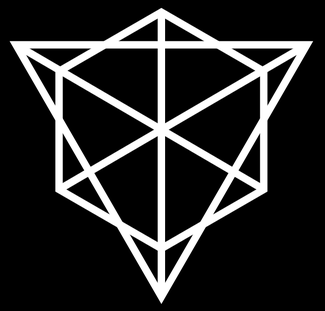
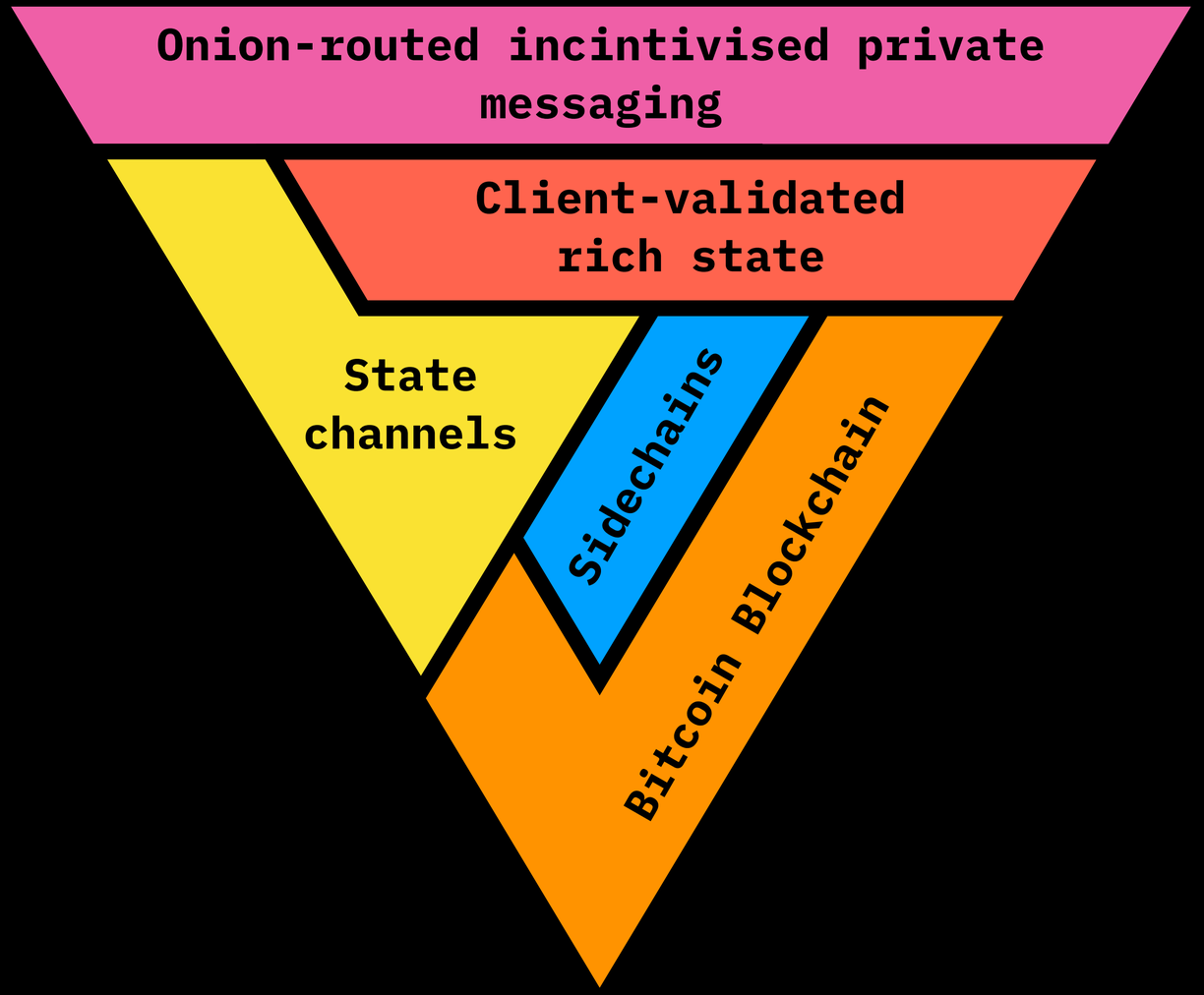
RGB, SPECTRUM, STORM:
LESSONS LEARNED FROM AN APPLIED LNP/BP TECH

