Bitcoin (Blink) - Peer to Peer Global Cash System

[WORKING-DRAFT-v0.1]

Joby Reuben
joby@projectblink.org

Purva Chaudhari
purva@projectblink.org

Abstract : Bitcoin's PoW is replaced with a propagation competition on blocks sent across validators under a certain time interval stamped with cryptographic proofs to claim fees and solve forks as per proof weight. To achieve adaptable scalability, block sizes are determined by consensus among elected nodes in specific epochs to reduce transaction waiting time. Gossip systems are replaced with a privacy-centered direct messaging system by constructing encrypted paths to deliver unconfirmed transactions and confirmed blocks. Apart from bringing speed, we resolved the need for a single transaction fee token for a blockchain by bringing forth a novel non-custodial per-token staking system to allow users to pay in any token. Bitcoin as a currency will ensure network security and hold Layer-1 tokens with staking and yielding fees. Since Bitcoin script adapts a Turing-incomplete language, the fees imposed for renting UTXOs make the transactions cheaper and the chain's ledger size optimized. We propose solutions for regulation revolving around taxation within the self-custody wallet ecosystem without compromising users' privacy.

1 Introduction

The Bitcoin Network [1] and other altcoin blockchains with newer consensus and programmable money are unable to compete with centralized payment providers in speed and volume due to their sheer inability to scale with centralization issues. Many consensus models rely on external validation concepts such as finding a nonce and proving stake instead of rules tied to propagation itself. Imposing heavy reliance on users to acquire native chain tokens diminishes the adoption of blockchain, thus keeping them far from the wonders of this technology. Decentralized networks can effectively adapt to users' needs by 1. Increasing block size 2. Decreasing block time 3. Eliminating low-efficient nodes 4. Increasing the requirement for node joining. Retail Staking with non-custodial solutions encourages users to stake their Bitcoin to become a world reserve currency for every financial instrument with an additional restrictive monetary policy that helps to reduce volatility in times of recession.

Instead of storing UTXOs for an indefinite period, which compromises storage, renting UTXOs and replacing them with a fingerprint after they have expired without altering the block's Merkle root provides cheaper fees. With Bitcoin's unlocking script and use of sCrypt [2], developers can create custom scripts with regulatory options involving various types of taxes within its UTXOs, performing identity verification off-chain with signatures instructing nodes to validate regulated payments with self-custody of tokens. Altcoins can be bridged one-way and collateralized for a floating rate stable coin directly used for staking and yielding fees along with bitcoin bringing utility. Basic banking solutions can be developed in Bitcoin Script, whereas common computable programs can be deployed to a Layer 2 State Machine, where nodes update the state by providing a Proof-of-Fee-Receipt paid in any token in the Bitcoin network.

2 Bitcoin (Blink)

2.1 Election

Block size denotes the amount of data that can be propagated across every producer node on the Bitcoin network hence, its success rate is directly dependent on the bandwidth each node allocates for confirmed block transmission. Block size is not limited and is fixed in every new epoch, which validates that every producer node on the network can send and receive the data size. Variable block size helps to scale the network by increasing transactions per block when the nodes upgrade and announce their bandwidth. A ZK-SNARK-based proof takes a node's bandwidth as a public input signal, which is compared to the threshold range of the desired bandwidth of the network. To prevent tampering, the node will commit a salt, which is a large random number that will be added to the bandwidth to generate the xyz hash which will be verified in the proof. Thus, the proof will attest its bandwidth output to the UTXO's script and get validated.

A vote can be taken across producer nodes if there is an increase in unconfirmed transactions that cannot fit into a block. The network in consensus can forbid low bandwidth producer nodes from participating in the election, thus increasing the joining requirement and capacity to hold more transactions. The proofs to elect low bandwidth nodes are based on Semaphore protocol [ref]. The protocol allows users to prove their membership in a group and send vote signals without revealing their original identity. This decentralized pseudonymous voting would increase network trust by protecting against any kind of ballot/vote rigging.

