LNP/BP Standards Association

RGB

Roadmap: a way towards release

Dr Maxim Orlovsky,

Pandora Core AG

With bitcoin, lightning network &
client-side-validation it is possible to do
everything which was promised by Ethereum/
“next gen blockchain” projects

– but in scalable, censorship-resistant and
private way

RGB: smart contracts for LNP/BP

- **Bearer rights**, not a registry as in blockchain
- **Confidentiality**, brought to extreme
- **Scalable**, since off chain: layer 2 & 3, can operate on top of LN
- **Bitcoin programmability** layer

Proper smart contract systems

- Bitcoin Script for ownership (Bitcoin, data, assets)
- Scriptless-script based solutions for futures DLCs
- RGB for complex scenarious and client-validated state

RGB smart contracts are different

- No “utility token”, just money (Bitcoin and Bitcoin over Lightning)
- Let’s reuse best consensus of today: Bitcoin PoW
- Declarative & functional business logic: let’s avoid EVM/WASM mistakes
- No global state: lets make it scalable & functional
- Contracts are sharded from the day 1

RGB: a long way since 2016

- 2016: Peter Todd coined concepts of client-side-validation and single-use-seals
- 2016-2018: Giacomo Zucco envisioned their application for better fungible assets protocols on top of Bitcoin & LN and did first PoC with Alekos Fillini
- 2019-2021: Maxim Orlovsky extended protocol for real-world applications and for handling full-fledged smart contracts; made full Bitcoin & LN implementation

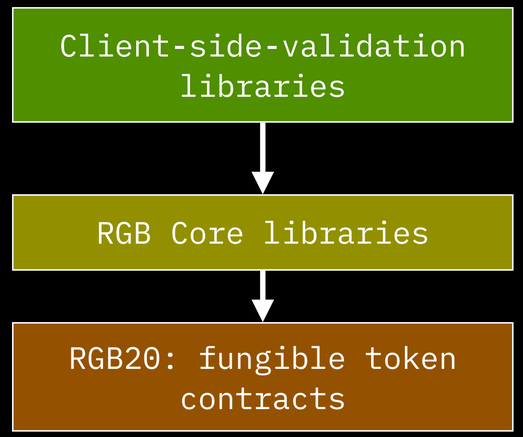
No ICO, token issuance throughout all of these years;
all protocols & standards are open source under CC0, MIT & Apache2

RGB today:

- Lead by LNP/BP Association (non-profit, Swiss-based, “no-org”)
- 25+ contributors to standards and source code
- ~1 million of lines of rust 🦀 code
  (most went to the lower-level bitcoin & LN libraries)
- Consumer-facing products are developed by for-profit orgs:
  - Pandora Core AG
  - Condensat Bank AG
  - Nodl
  - HodlHodl
  - Blockchain of Things
  - … and many others

Color legend

100%	Fully completed
75%	Requires update
50%	Alpha ver / PoC
25%	Concept, needs impl
0%	Requires research
-	Non-existent



Lightning network (LNP):

Bitcoin protocol (BP):

RGB smart contracts:

Internet2:

RGB public “reckless” release

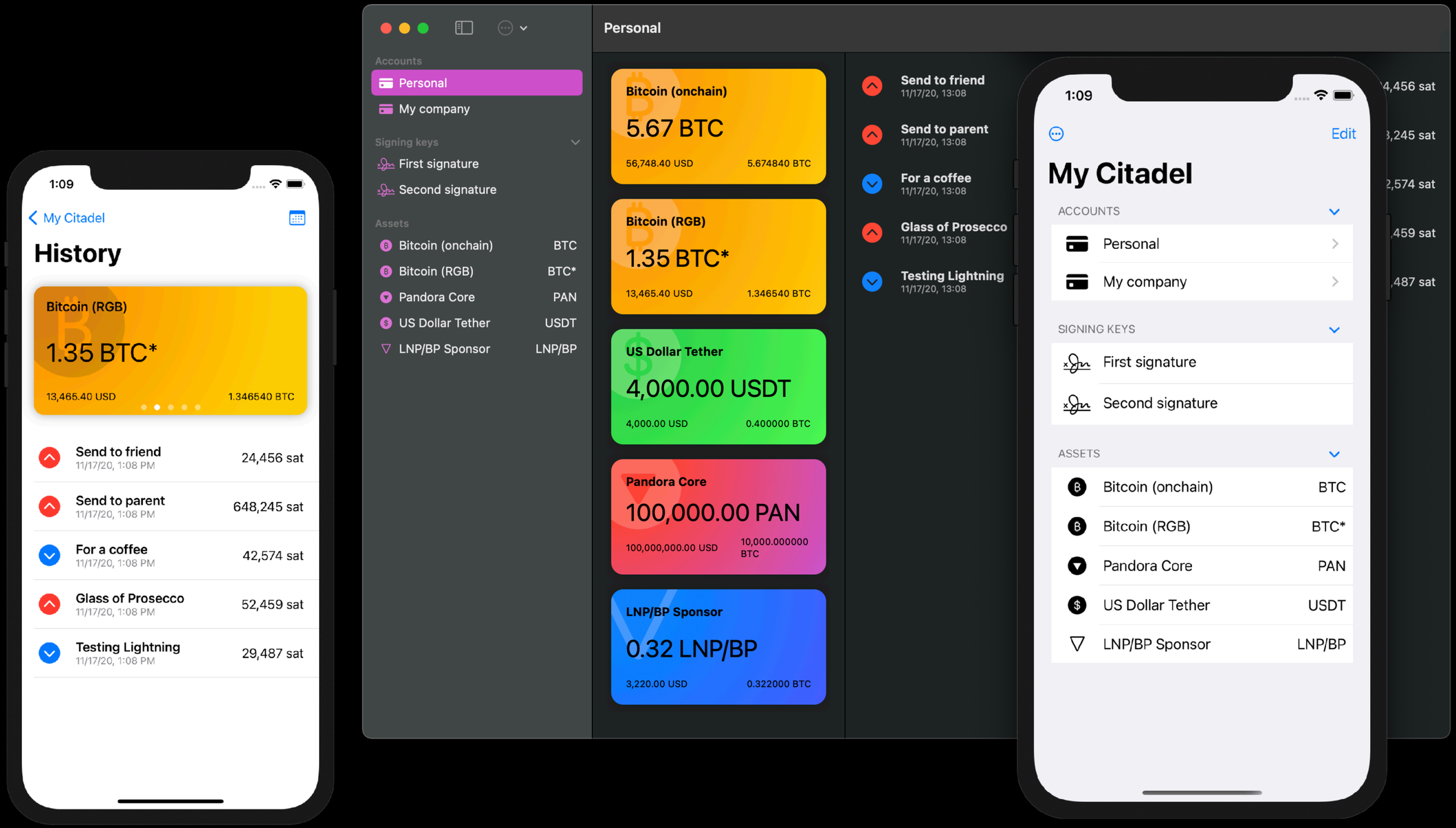
Lightning network (LNP): *Bitcoin protocol (BP):* *RGB smart contracts:* *Internet2:*

RGB-enabled wallets
(MyCitadel, …)

RGB public “reckless” release

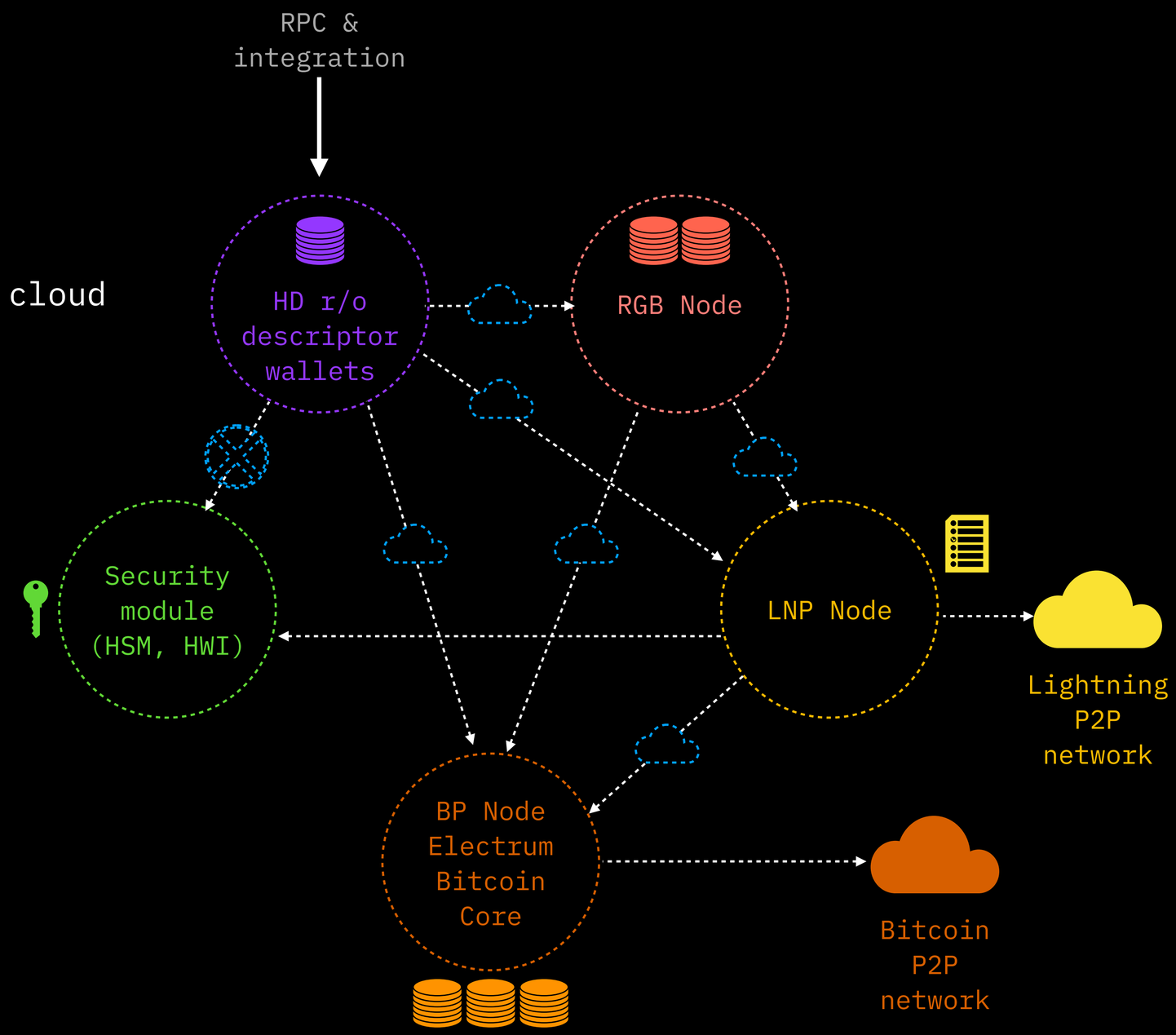
100%	Fully completed
75%	Requires update
50%	Alpha ver / PoC
25%	Concept, needs impl
0%	Requires research

MyCitadel wallet

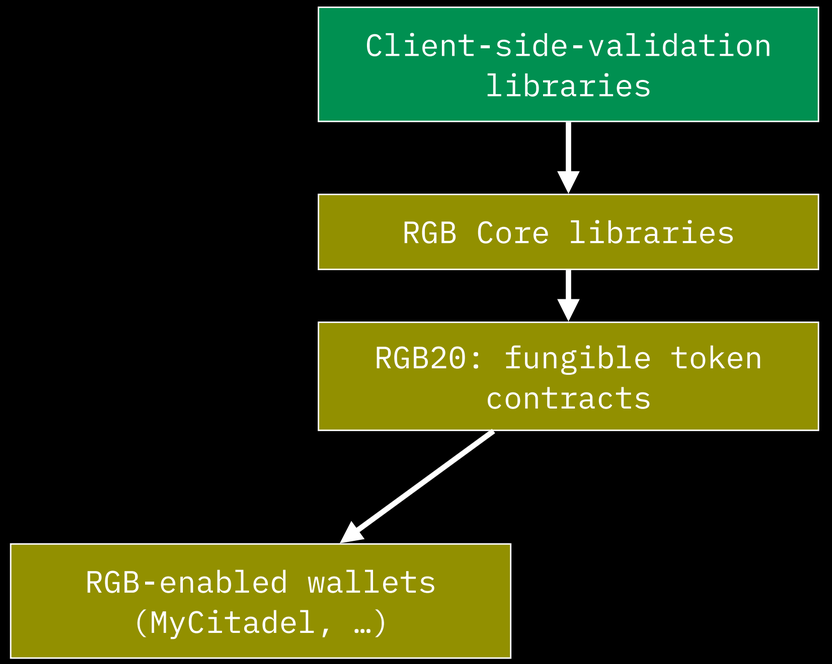


Citadel Runtime

- Embeddable into any wallet: mobile, desktop, command-line, server-side, cloud based
- RGB
- LN
- Taproot / schnorr
- Multisigs & miniscript