Dr Maxim Orlovsky

Chief **Engineering** Officer (CEO) @ Pandora Core AG, Switzerland



Storage, Messaging, Assets, Computing



2 types of extending LN functionality

- Extending payment LN functionality, like:
  - routing algorithms
  - channel rebalancing
  - submarine swaps
  - ...
- Layers on top of LN, utilising:
  - onion-routed messaging protocol
  - gossip protocol
  - commitment transactions for commitments :)

LNP/BP applications leveraging LN

- **RGB**: digital assets (securities, collectibles)
- **Spectrum**: DEX Lightning Network solution
- **Storm**: trustless escrowed storage & messaging
- **Prometheus**: high-load parallel computing

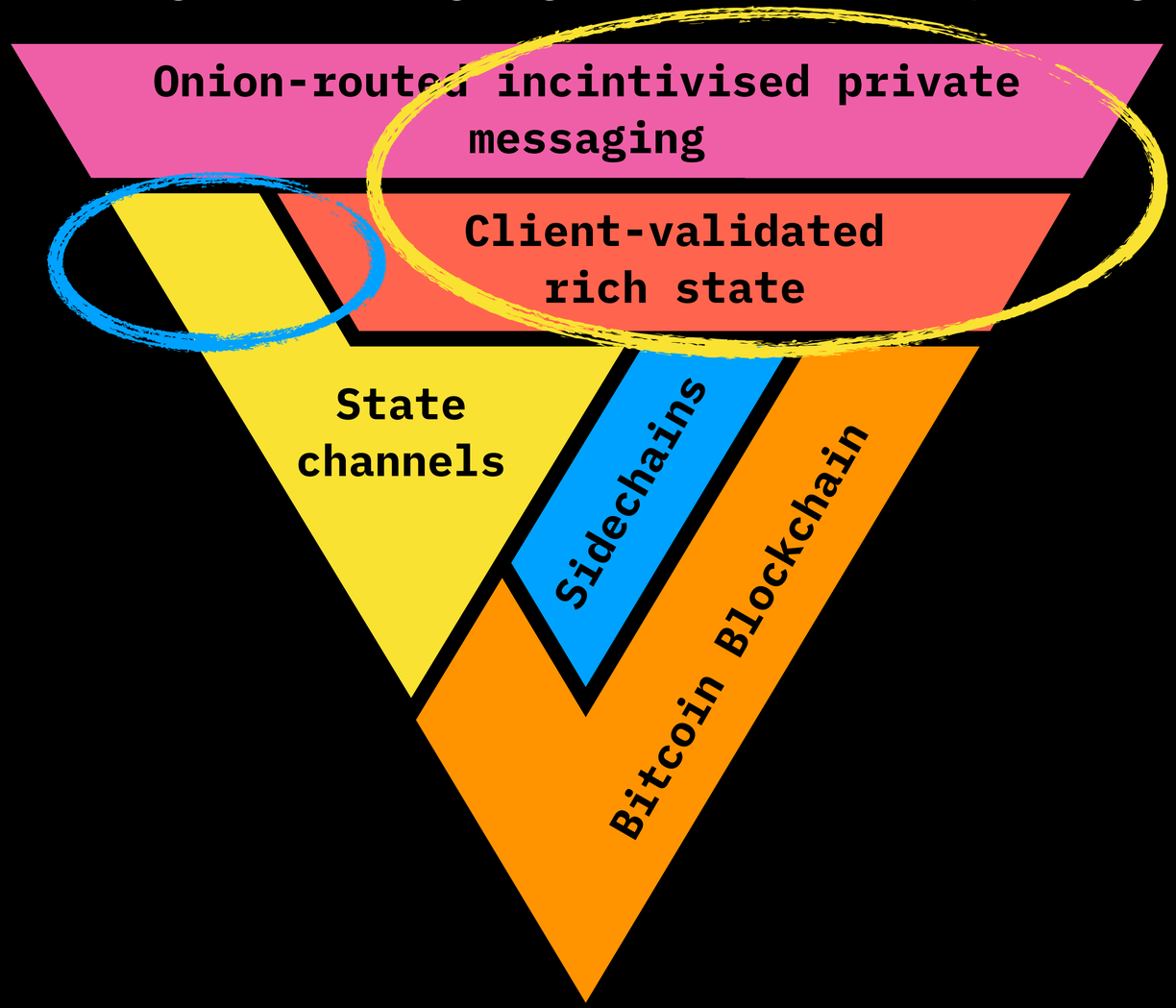
RGB & Spectrum

- Multiple **privacy** enhancements:
  - Client-side validation: zero L1/L2 visibility
  - Zero-knowledge bulletproofs for transfers
  - Hardened LN network transfer analysis
- Enhances **LNP/BP adoption**, especially for stable coins
- New **node monetisation** options
- Potential **off-chain BTC** transfer L2/L3 solution

L3 building components

- Can be done with L1 alone, but needs LN to scale:
  - Cryptographic commitments
  - Client-side validated state
- Can’t be done without Lightning Network:
  - Onion-routed p2p messaging
  - Network-wide gossip messaging

Storage, Messaging, Assets, Computing



What is required to enable L3 solutions

- Cryptographic commitments:
  - public key tweaking in channel transactions
  - fee adjustments in channel transactions

dr-orlovsky commented 3 days ago • edited

Member

Commit to a value with public keys (C2VPK): Generalized commitments based on public key tweaking