As bandwidth plays a major role in scalable infrastructure, nodes are required to have better of it to achieve maximum production rate per epoch, as elections will be conducted based on it and each node's honesty weight. Nodes willing to participate in the next epoch of block production will be public whereas their real identity is masked by their public keys. For a random input value to commence the next epoch's election, from $range = n$ to m confirmed blocks, the Merkle chain root constructed by validators after resolving forks is taken. For a 2^{256} possible outcome, d times it is divided evenly and finds if any block's MCR lies on the selected r range. With the selected MCR value of a random block, the producers for the next epoch are randomly assigned without repetitive block heights.

2.2 Staking

Bitcoin as a token can be staked for public keys with specified token IDs, where the collateral can be used only once for a block. This results in a stake per token per block bringing the throughput (tps), per token basis. Each token per block's collateral requirement (y) is given in market price (USD) by taking the median volume of all the blocks of the previous epoch. Staked bitcoins can be withdrawn at any time, with no vesting period, except when the block is being created. This brings in a retail and non-custodial solution as opposed to security deposit-type PoS chains. Delegators won't lose their stakes as slashing is done directly on the fees. Bitcoins can be staked to a specific node that chooses to include the stake by collateralizing or locking in its allocated block. Clients can provide their general users with a savings wallet in which they can benefit from an annual percentage yield by providing liquidity for staking. In this way, for a specific token's transaction to be included in a block, the first transaction should prove the collateral. By imposing a timelock of 500 epochs during staking, an additional interest rate (r) is added to restrict the free flow of tokens to increase demand in exchanges. This interest rate is determined by analyzing the market price of the staking token, i.e., Bitcoin from current and previous epochs.

$$x = \epsilon_{n-2}[\widetilde{x}(\{b_1, b_2, \dots b_n\}(\sum_{0}^{n} \Psi T_h(b_n)))]$$

$$\forall r(T_h(\epsilon_n)) = r(T_h(\epsilon_{n-1})) + (\frac{C_{\epsilon_{n-3}} - C_{\epsilon_{n-2}}}{10 \cdot C_{\epsilon_{n-2}}}) \geq 10^{-2}$$

$$y = x + [r(T_h(\epsilon_n)) \cdot (x)]$$

$\widetilde{x}$ = Median Value	ϵ_{n-k} = Target Epoch	r = Interest Rate
T_h = Token ID Hash	b_n = Target Block	C = Closing \$ (ϵ_n)
ϵ_n = Current Epoch	Ψ = UTXO outs in \$	$y = x + r$ for a $T_h(\epsilon_n(b))$

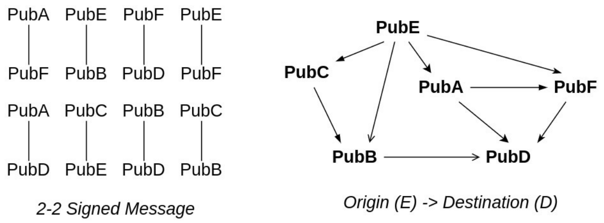
Additionally, a new collateralized floating-rate coin can be issued from lending L1 altcoins which can be used for staking to receive yield benefits.

2.3 Regulation

Regulation via centralized exchanges & custodians risks funds and doesn't encourage a self-custodial ecosystem. A regulator must have the authority to sign or approve transactions. Whitelisting specific hashed public addresses belonging to specific countries, verified and signed by government-assigned client wallets or regulators, by either doing full KYC or something minimal such as mobile number-based OTP verification, could work with maximum privacy.

UTXOs are stamped with region proof on their unlocking script based on specific spending conditions that will only allow a transaction on-chain if taxes are deducted properly. Bitcoin scripts can work efficiently and securely, as opposed to turing-complete smart contracts in this case. Tax models such as capital gains slabs can be issued by governments trustlessly and are validated during the user's script execution. External taxes such as TDS and sales tax can be imposed off-chain with maximum flexibility. These three types of taxes provide almost all the tax models that a regulator can siphon into a centralized payment infrastructure.