Lightning network (LNP): *Bitcoin protocol (BP):* *RGB smart contracts:* *Internet2:*



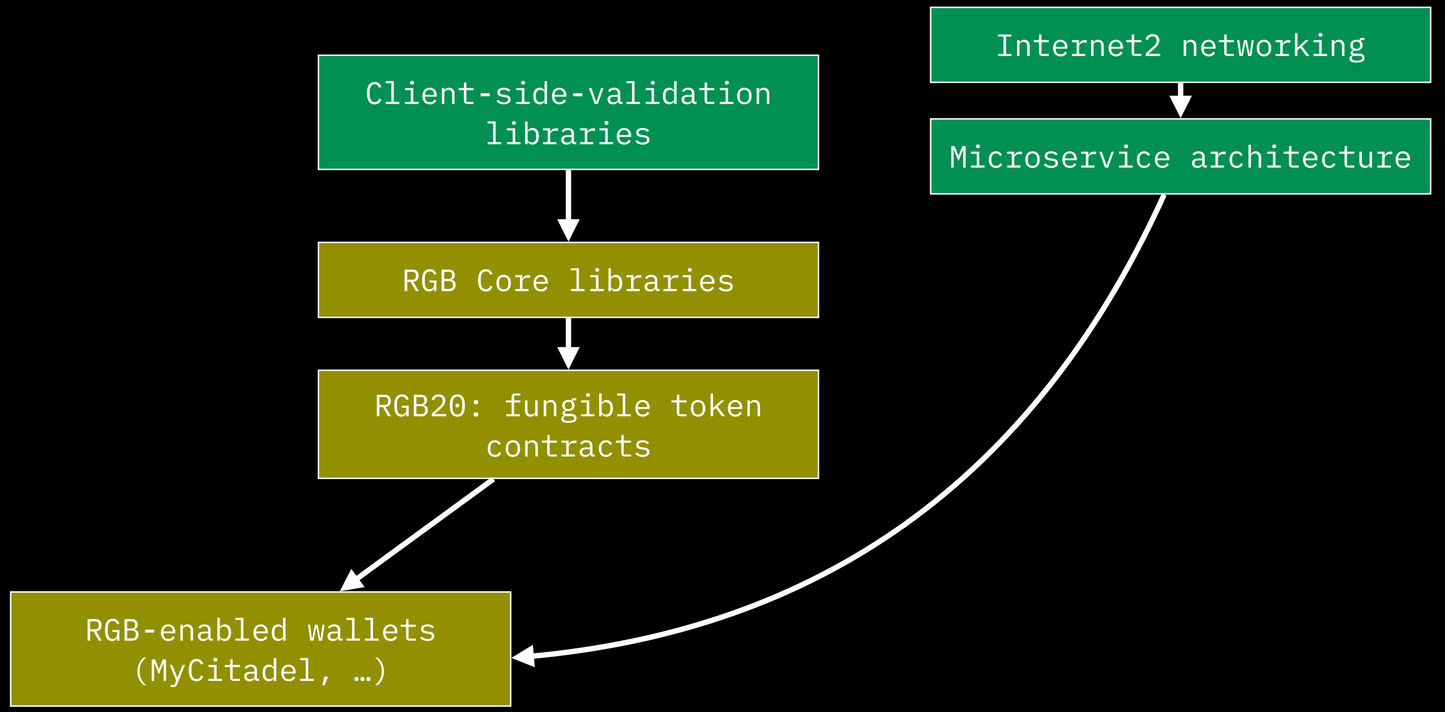
RGB public “reckless” release

100%	Fully completed
75%	Requires update
50%	Alpha ver / PoC
25%	Concept, needs impl
0%	Requires research

Client-side-validation libraries

- **Client-side-validation Foundation Lib**: released May 2021
  general (bitcoin-unspecific) concepts of
  - single-use-seals
  - commitments & validation
  - strict encoding
- **BP Core Lib**: pre-released these days, awaits Taproot
  - deterministic bitcoin commitments
  - bitcoin-specific single-use-seal implementation (UTXO-P2C)
- LNP/BP **Universal invoicing library**: pre-released March 2021
- **Libraries implementing minor LNPBP standards**: released today
  - Bech32 encodings (updated to Bech32m)
  - Universal chain parameters

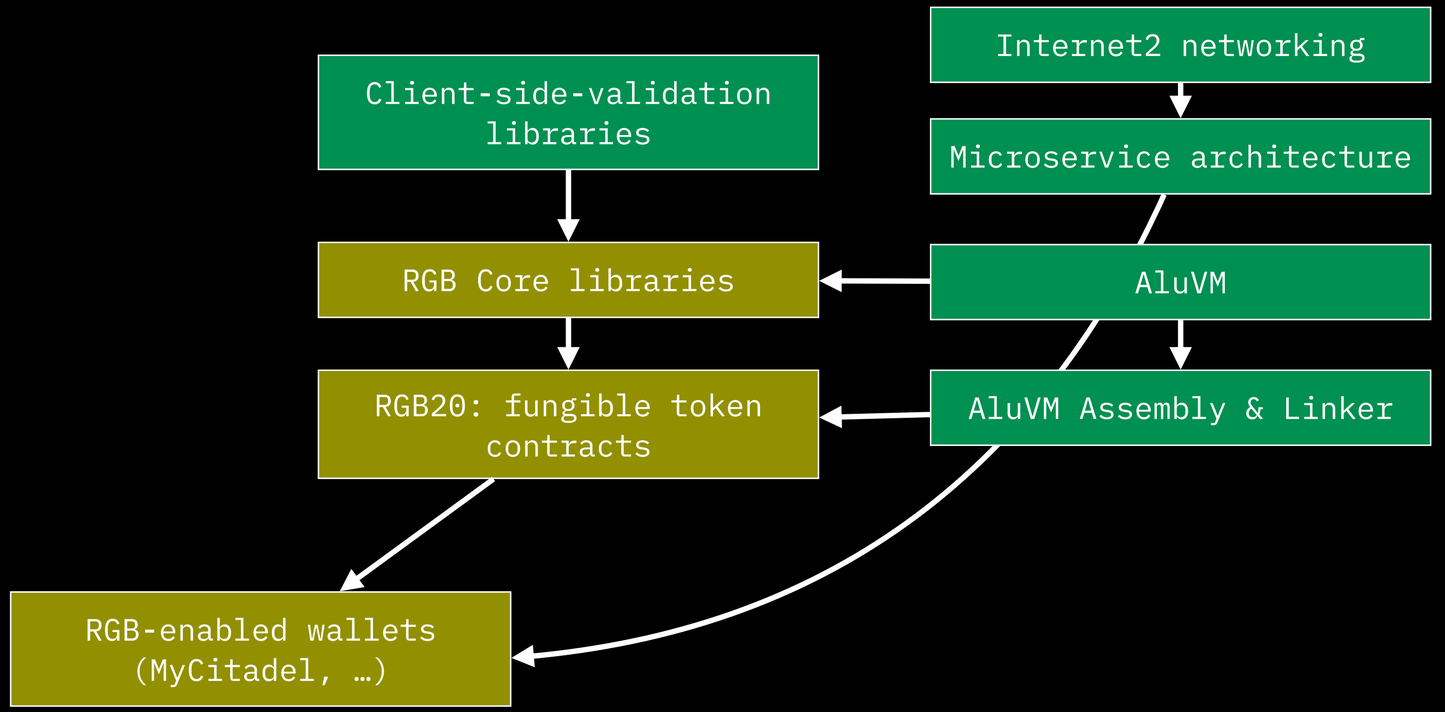
Lightning network (LNP): *Bitcoin protocol (BP):* *RGB smart contracts:* *Internet2:*



RGB public “reckless” release

100%	Fully completed
75%	Requires update
50%	Alpha ver / PoC
25%	Concept, needs impl
0%	Requires research

Lightning network (LNP): *Bitcoin protocol (BP):* *RGB smart contracts:* *Internet2:*



RGB public “reckless” release

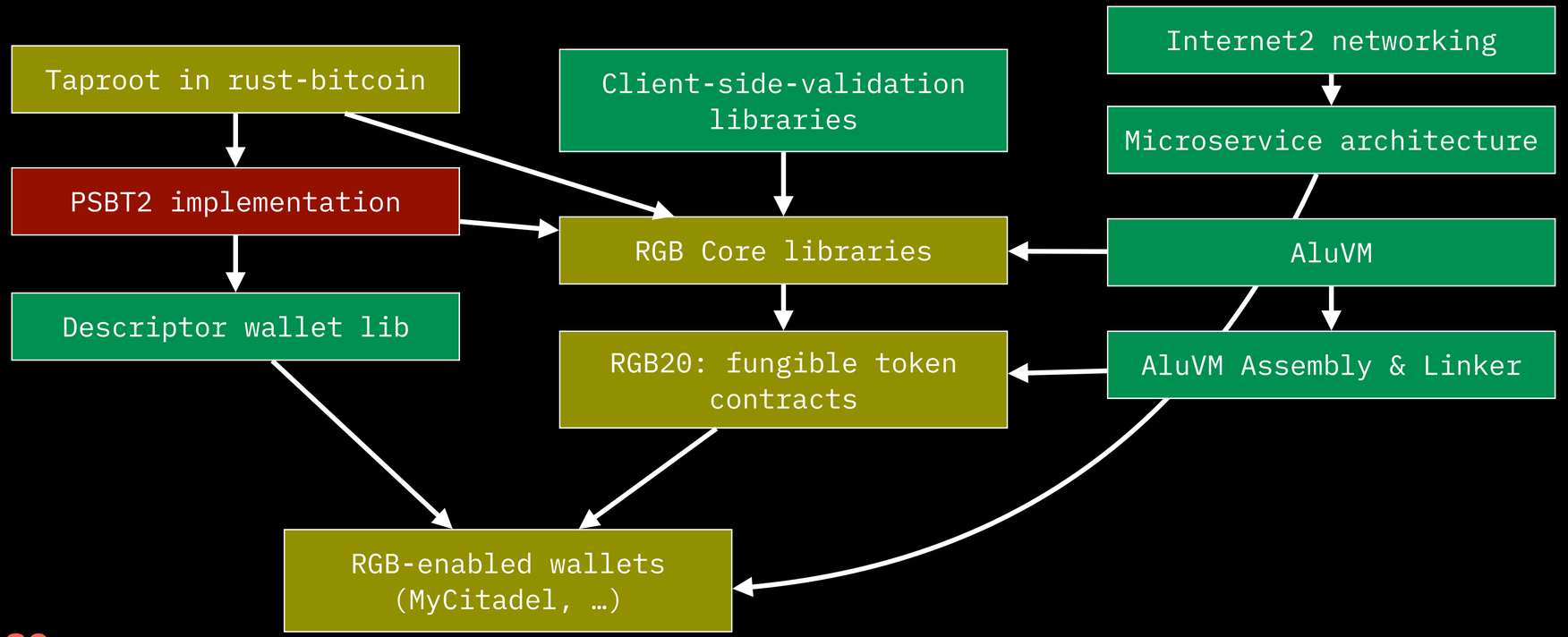
100%	Fully completed
75%	Requires update
50%	Alpha ver / PoC
25%	Concept, needs impl
0%	Requires research

RGB programmability

- Functional AluVM virtual machine, designed specifically for client-side-validation needs
- AluVM assembly language (AluAsm)
- Alu assembly toolchain: compiler & linker
- Alu library ecosystem & runtimes

```
.ISAE ; ISA Extensions segment
                ALU
                RGB

.LIBS
                pedersen alu145mc48u7f6n9lzesm5wpvrq5y8rck9ggyyjpd2vshpv3ww7cp89

.ROUTINE sum_inputs
                scn.i   0, a16[2]
loop:           pld.i   0, a16[2], r512[3], r512[4]
                call    sum
                dec     a16[2]
                ifz     a16[2]
                stinv
                jif     loop
                ret

.ROUTINE sum_outputs
                scn.o   0, a16[1]
loop:           pld.o   0, a16[1], r512[2], r512[3]
                call    sum
                dec     a16[1]
                ifz     a16[1]
                stinv
                jif     loop
                ret

.ROUTINE verify
                call    sum_inputs
                call    sum_outputs
                call    pedersen -> verify
                succ