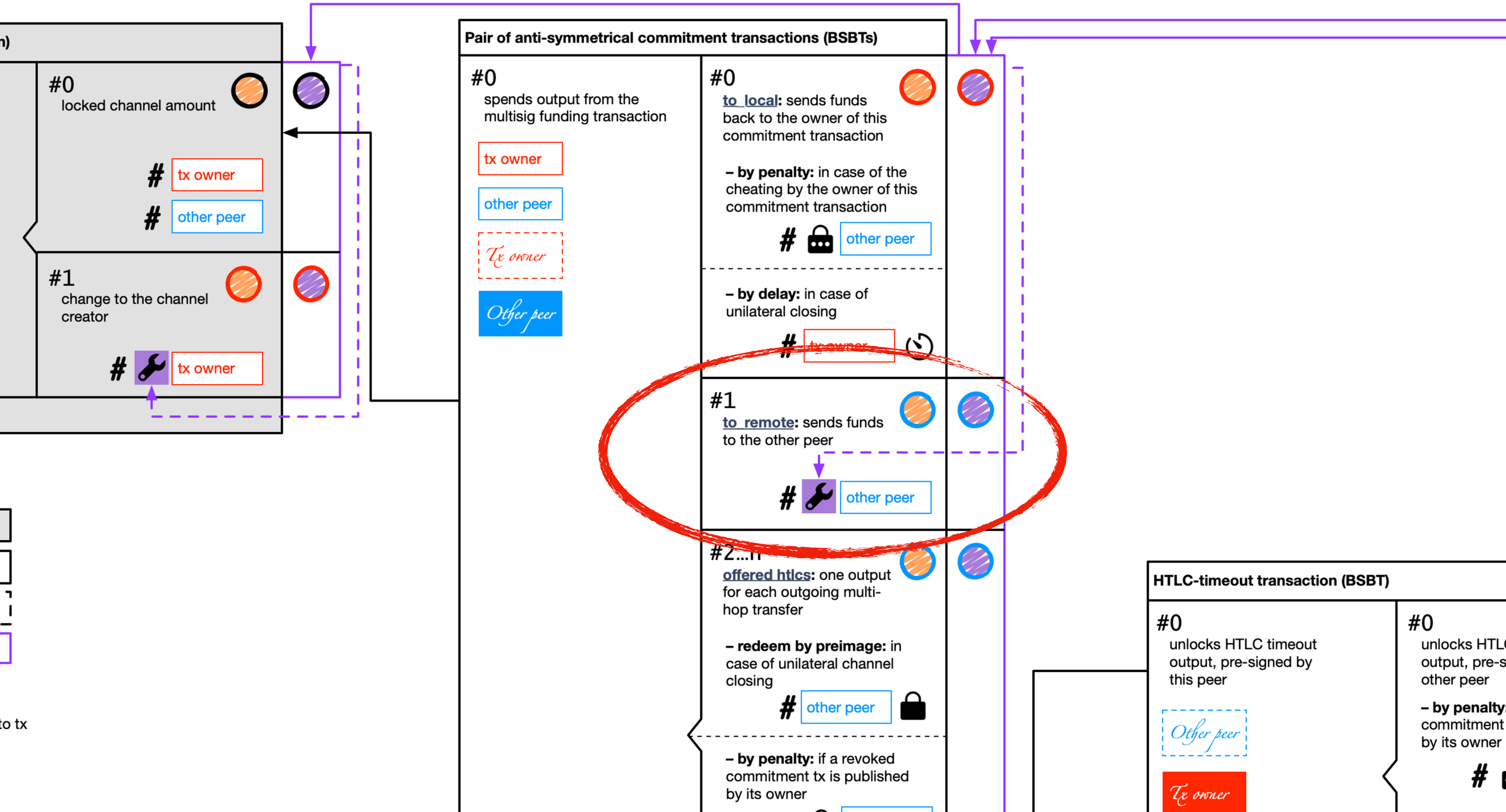
Motivation

Lightning network channel construction already lacks non-P2WSH outputs in its HTLC-success and HTLC-timeout transactions, making it impossible to utilize modern RGB and single use seal data for both milti-hop and direct payments/state updates. The specifics of the protocol is that it does require creation of HTLC-based output and related HTLC-output spending transactions even for a direct payments, so the support of P2WSH commitments becomes required issue to operate RGB assets over Lightning Network.

Moreover, the next changes to BOTL's assumes that commitment transactions will also be left without P2PKH outputs, since `to_remote` output will require CSV script option in order to fix existing misalignment of incentives during the channel close. Thus, it is required to enable P2WSH pay-to-contract commitments.

This specification proposes a generalized way to make a cryptographic commitments bases on pay-to-contract-style public key tweaking for any kind of transaction output, namely:

- legacy P2PK
- OP_RETURN and non-standard P2S
- P2(W)PKH
- P2(W)SH
- P2WPKH and P2WSH wrapped into P2SH


n)
#0
locked channel amount
tx owner
other peer
#1
change to the channel creator
tx owner
Pair of anti-symmetrical commitment transactions (BSBTs)
#0
spends output from the multisig funding transaction
tx owner
other peer
Tx owner
Other peer
#0
to_local: sends funds back to the owner of this commitment transaction
– by penalty: in case of the cheating by the owner of this commitment transaction
other peer
– by delay: in case of unilateral closing
tx owner
#1
to_remote: sends funds to the other peer
other peer
#2...n
offered htlcs: one output for each outgoing multi-hop transfer
– redeem by preimage: in case of unilateral channel closing
other peer
– by penalty: if a revoked commitment tx is published by its owner
HTLC-timeout transaction (BSBT)
#0
unlocks HTLC timeout output, pre-signed by this peer
Other peer
Tx owner
#0
unlocks HTLC
output, pre-s
other peer
– by penalty
commitment
by its owner
to tx

What is required to enable L3 solutions

- Cryptographic commitments:
  - public key tweaking in channel transactions
  - fee adjustments in channel transactions
- Simple TLV extensions in all LN messages
  - gossip protocol for announcing client-validated state
  - onion routing as a generic messaging protocol
  - p2p messages to agree on the client-validated state parameters
- Generic LN node plugin extension standard supporting the above

