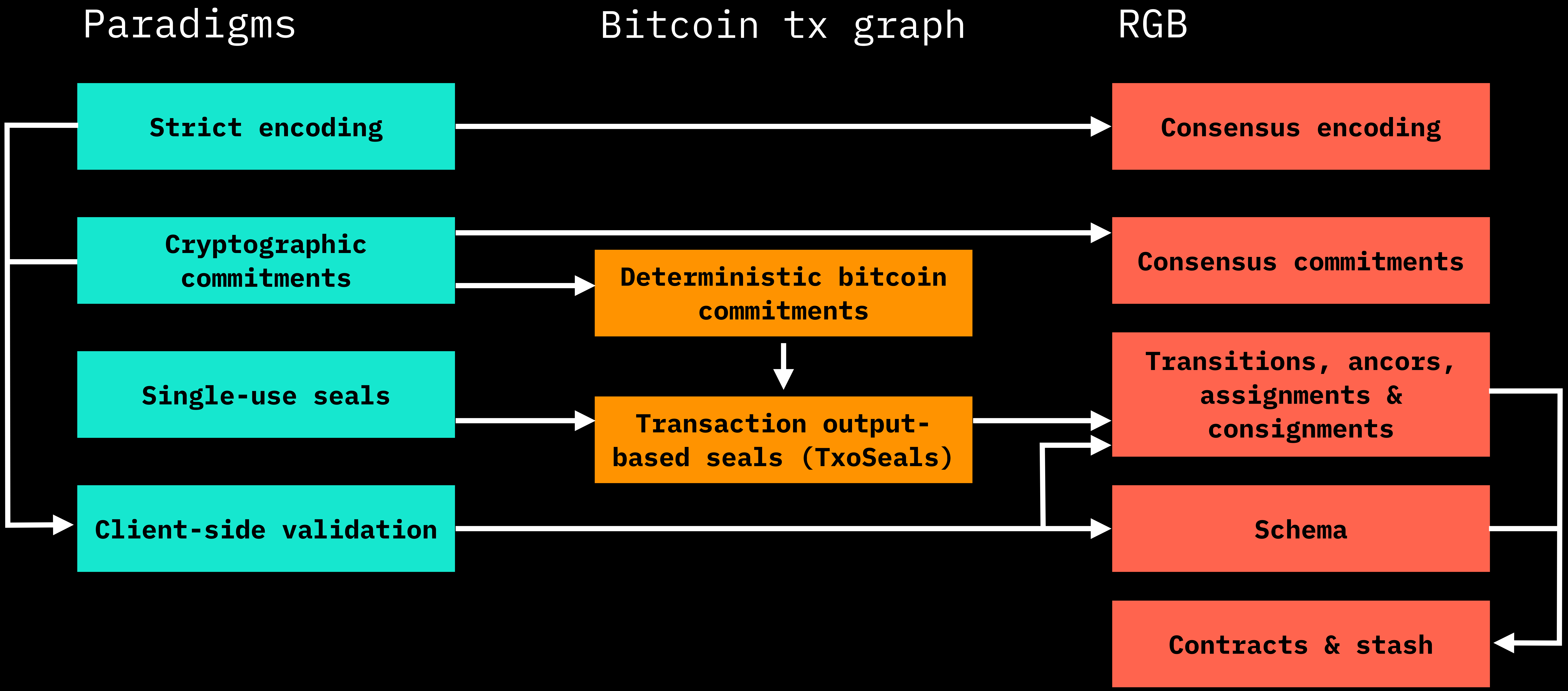


From paradigms through bitcoin to RGB



Paradigms
Bitcoin tx graph
RGB
Strict encoding
Cryptographic commitments
Single-use seals
Client-side validation
Deterministic bitcoin commitments
Transaction output-based seals (TxoSeals)
Consensus encoding
Consensus commitments
Transitions, ancors, assignments & consignments
Schema

From paradigms through bitcoin to RGB


Paradigms
Bitcoin tx graph
RGB
Strict encoding
Cryptographic commitments
Single-use seals
Client-side validation
Deterministic bitcoin commitments
Transaction output-based seals (TxoSeals)
Consensus encoding
Consensus commitments
Transitions, ancors, assignments & consignments
Schema
Contracts & stash

RGB Schema

- “Blueprints”/standards for constructing RGB contracts may think as of “ERC* of RGB”
- “Fungible asset” or “collectible” is a schema
- Issuer defines issuance contract, but for being supported by wallets/exchanges it must stick to (“validate against”) particular schema
- Actual wallets or exchanges will always use schema-based libraries (like “RGB fungible assets”, “RGB collectibles”), and not complex & universal core RGB library

Bitcoin smart-contracts:

- Bitcoin script: bare, hashed and Taproot
  - Multisigs, state channels, swaps…
- Scriptless scripts: private, less footprint

ownership

- RGB
  - Using schema & simplicity language
  - No blockchain footprint
  - Confidential
  - Nearly fully Turing-complete

state validation

Core RGB smart contract components

- **State**: must be assigned to a *single-use-seal*, linking to bitcoin ownership and double-spend protection (with *deterministic bitcoin commitments*)

- **Schema**: defines which state can be out there and how it can evolve in time

- **Scripts**: must define state validation rules with *Simplicity language*

Smart contracts

	“Ethereum-style”	
● **Parties of the agreement**	loosely defined	
● **Agreement:**	Bockchain-stored contract + ABI file	
- **Current state**	blockchain-stored data: ★ publicly visible ★ non-confidential ★ non scalable ★ no 2nd layer support	
- **State change rules**	custom EVM code	
- **Ownership rights**		
● **Mutability**	Pseudo-immutable: immutable in promice, censored my miners & creators in fact	

Pure blockchain/layer 1 approach is wrong:

- Mixing ***code***, ***ownership*** and ***access rights*** into a single layer (“blockchain”)

- which is inherently ***unscalable*** and well-trackable (***anti-privacy***) since VERIFICATION is needed by the whole world

- With Turing-complete ***code*** operating at the same level, ***compromising security***

- Running ***non-censorship-resistant*** consensus algorithms (PoS, PoW forks with small hashing power)

Smart contracts

	“Ethereum-style”	RGB
● **Parties of the agreement**	loosely defined	issuer and current owners
● **Agreement:**	Bockchain-stored contract + ABI file	Client-stored contract genesis + state transitions
- **Current state**	blockchain-stored data: ★ publicly visible ★ non-confidential ★ non scalable ★ no 2nd layer support	client-stored data: ★ no chain analysis ★ confidential ★ scalable ★ 2nd layer support
- **State change rules**	custom EVM code	schema & simplicity script
- **Ownership rights**		bitcoin script
● **Mutability**	Pseudo-immutable: immutable in promice, censored my miners & creators in fact	Well-defined mutability rights at genesis & schema level by issuer Mutable by new owners within the scope of rules