```

Lightning network (LNP): *Bitcoin protocol (BP):* *RGB smart contracts:* *Internet2:*

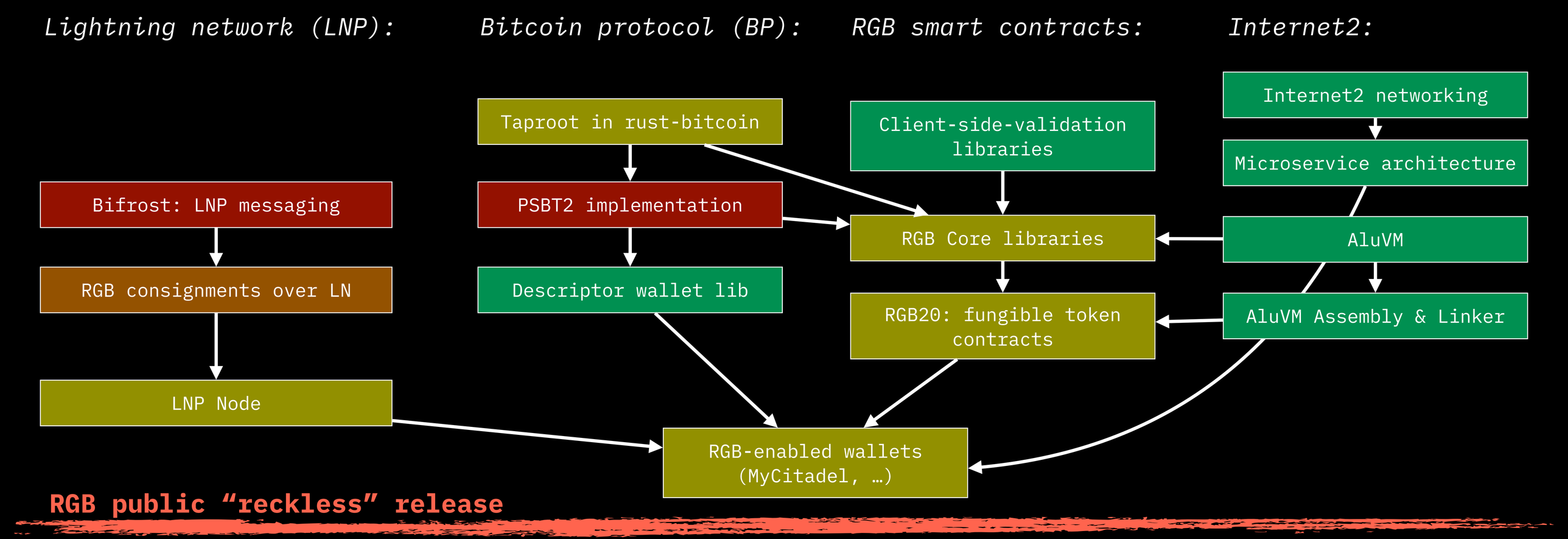


RGB public “reckless” release

100%	Fully completed
75%	Requires update
50%	Alpha ver / PoC
25%	Concept, needs impl
0%	Requires research

Descriptor wallet lib

- Support for key tweaking (RGB, taproot)
- Support for private-key-less BIP32 derivations
- Better PSBT & PSBT2
- Private key management fully separated from wallet function and interfaced only via PSBTs
- Going to be released once PSBT2 and Taproot support will be completed



100%	Fully completed
75%	Requires update
50%	Alpha ver / PoC
25%	Concept, needs impl
0%	Requires research

LN should upgrade for …

✓ Dually-funded channels

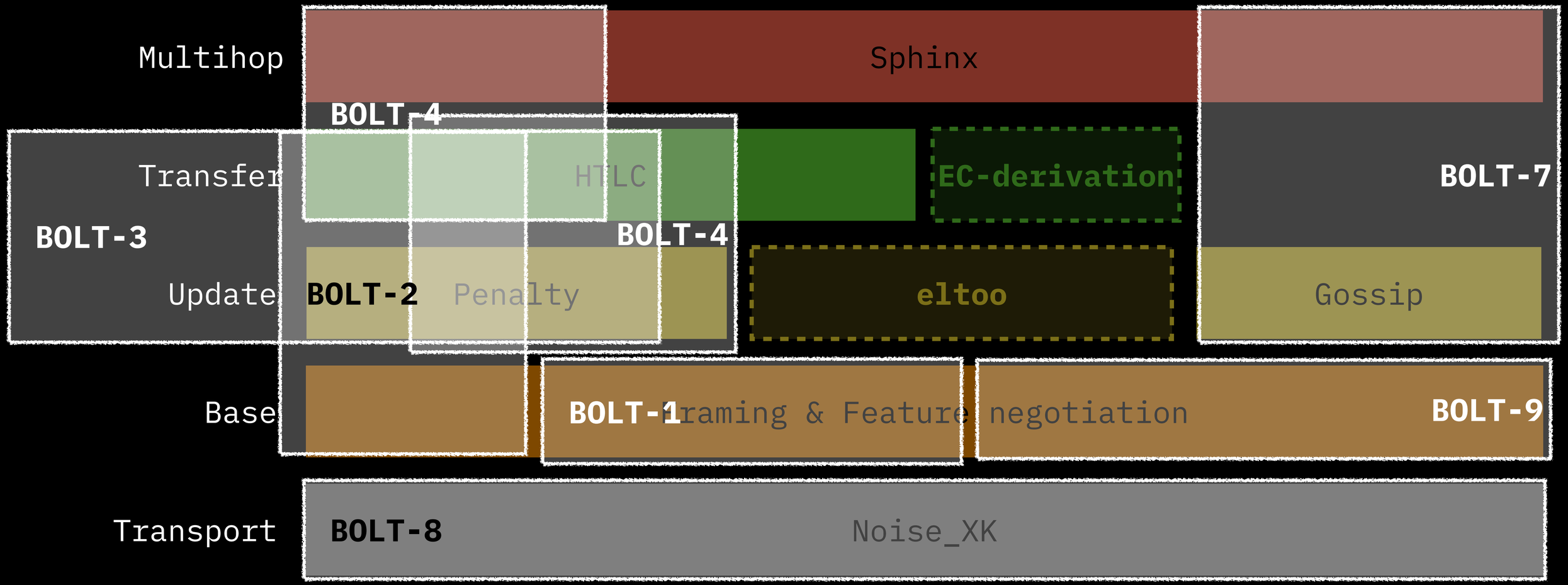
- Channel factories, multi-peer channels
- Channel splits
- Schnorr signatures
- Taproot
- Payment points (PTLCs)
- Eventually, eltoo

- RGB
- Discreet log contracts on LN
- Lightspeed: high-frequency micropayments
- Storm: storage and messaging; with multi-peer channels
- Prometheus: high-load computing; with multi-peer channels

What to do to extend LN?

- Do proper standards as LNPBPs

Current BOLT Specifications



What to do to extend LN?

√ Do proper standards as LNPBPs

- Reduce new protocol complexity to a few clear LN extensions: generalized Lighting Network
- Do reference implementations for the tech: LNP Core Lib
- Create extensible nodes & SDKs:
  LNP node, BP node, Wallet SDK

Step 1

Separate networking from payment protocol

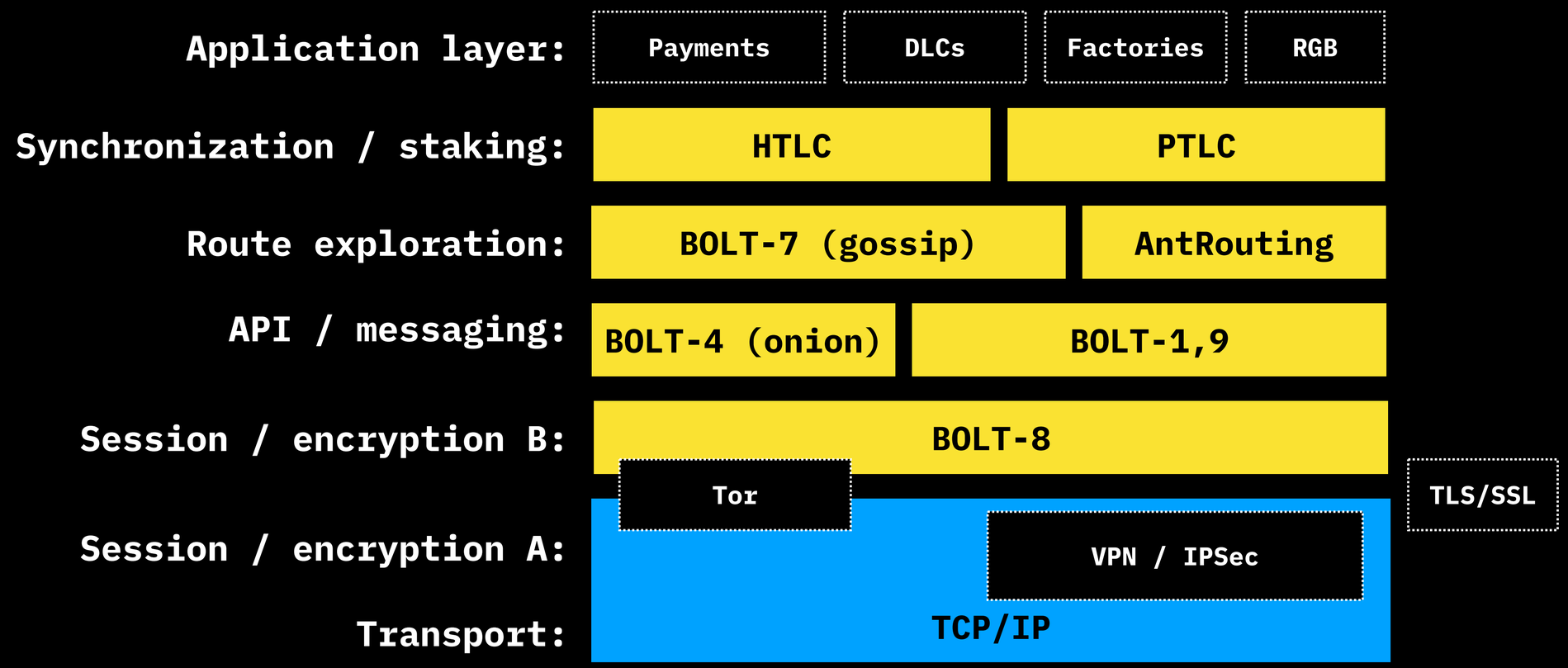
LN “networking”

- Session-level: **Noise_XK** protocol (BOLT-8)
- **P2P messaging** protocol
- Sphinx onion messaging

LN payments

- Network topology discovery
  - gossiping & minimum fee
  - ant-routing
- Specific data inside LN messages
- Structure of bitcoin transactions
- “Staking” protocol
  - penalty-based
  - eltoo
- Cryptographic atomicity
  - HTLC
  - PTLC

Networking part of LN



Step 2

Generalize channels: from payment channels to state channels

Extensible state channels P2P

- Allows negotiation of channel structure
  - Channel splits
  - Channel factories / multiplier channels
  - More options for liquidity from on-chain
  - DLCs and future protocols (Lightspeed, Storm, Prometheus)
- Allows tweaking of used keys
  - Taproot
  - RGB

Step 3

Unlock layer 3 applications

Extensible state channels P2P

- Abstract application protocols on top of networking:
  - Sending RGB data (consignments, asset info, NFT data containers)
  - DEX (Kaleidoscope)
  - Data storage (backups, watchtowers)
  - Running data queries
- Unlock more types of state channels:
  - Factories / multi-peer
  - DLCs
  - Lightspeed
  - Storm & Prometheus