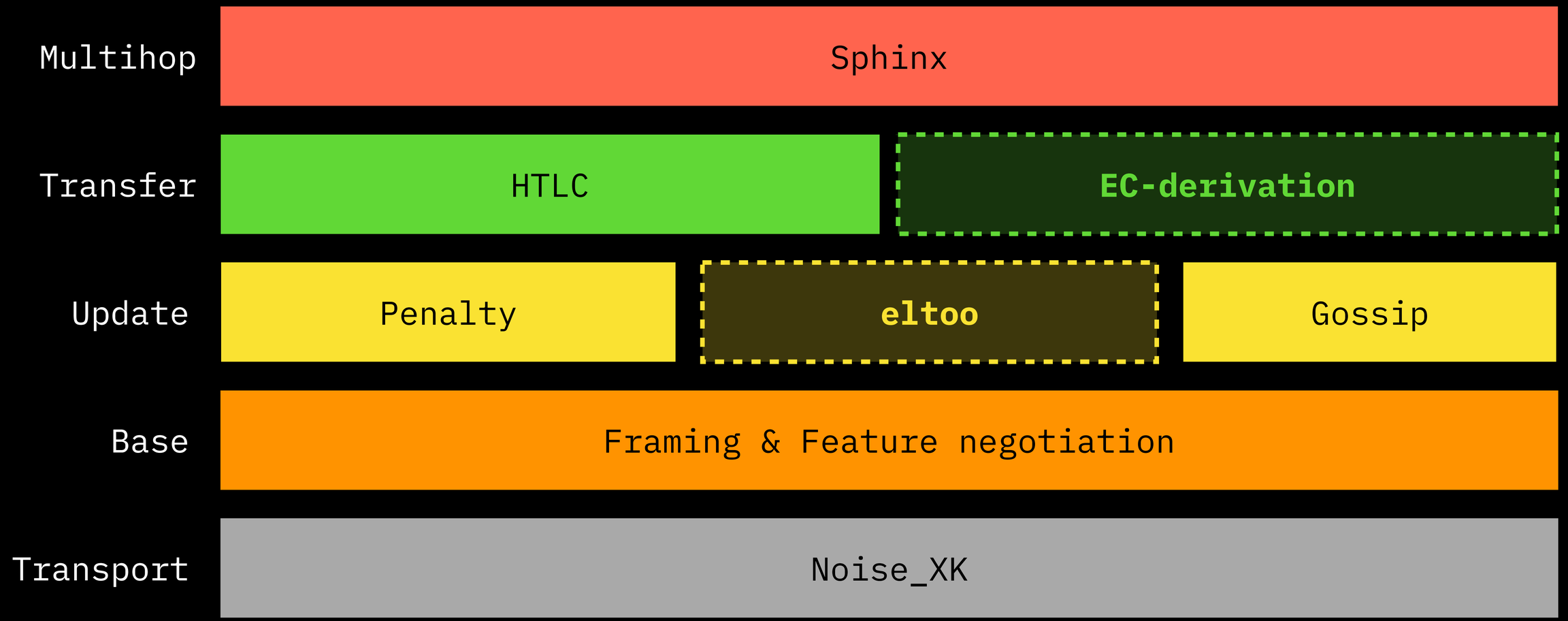
IS LIGHTNING NETWORK READY FOR THIS?