Upgrading LN with Bifrost: Reckless^2

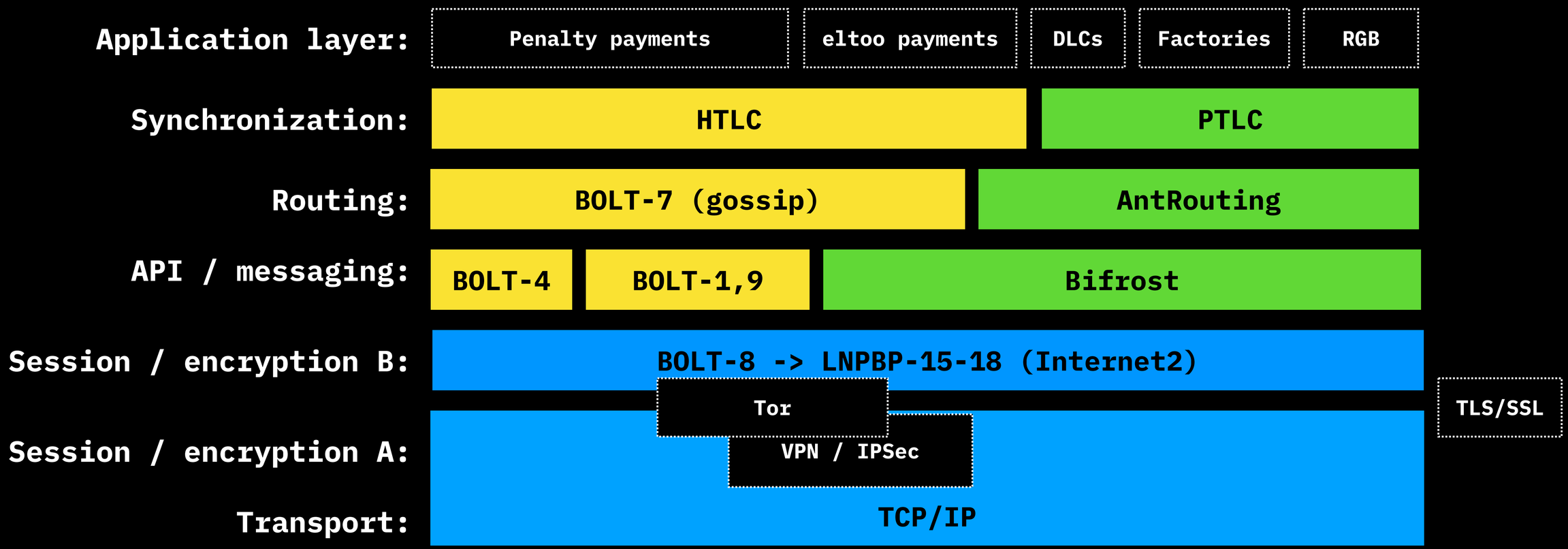
- We want to reduce the number of required changes to BOLTs to ZERO
- We need to separate risks of new experimental protocol from more stable LN
- We need clear layerization / abstractions

Doing Bifrost: Reckless^2

Bifrost will become a lightning “subnetwork”:

- Separate port number
- Modified BOLT-8/1 for session level (“Internet2”)
- A single custom message over “legacy” LN connection to upgrade to the new port/protocol (like from HTTP to WebSocket)

Networking protocols in LNP



realistic only with Bifrost

- BOLT-8 is extended to LNPBP-15-18 adding tunneling over HTTP, WebSockets and ZMQ
- LN P2P message protocols (BOLT-1, 4, 9) are replaced with **Bifrost**: extensible LN P2P messaging, which allows composable layered stack of specific LN-related protocols
  - network topology and payment routing (BOLT-7, AntRouting)
  - HTLC and PTLC protocols
  - multiple application-level protocols for
    - messaging (like in Sphinx)
    - RGB and DEX (Kaleidoscope)
    - channel factories
    - channel extensions (DLCs, Lightspeed)
    - non-classical state channels (Storm, Prometheus)

With Bifrost, anyone can propose an extension / new application for LN at any level (routing, payments, custom channels, data propagation)

– without the need to standardize it to experiment with it

LNP Node will be the only node supporting
both Bifrost and arbitrary
LN protocol extensibility

We hope others will work on Bifrost
support in the future

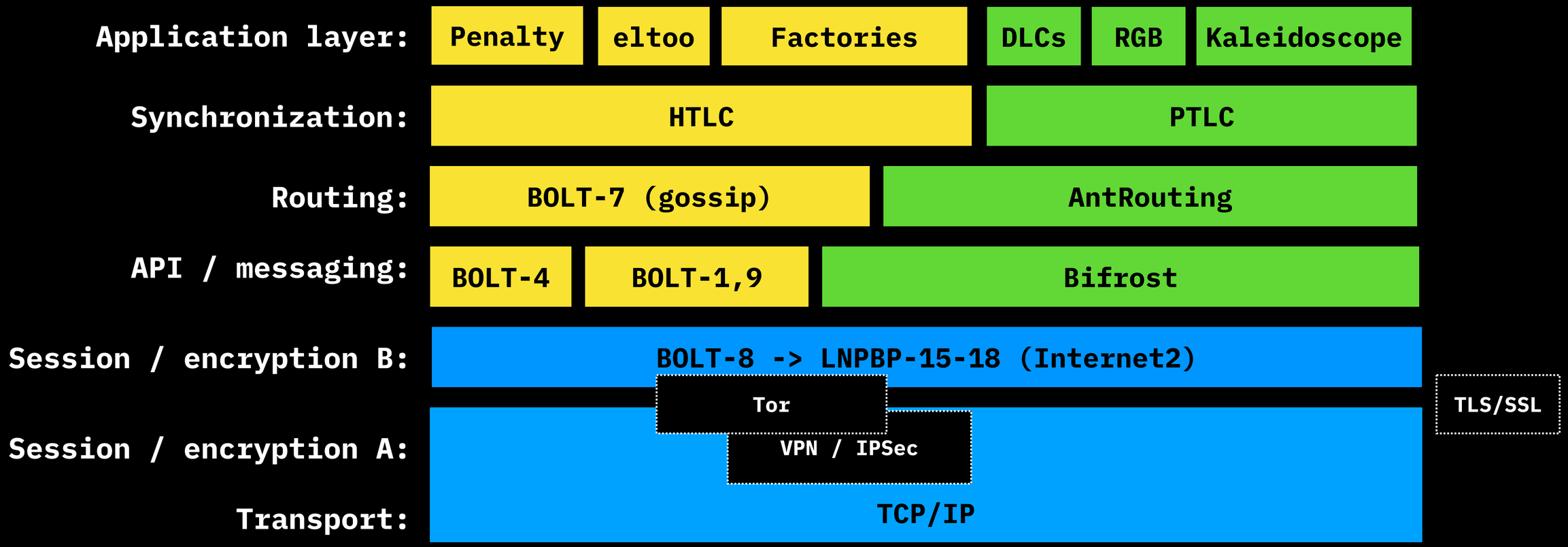
LNP Node extensions today are rust-written modules requiring re-compiling of the node.

In 2022 we plan to add support for hot-pluggable AluVM LNP extensions

Disambiguating terminology

- **LNP**: everything of lightning on top of session layer, including
  - Lightning network BOLTs (“legacy Lightning”)
  - Bifrost
  - Specific applications (DEX, RGB, …)
- **Bifrost**: new lightning P2P messaging protocol.
  Extensibility layer of LNP, allowing custom apps.
  Foundation for bridging worlds of RGB, DEX, legacy LN, …
- **Kaleidoscope**: specific application on top of Bifrost and RGB for decentralized exchange (DEX)

LNP stack



realistic only with Bifrost

Bifrost: how the nodes communicate & exchange commands + data

This allows such applications as:

- publishing info about RGB assets
- sending RGB consignments (even for on-chain payments)
- providing pre-paid storage for data backups, consignments, NFTs
- providing general data query network
  (naming, identity, route construction)

Bifrost: how the nodes communicate & exchange commands + data

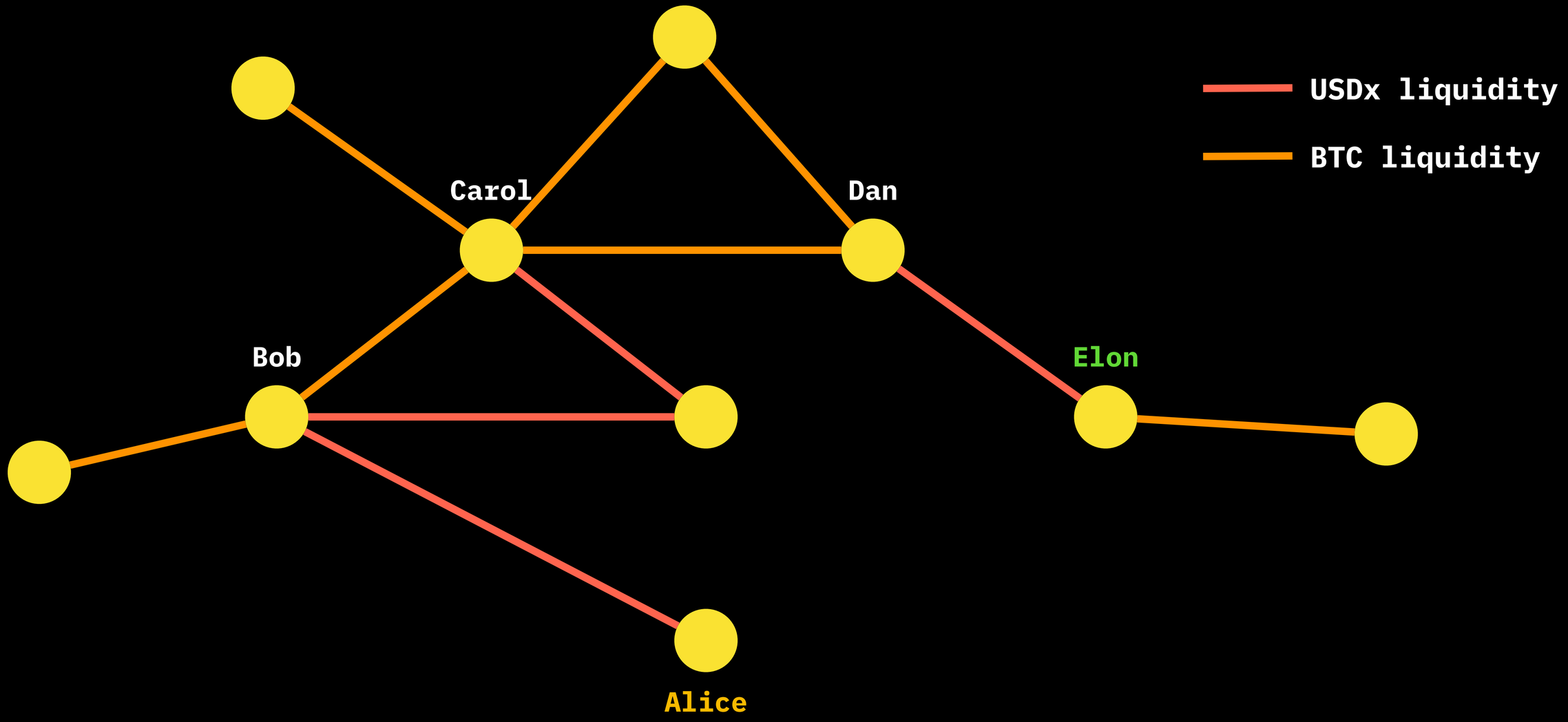
… this allows such applications as:

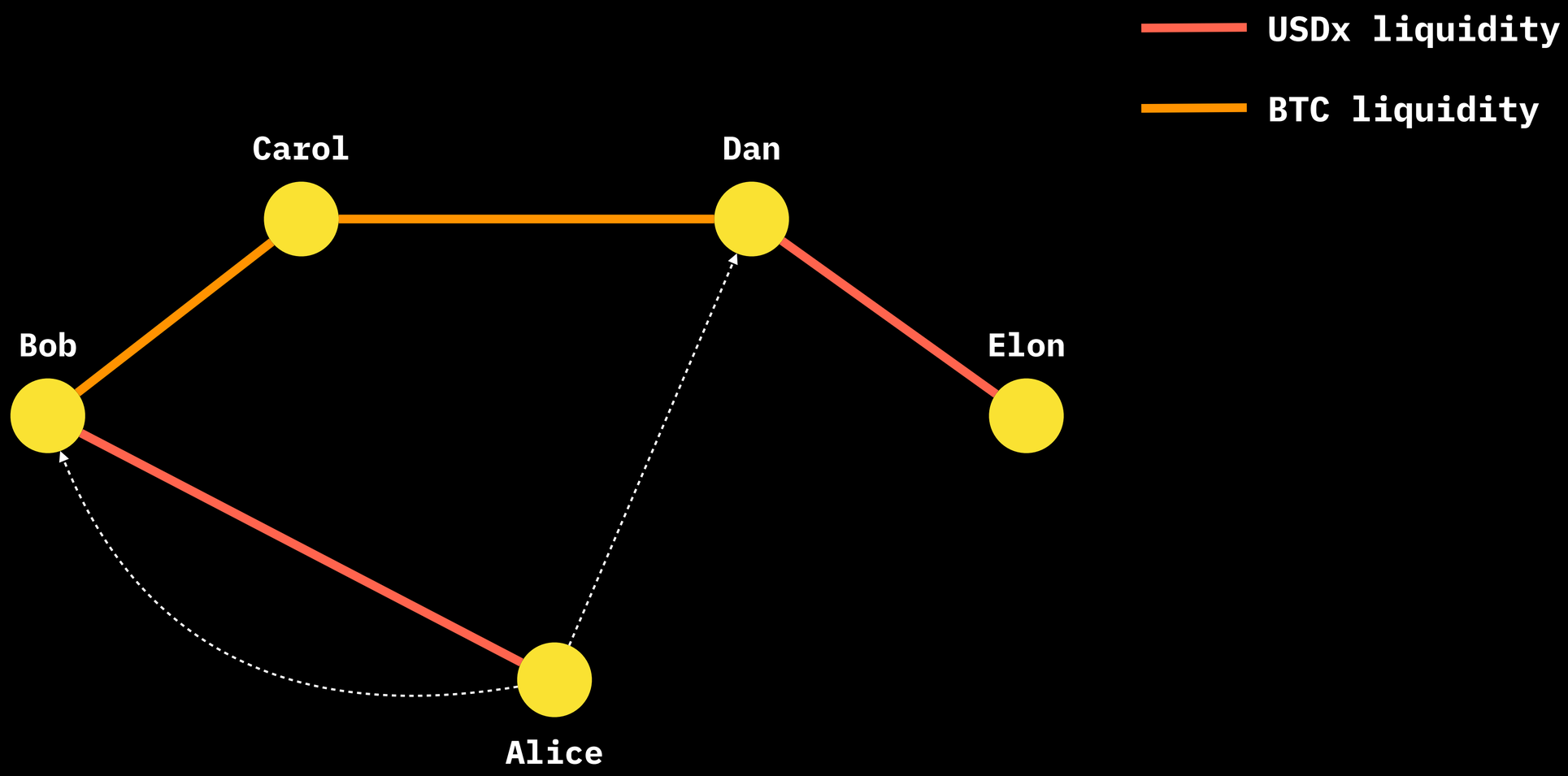
- Better (RGB-enabled) watchtowers
- More channel operations: splits, re-funding, dual-funding
- More channel types: channel factories, Storm
- More channel extensions: DLCs, Lightspeed

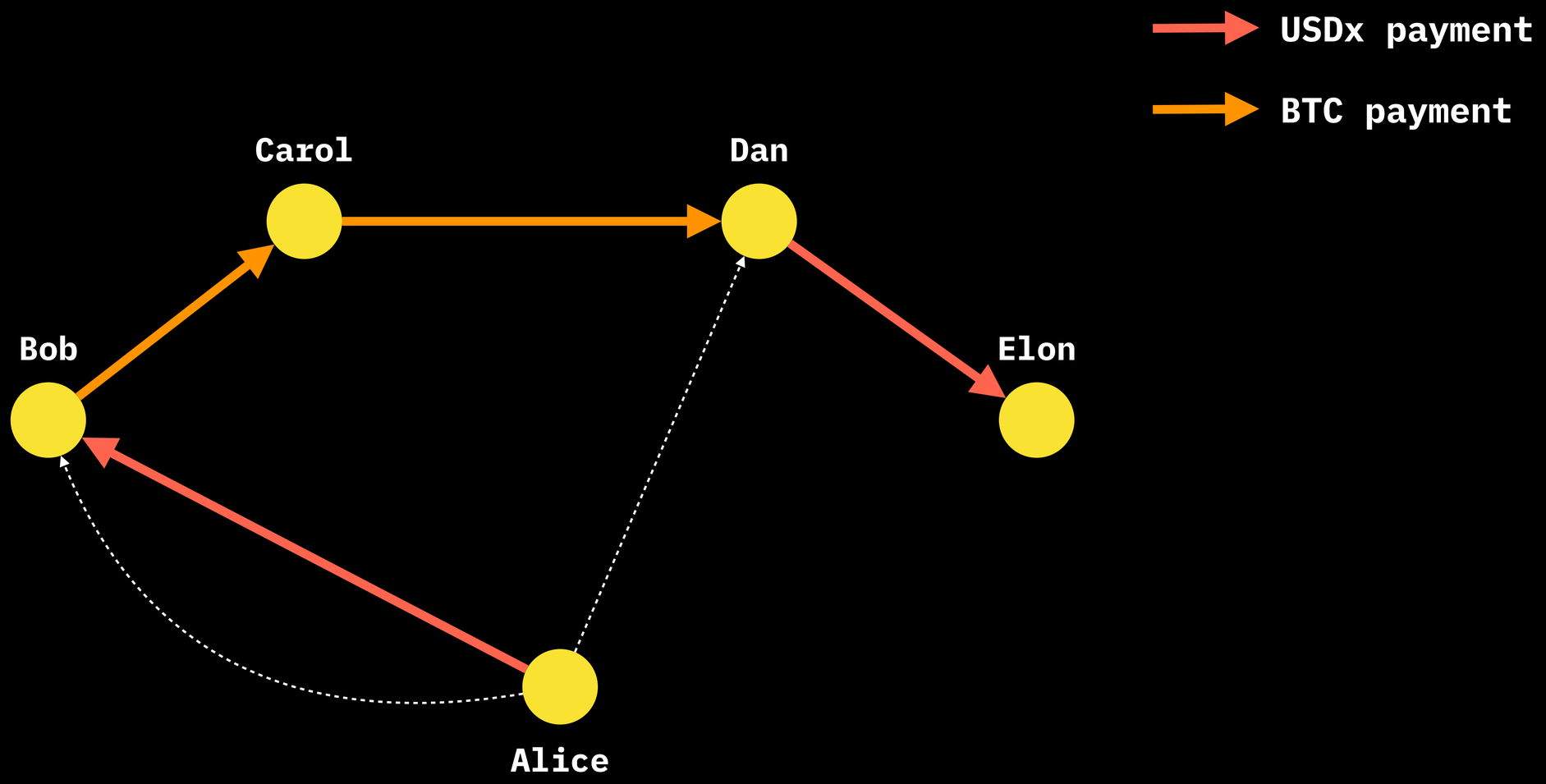
DEX (**Kaleidoscope**):

- LN network already (today) has information about liquidity, nothing required here
- During path construction one connects (via Bifrost) to the nodes which has channels in different assets and asks for them for exchange rate, fees and necessary liquidity
- HTLC/PTLC “coupling” of the exchange will prevent reverse American call option problem (still area of research)

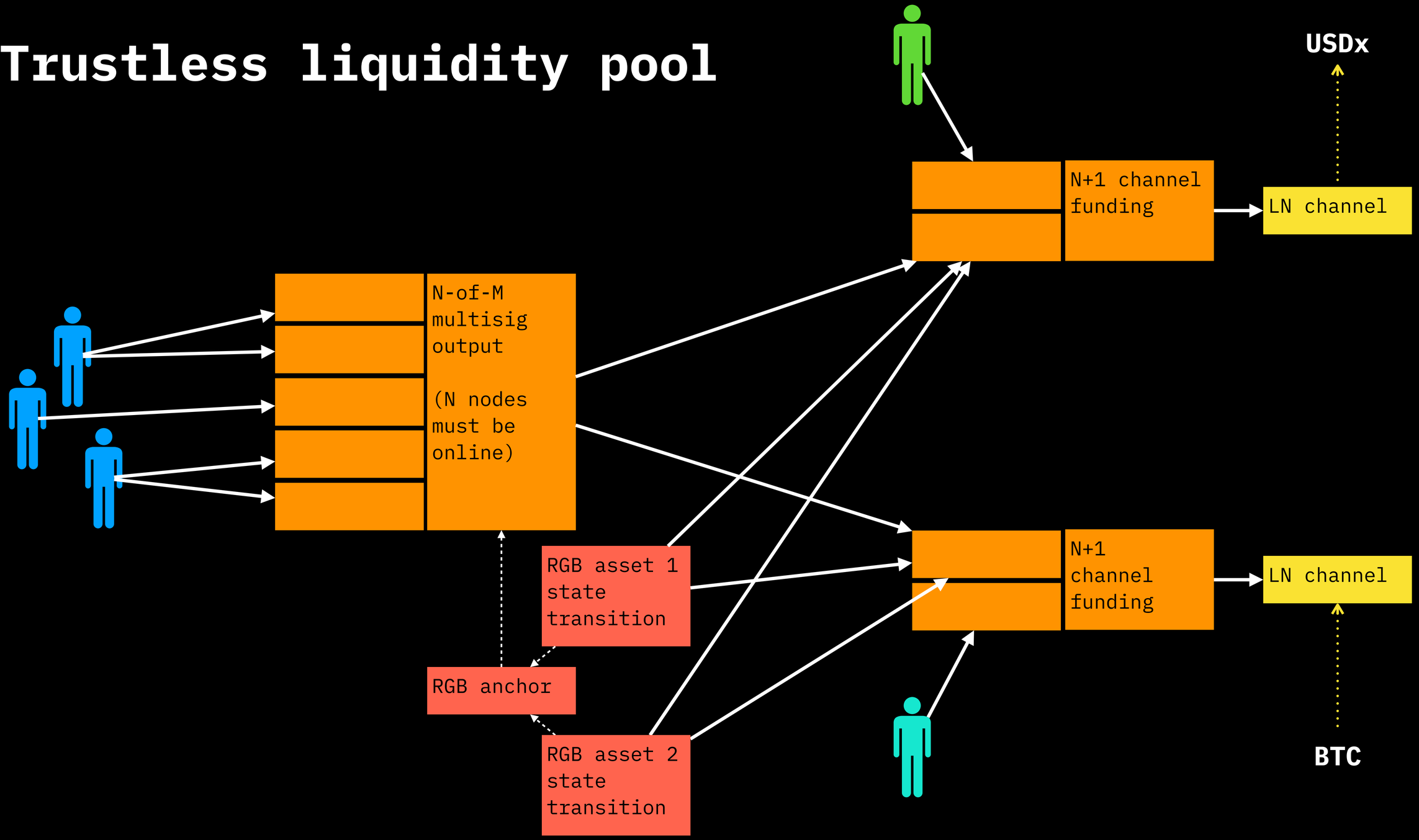
This requires neither order book or liquidity pool, it is a new type of DEX