NO

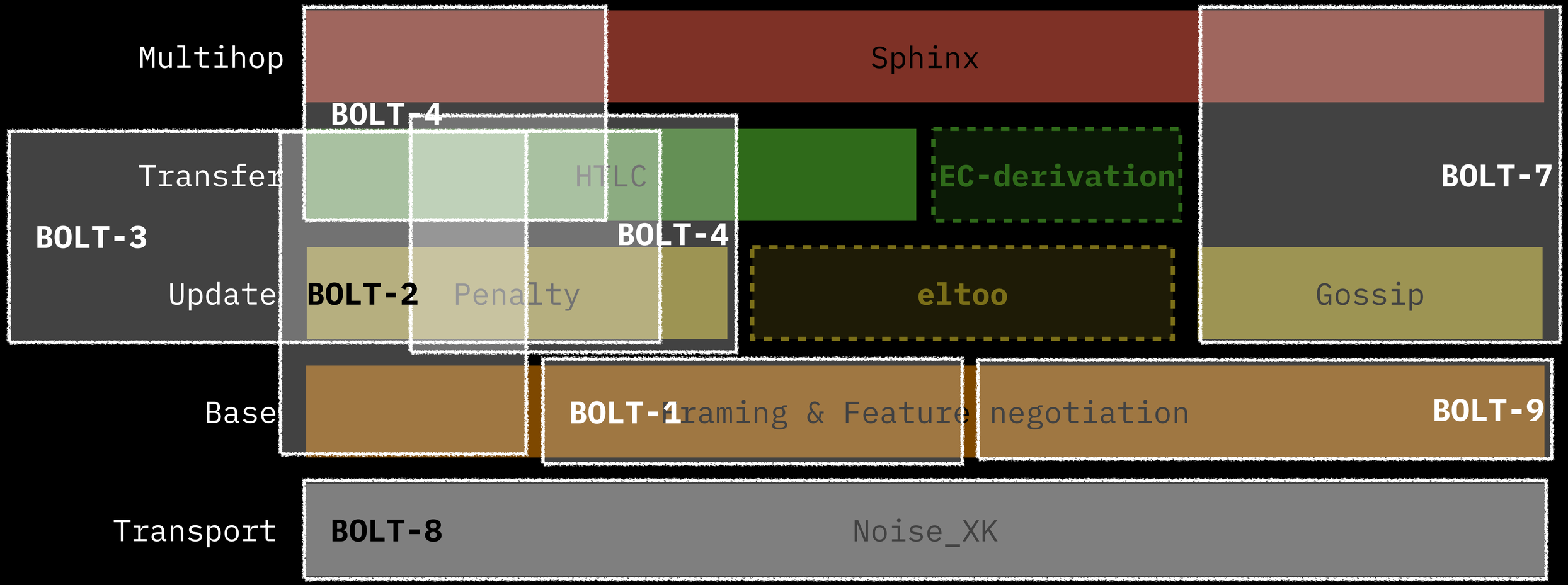
PROBLEM I
SPECIFICATIONS

Lightning Network Architecture

after Christian Decker

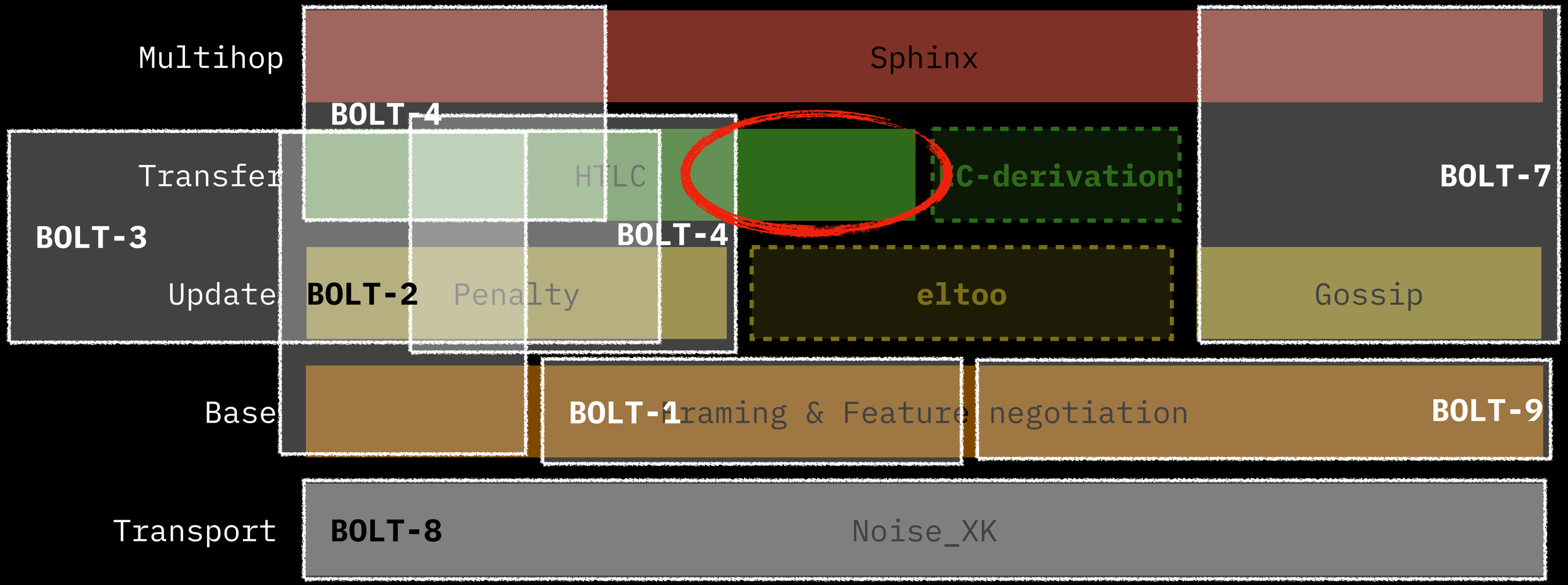


Real BOLT Specifications



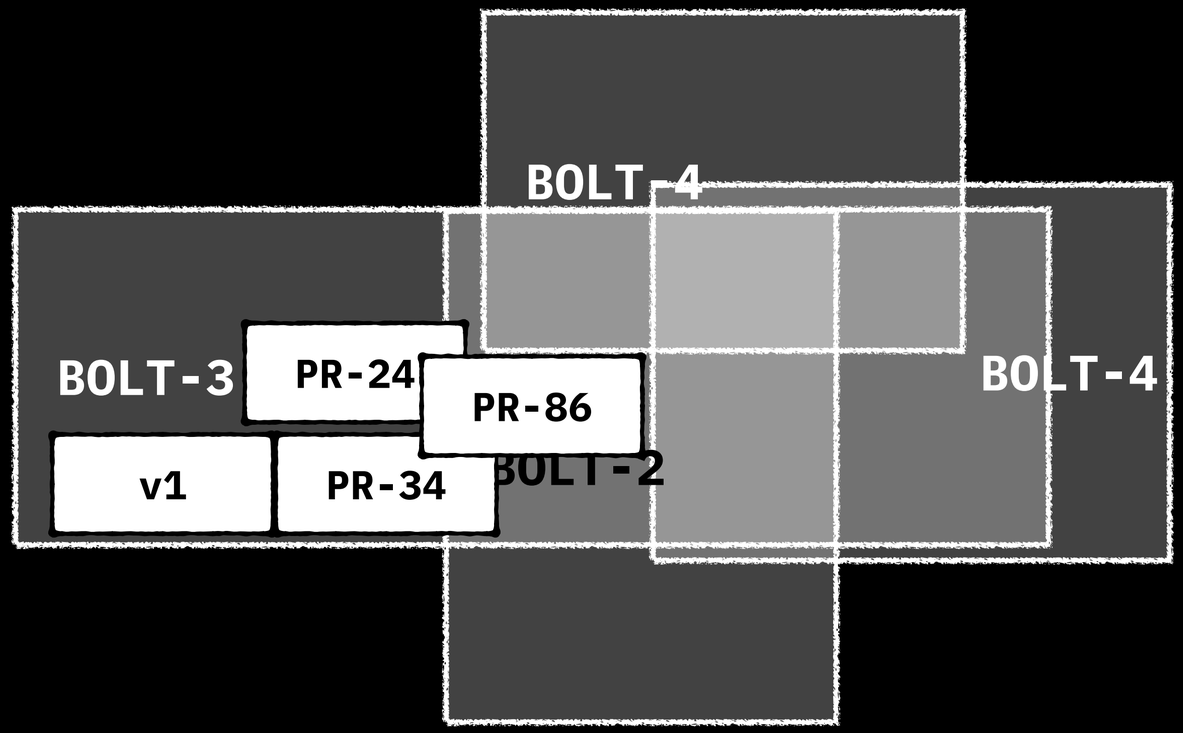
With the current specification approach it is impossible to reconstruct even such simple things as HTLC sequence diagram that can simultaneously cover channel transaction updates and messaging between the nodes for multi-hop payments

BOLTs coverage for real LN is lacunar



Specification Timechain

of healthy man



of ~~smoker~~ BOLT

Why proper abstraction _is_ important

- Security
- Interoperability
- Software upgradability
- New features

PROBLEM III

LN IS NOT DESIGNED WITH L3+ IN MIND

The missing parts:

- No proper extensibility for LN in specs; it’s up to node implementation to decide:
  - ➡ No way to create cross-node extensions
- No proper layering of the LN tech itself
- No clear way to introduce features:
  - ➡ even with feature flags, it’s unclear how to introduce new TLV types

PROBLEM II
NOT EVEN A BETA YET

Things that are going to change:

- EC-derived secrets instead of HTLC
  (Schnorr’s signatures required)
- American call option problem
  https://lists.linuxfoundation.org/pipermail/lightning-dev/2018-December/001752.html
- to_remote output: P2WPKH->P2WSH with CSV-lock
- eltoo after SIGHASH_NOINPUT
- Gossip protocol changes improving traffic issues