USDx liquidity
BTC liquidity
Carol
Dan
Bob
Elon
Alice


USDx liquidity
BTC liquidity
Carol
Dan
Bob
Elon
Alice

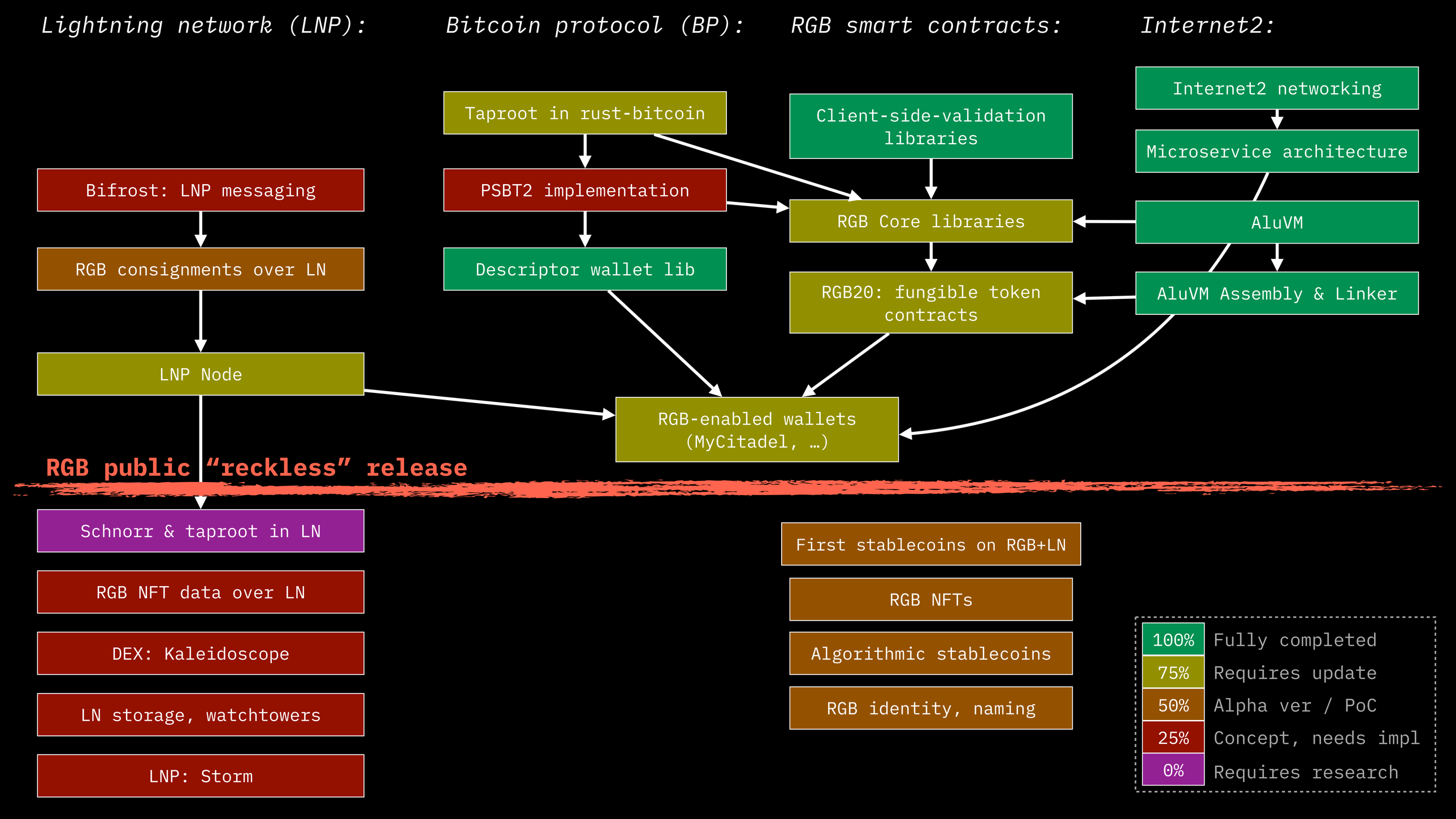

USDx payment
BTC payment
Carol
Dan
Bob
Elon
Alice

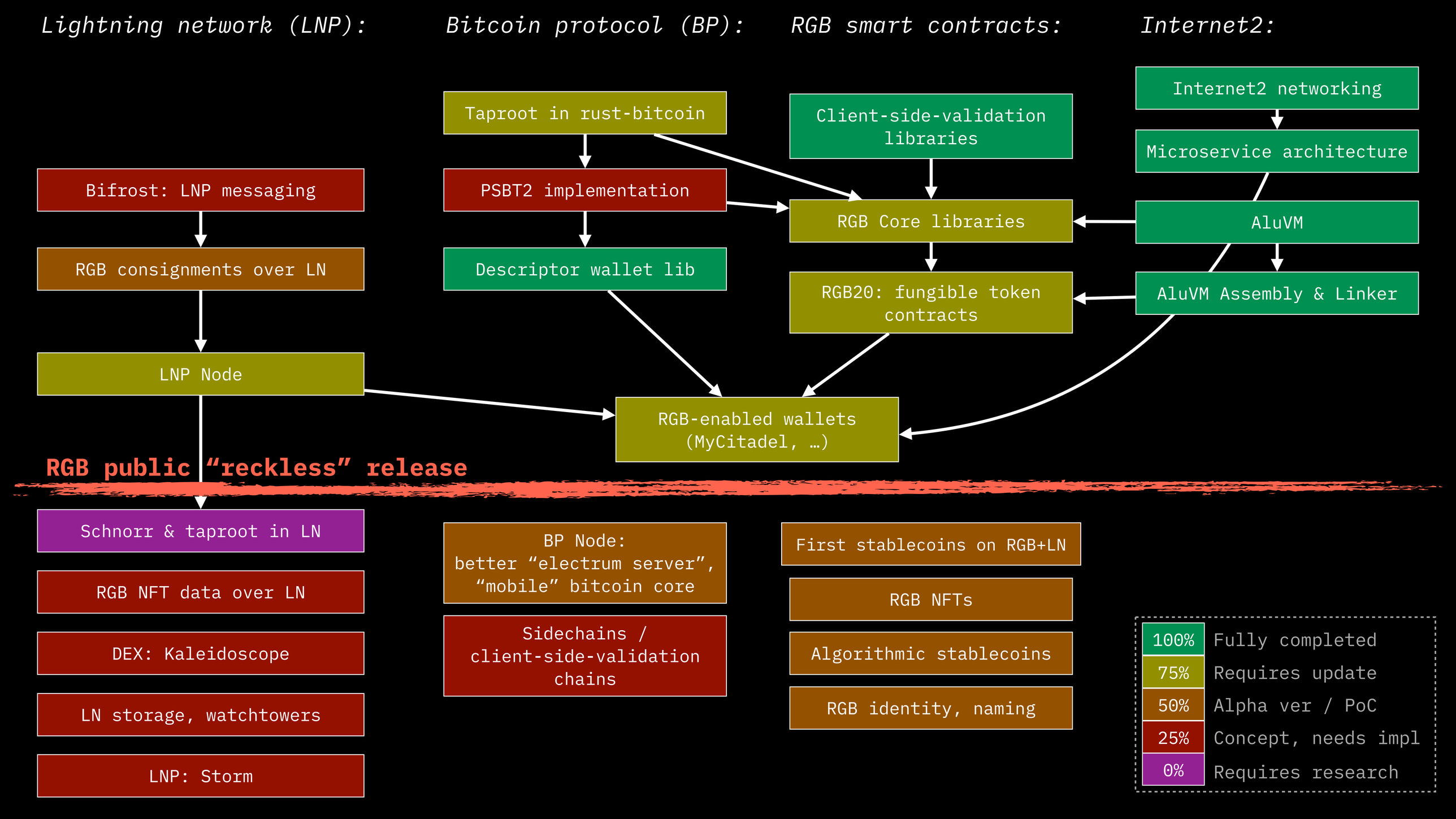
Trustless liquidity pool

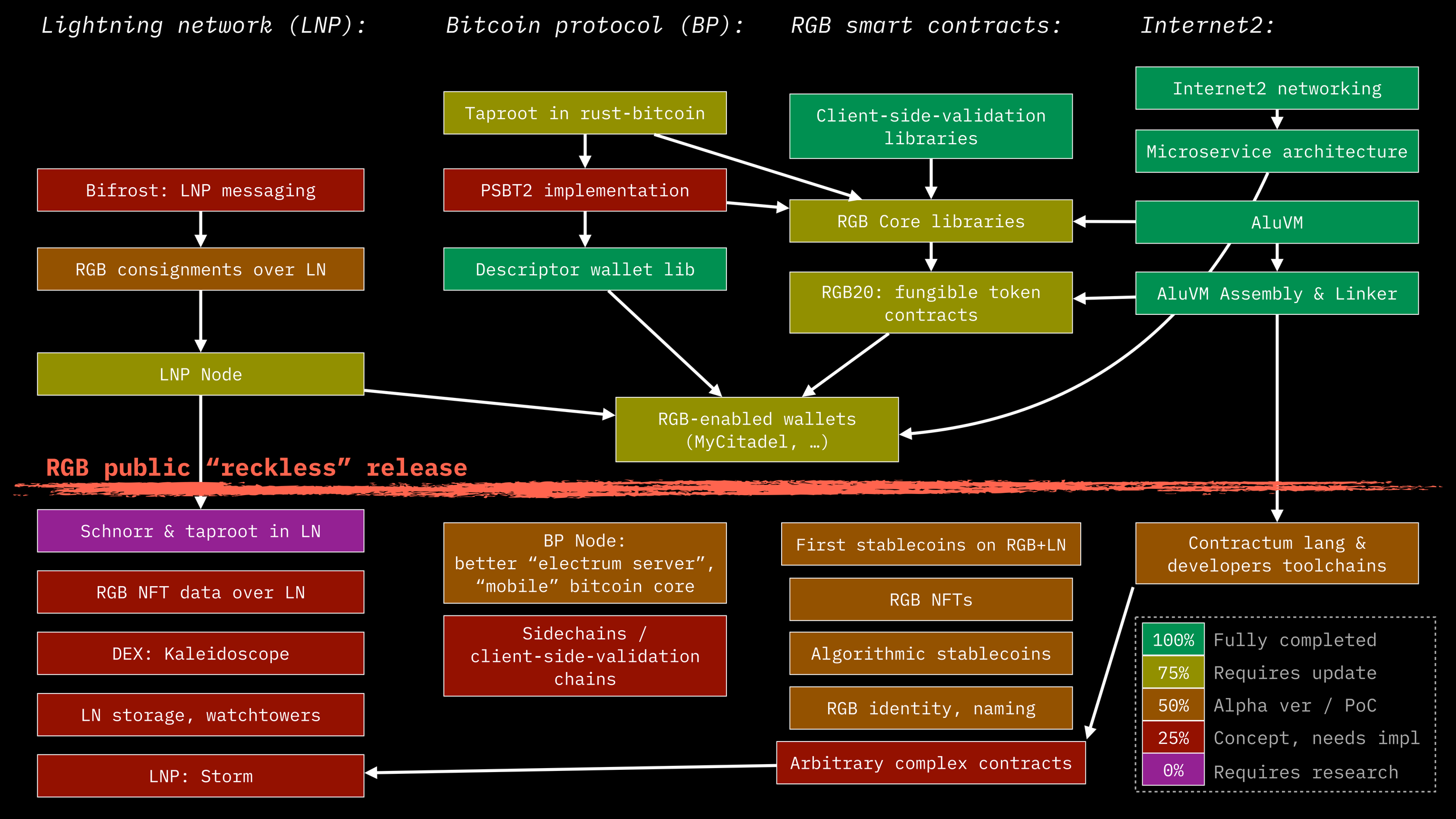


LNP roadmap

- **Stage 1**: together with RGB release (~Nov 2021)
  - Bifrost
  - Publishing information about RGB assets
  - Sending RGB consignments
- **Stage 2**: (2022 H1)
  - Schnorr, taproot & PTLC-based channels
  - More RGB apps: sending / storing NFTs, backups, watchtowers
- **Stage 3**: (2022 H2)
  - DEX (codename “Kaleidoscope”)
- **Stage 4**: (2023)
  - Trustless storage (Storm)


Lightning network (LNP):
Bitcoin protocol (BP):
RGB smart contracts:
Internet2:
Taproot in rust-bitcoin
Client-side-validation libraries
Internet2 networking
Microservice architecture
Bifrost: LNP messaging
PSBT2 implementation
RGB Core libraries
AluVM
RGB consignments over LN
Descriptor wallet lib
RGB20: fungible token contracts
AluVM Assembly & Linker
LNP Node
RGB-enabled wallets (MyCitadel, …)
RGB public “reckless” release
Schnorr & taproot in LN
RGB NFT data over LN
DEX: Kaleidoscope
LN storage, watchtowers
LNP: Storm
First stablecoins on RGB+LN
RGB NFTs
Algorithmic stablecoins
RGB identity, naming
100% Fully completed
75% Requires update
50% Alpha ver / PoC
25% Concept, needs impl
0% Requires research

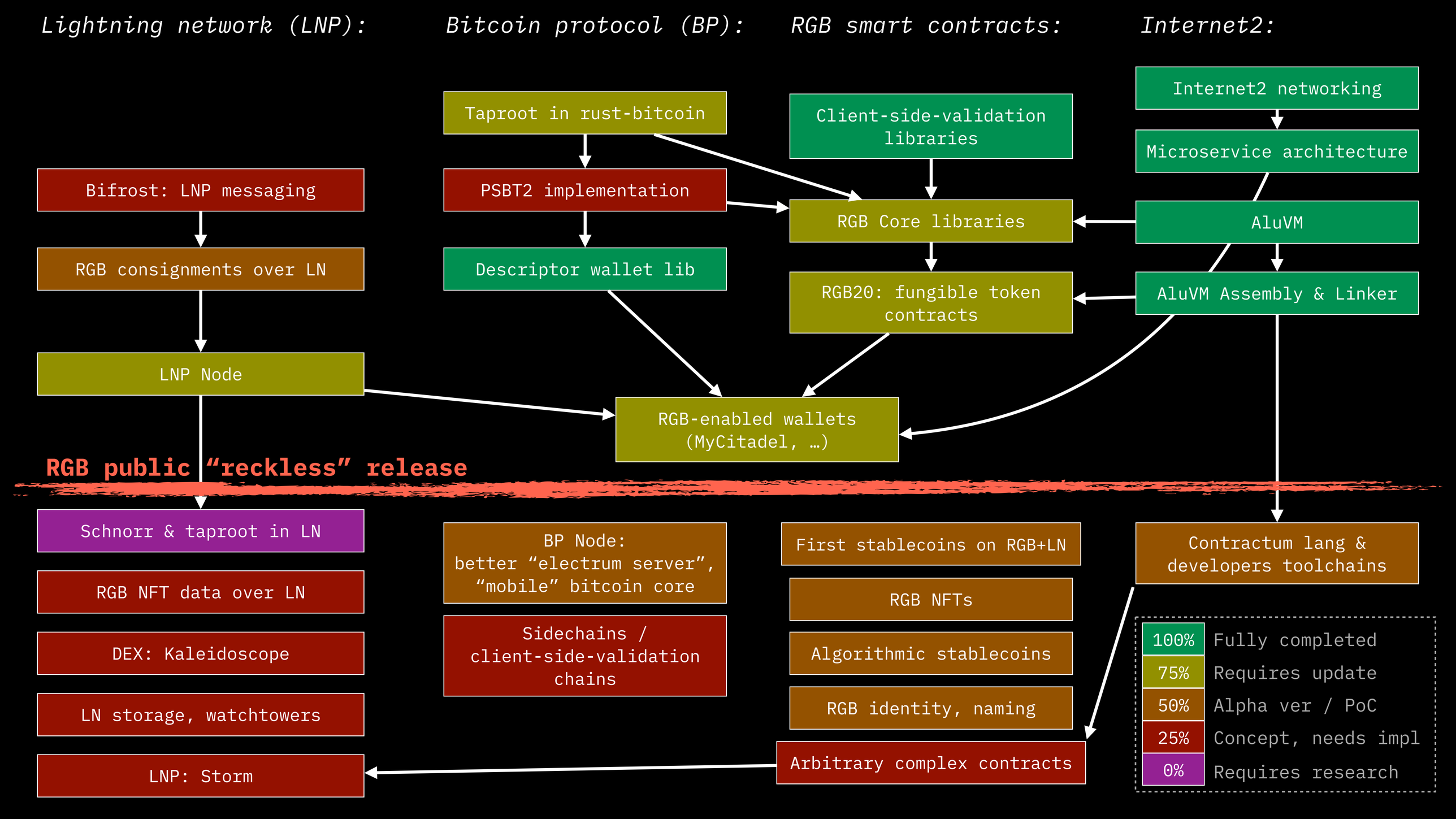

Lightning network (LNP):
Bitcoin protocol (BP):
RGB smart contracts:
Internet2:
Taproot in rust-bitcoin
Client-side-validation libraries
Internet2 networking
Microservice architecture
Bifrost: LNP messaging
PSBT2 implementation
RGB Core libraries
AluVM
RGB consignments over LN
Descriptor wallet lib
RGB20: fungible token contracts
AluVM Assembly & Linker
LNP Node
RGB-enabled wallets (MyCitadel, …)
RGB public “reckless” release
Schnorr & taproot in LN
BP Node: better “electrum server”, “mobile” bitcoin core
First stablecoins on RGB+LN
RGB NFT data over LN
RGB NFTs
DEX: Kaleidoscope
Sidechains / client-side-validation chains
Algorithmic stablecoins
LN storage, watchtowers
RGB identity, naming
LNP: Storm
100% Fully completed
75% Requires update
50% Alpha ver / PoC
25% Concept, needs impl
0% Requires research


Lightning network (LNP):
Bitcoin protocol (BP):
RGB smart contracts:
Internet2:
Taproot in rust-bitcoin
Client-side-validation libraries
Internet2 networking
Microservice architecture
Bifrost: LNP messaging
PSBT2 implementation
RGB Core libraries
AluVM
RGB consignments over LN
Descriptor wallet lib
RGB20: fungible token contracts
AluVM Assembly & Linker
LNP Node
RGB-enabled wallets (MyCitadel, …)
RGB public “reckless” release
Schnorr & taproot in LN
BP Node: better “electrum server”, “mobile” bitcoin core
First stablecoins on RGB+LN
Contractum lang & developers toolchains
RGB NFT data over LN
RGB NFTs
DEX: Kaleidoscope
Sidechains / client-side-validation chains
Algorithmic stablecoins
LN storage, watchtowers
RGB identity, naming
LNP: Storm
Arbitrary complex contracts
100% Fully completed
75% Requires update
50% Alpha ver / PoC
25% Concept, needs impl
0% Requires research

Introducing Contractum

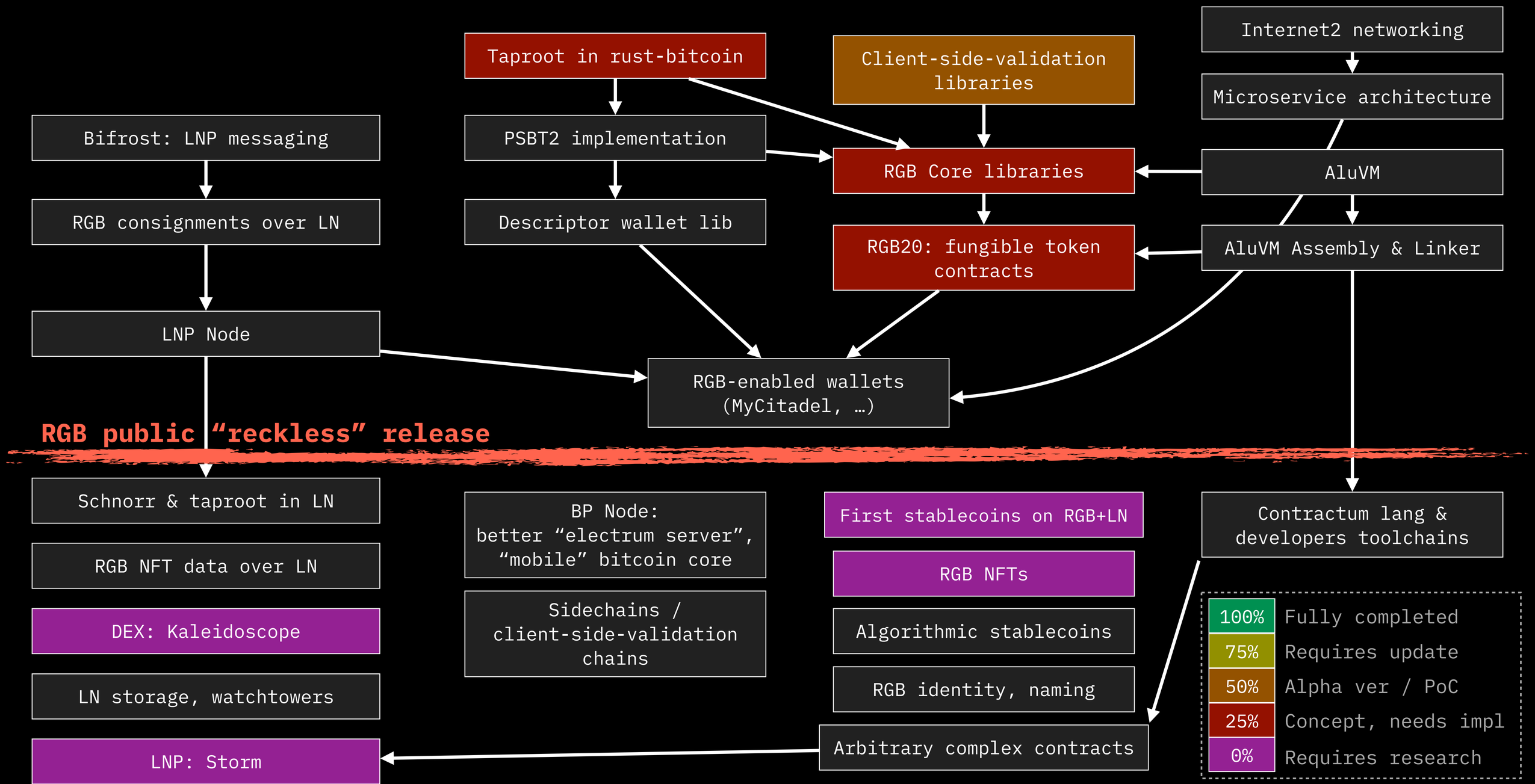
- Haskell-inspired language for RGB smart contracts
  (but not subset of Haskell itself)
- Avoids visual clutter
- Focus on validation procedures
- Avoids foot guns as much as possible
- Designed with composability & category theory in mind,
  made for formal verification

```
schema RGB20Simple using Assets
    field Ticker :: UTF8^3..8
    field Name :: UTF8^8..32
    field ContractText :: UTF8*
    field DecFractions :: U8 <- precision U8
        -- here we must validate the value range,
        -- so we are taking U8 value with `<-`
        -- and making sure it fits into our requirements
        assert precision in 0..=18
    field IssuedAmount :: AssetAmount

    -- if this was NFT, we can add something like
    -- container Painting :: image/png

    homomorph AssignedAmount AssetAmount
    right Renomination
    -- hashed accumulating Engraving -- used in NFTs

    genesis :: Ticker, Name, ContractText?, IssuedAmount, allocation AssignedAmount*, Renomination?
        assert sum! *allocation == issuedAmount

    transition transfer :: spent AssignedAmount+ -> sent AssignedAmount+
        assert sum! #spent == sum #sent

    transition renominate :: Renomination -> Renomination?, Ticker?, Name?, ContractText?
        assert ticker? or name? or contractText?

    method totalIssued :: () -> AssetAmount
        return @self.issuedAmount
```