+ No fixed specs that LN nodes are stick to

Many of those require Bitcoin softforks

It will take years for LN protocols to become stable

Before that, using LN in L3 means constant re-implementation and re-integration

Our aim is to fix some subset of LN to be able to provide robust operations for LN-based payments with bitcoin and USTD, which should boost adoption of censorship-resistant money

Proposed solution:

- Production-ready subnet for LN with stable functionality signalled by a feature flag
- Strict standards-following basic LN node with all new changes and enhancements moved into separate external code modules (plugins, alternative channel daemons implementations)

LNP/BP Standards (outside of BIP scope)

GitHub, Inc. (US) | https://github.com/lnp-bp/lnpbps

LNP-BP / lnpbps

Unwatch 2 | Unstar 3 | Fork 0

Code | Issues 2 | Pull requests 0 | Projects 0 | Wiki | Security | Insights | Settings

LNP/BP Specifications

Edit

bitcoin | lightning-network | decentralization | distributed-systems | privacy | cryptography | Manage topics

3 commits | 1 branch | 0 releases | 1 contributor

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

dr-orlovsky LNPBPs-0001 (early draft): Cryptographic commitments with public key ... — Latest commit 55b573c 17 days ago

assets	README: Project description, inclusion criteria, layers and initial l...	17 days ago
.gitignore	README: Project description, inclusion criteria, layers and initial l...	17 days ago
README.md	LNPBPs-0001 (early draft): Cryptographic commitments with public key ...	17 days ago
lnpbps-0001.md	LNPBPs-0001 (early draft): Cryptographic commitments with public key ...	17 days ago

README.md

LNP/BP Specifications

LNP/BP stands for "Bitcoin Protocol / Lightning Network Protocol". This set of specifications covers standards & best practices for Layer 2, 3 solutions (and above) in cases when they do not require soft- or hard-forks on the Bitcoin blockchain level and are not directly related to issues covered in Lightning Network RFCs (BOLTs).

https://github.com/lnp-bp

Number	Layer	Field	Title	Owner	Type	Status
1	Transaction (1)	Cryptographic primitives	Cryptographic commitments with public key tweaking	n/a	Standard	Draft
2	Transaction DAG (2)	Client-side validation	Simple single-use seal for LNP/BP	n/a	Standard	Draft
3	Offchain data (3)	Consensus rules	State history directed acyclic graphs on Bitcoin	n/a	Standard	Draft
4	Offchain data (3)	Serialization	Serialization for state history DAGs, LNPBPs-4	n/a	Standard	Draft
5	Offchain metadata (4)	Serialization	Schemata for rich state	n/a	Standard	Draft
6	Application (5)	Assets	RGB, part 1: Fungible centrally-issued assets with client-side validation	n/a	Standard	Draft
7	Offchain data & metadata (3-4)	P2P messaging	State announcements for Lightning Network gossip protocol	n/a	Standard	Draft
8	Offchain data & metadata (3-4)	P2P messaging	State updates over Lightning Network onion messaging	n/a	Standard	Draft
9	Application (5)	DEX/DMP	Spectrum: decentralized market / exchange over Lightning Network	n/a	Standard	Draft
10	Offchain metadata (4)	Assets	RGB, part 2: Zero-knowledge proofs for asset transfers	n/a	Standard	Draft

To find out more

LNP/BP Standards: **https://github.com/lnp-bp/lnpbps**

- RGB & Spectrum (digital assets & DEX): https://github.com/rgb-org/spec
- Storm (storage & messaging): https://github.com/storm-org/storm-spec
- Prometheus (high-load computing): https://github.com/pandoracore/prometheus-spec

Pandora Core AG, works on implementation of these technologies, aiming to bring scalability and trustlessness into distributed storage and computing

Other participants of the protocols development & sponsorship include:
Giacomo Zucco, Peter Todd, John Carvalho, Blockstream, inbitcoin, Chainside, Bitfinex, Poseidon Group, Fulgur Ventures, Hyperdivision and many others

We are welcoming you to join the work on these projects!