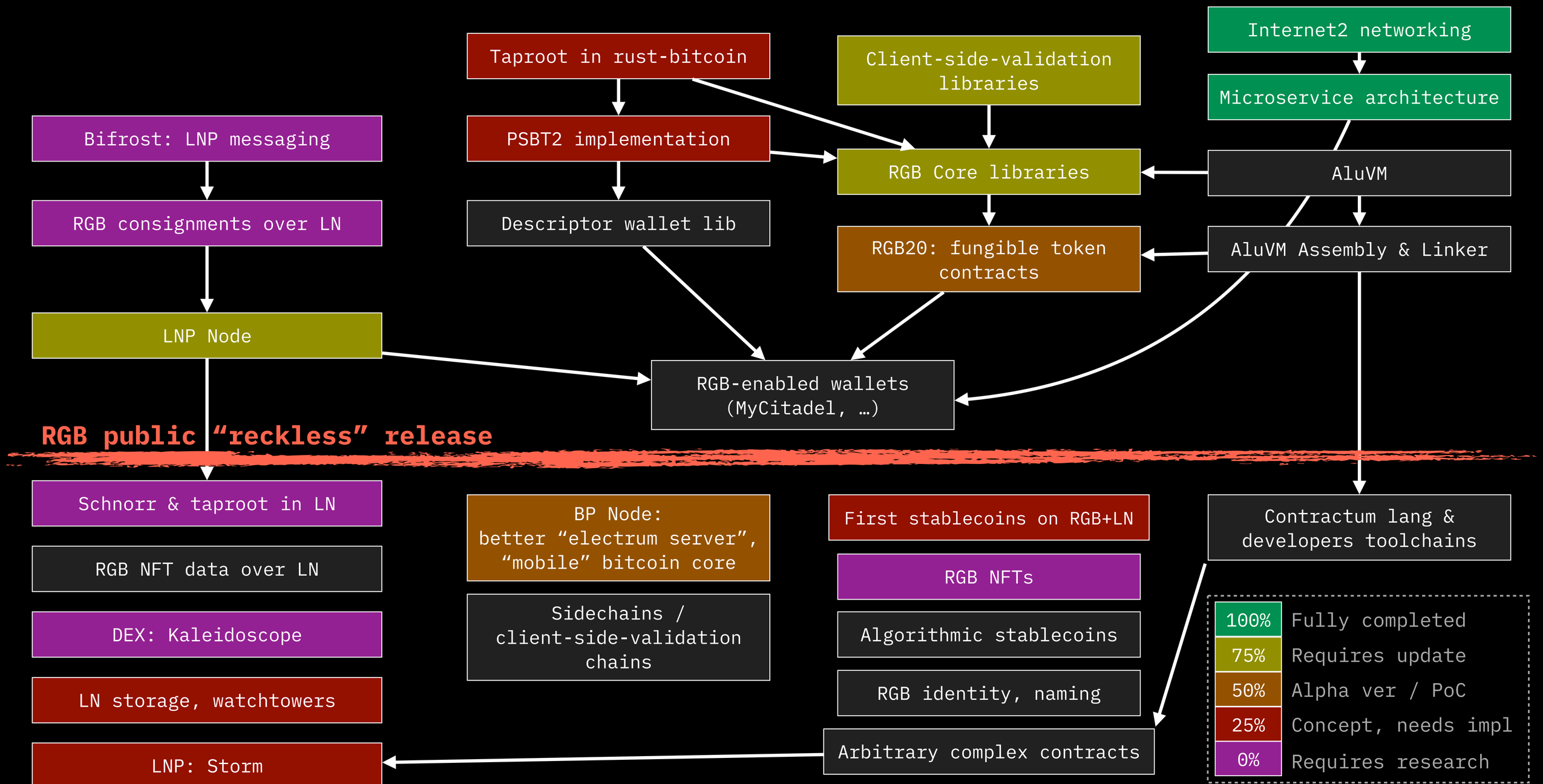

Lightning network (LNP):
Bitcoin protocol (BP):
RGB smart contracts:
Internet2:
Taproot in rust-bitcoin
Client-side-validation libraries
Internet2 networking
Microservice architecture
Bifrost: LNP messaging
PSBT2 implementation
RGB Core libraries
AluVM
RGB consignments over LN
Descriptor wallet lib
RGB20: fungible token contracts
AluVM Assembly & Linker
LNP Node
RGB-enabled wallets (MyCitadel, …)
RGB public “reckless” release
Schnorr & taproot in LN
BP Node: better “electrum server”, “mobile” bitcoin core
First stablecoins on RGB+LN
Contractum lang & developers toolchains
RGB NFT data over LN
RGB NFTs
DEX: Kaleidoscope
Sidechains / client-side-validation chains
Algorithmic stablecoins
LN storage, watchtowers
RGB identity, naming
LNP: Storm
Arbitrary complex contracts
100% Fully completed
75% Requires update
50% Alpha ver / PoC
25% Concept, needs impl
0% Requires research

How far we are from the release?

State of development on 1 Jan 2020:



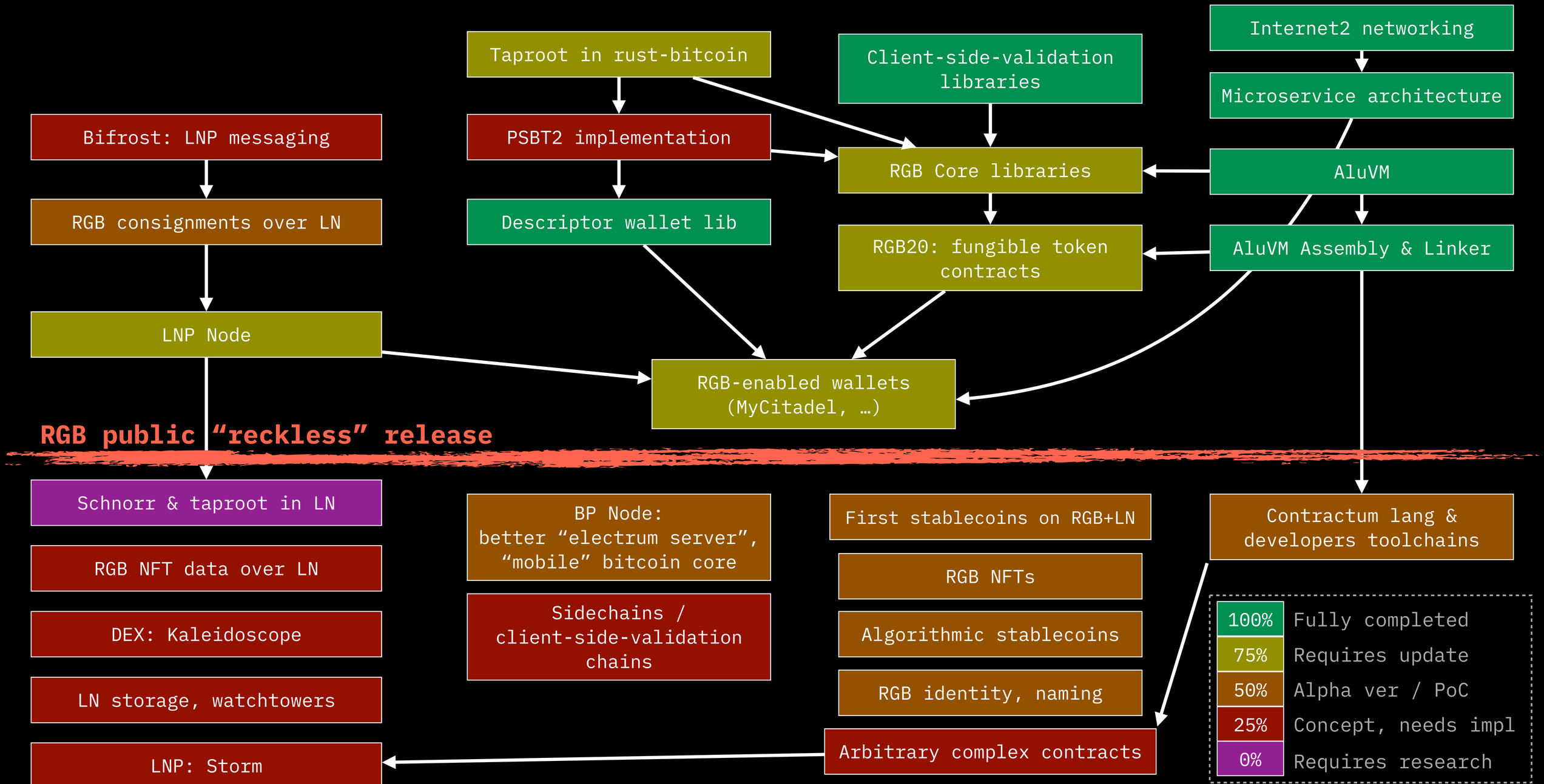
State of development on 1 Jan 2021:



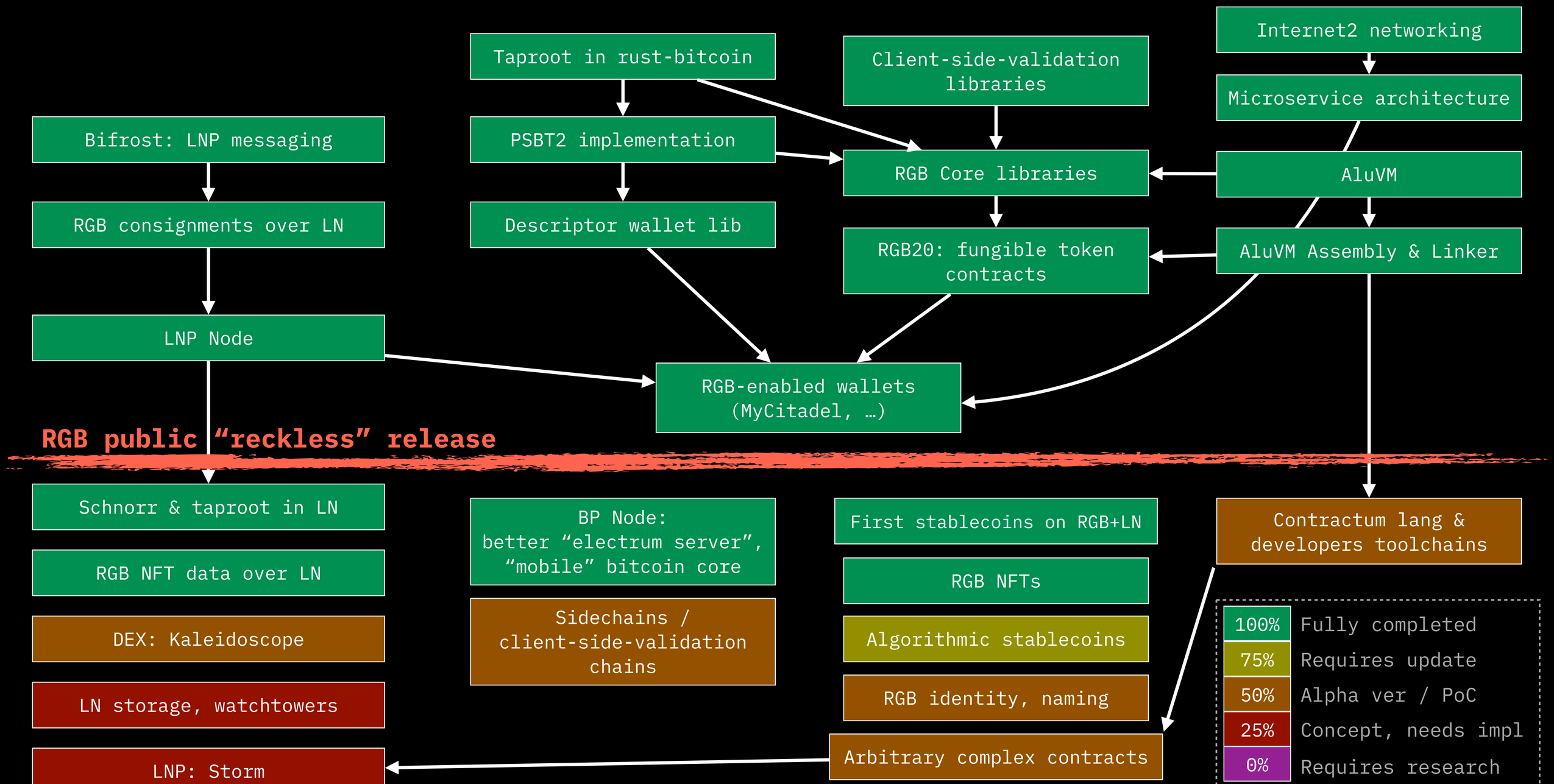
RGB: what was done in 2021 H1

- **Descriptor wallet** lib:
  - Support for key tweaking (RGB, taproot)
  - Support for private-key-less BIP32 derivations
- Runtime for RGB-enabled wallets and **first RGB PoC wallet** (MyCitadel) for mobile, server & desktop
- **Taproot & Schnorr** support in rust-bitcoin & secp256k1 (WIP)
- Release of **client-side-validation foundation** libs, standards etc
- **AluVM**: virtual machine, instruction set, assembly & linker

Status quo on 1st Sept 2021:



Expected by the end of 2021:



Today RGB & underlying foundations codebase
(client-side-validation, BP, LNP, Taproot/Schnorr)
consists of **~1m lines of rust code**,
created over last **2 years** (with multiple rewrites)

For **RGB going live** I expect another **~150k lines**
need to be written, which is doable in **~2-3 months**

Main remaining tasks & challenges

- **Taproot & Schnorr** finalization in rust-bitcoin & secp256k1:
  - *more reviewers needed*
- **PSBT2** implementation
  - *looking for co-authors*
- **Taproot-based LN** design
  - *looking for co-authors*
- Completion of **LNP Node**
  - *looking for contributors*

Peter Todd expressed interested in being involved into RGB completion & audit on half-time basis during the next year

We look for:

- Strong knowledge of rust
- Excellent knowledge of Bitcoin protocols, including Taproot / PSBT
- (optionally) good understanding of LN internals
- At least basic understanding of client-side-validation & RGB concepts, which can be taken from watching a couple of our videos

GitHub

Code & standards:

- github.com/LNP-BP – standards, client-side-validation, BP, LNP
- github.com/RGB-org – RGB smart contracts, Contractum language
- github.com/Internet2-org – AluVM, BOLT-8 based networking

Specifications (will be implemented later):

- github.com/Storm-org – decentralized storage & messaging
- github.com/Prometheus-org – decentralized trustless computing

LNP/BP Standards Association

- Swiss Association requiring no legal registration
- Founded in Oct 2019 by Maxim Orlovsky (Pandora Core AG) & Giacomo Zucco
- Oversees Layer 2 & 3-related standards and software development for Bitcoin / Lightning Network ecosystem
- Contributions/ideas from most prominent cryptographers (Peter Todd, Adam Back, Andrew Poelstra) & Lightning Network developers (Christian Decker, Rene Pickhardt)
- lnp-bp.org