2.4 Messaging



2-2 Signed Message

Origin (E) -> Destination (D)

Figure 1: Topology

Delivery of unconfirmed transactions to nodes plays an important role in finality. Shared mempools defile the network with duplicated data, resulting in a poor choice of transactions to include in a block. Miners accept only transactions with higher fees. A direct-messaging system should be deployed with messaging instructions specific to each party as opposed to gossip protocols. Paths are attached with unconfirmed transactions directly from the constructed network graph, which is available to all nodes with public keys as pseudonymous identities protecting privacy. Two peering parties mutually sign a 2–2 random message for every x block and are gossiped across the network to identify the connection as online. All the signed random messages prove that each pubkey signature can display a network topology map from a node's point of reference. Paths are encrypted end-to-end with routing [3] instructions so that the origin cannot be traced. Nodes route transactions to the destinations where producers can attach the transaction to their allocated block. Since the stake information is available in the public ledger, client wallets can construct transactions along with path information that routes to the nearest blocks for instant finality. Transactions are always atomic, providing a solution to queuing issues. Responsibilities are provided to all participants, where nodes only receive the transactions that they need to include, and client wallets should construct shorter paths to provide the best user experience.

2.5 Propagation

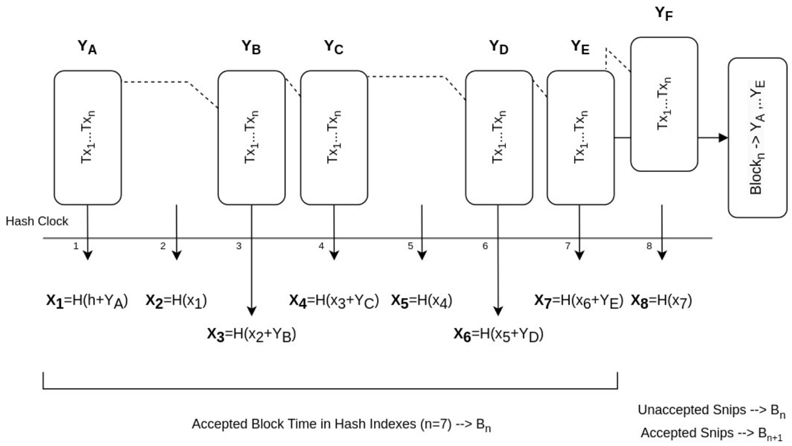


Figure 2: Snips

A Block is collectively validated but constructed as snips - divisible block chunks by the producer and directly messaged to most of the current epoch's producer nodes with routing instructions to gossip across the network. Each snip references the previous snip's hash similar to the chain of blocks for proper identification of each block's snips. For a block of an epoch, a competition to deliver all snips under x time intervals is required to win rewards and avoid slashing of fees. When a block fails to win it will not mint its last of snips which will contain the fees and rewards. VDF proofs [4] are attached for every snip during routing to declare the state of each block's competition and resolve forks. Failed block fees and rewards are slashed by sending them to a burn address when blocks get confirmed by the producers. Each failed block with various categories shall result in decreased block production for the node in the next epoch by adding negative weights which indirectly slashes the bandwidth costing capital which instructs nodes to act honestly with the performance required for the epoch. Null-Blocks are self-minted by validators. Positive weights are given as an incentive for honest and notable performance during the epoch production.

To synchronize time, each node's hash rate per second of a specific hash function is proved cryptographically onchain and taken in multiples of a common hardware's hash rate (R). An Individual hash-rate proof is provided along with bandwidth proof for every epoch which trustlessly synchronizes all nodes as a single hardware producing continuous hashes concated with all snips. The ZK IHR proofs are algorithmically similar to the bandwidth proofs, where the hash rate provided by the node is compared to the threshold range of the desired hash rate and salt is used to prevent tampering attacks. This provides cryptographic timestamped proofs to announce each epoch's block time, under which all snips have to arrive and win the time-based propagation competition. The competition is termed "Proof of Speed".

2.6 Rewards

Rewards are given for each snip hash concated with transactions validated by VDF proofs. Since newly minted bitcoins for a period of time are definite and each epoch's sum of all block time is capped, new bitcoins can be supplied exactly proportional to the current Bitcoin issuance rate with halving. Each hash represents work done by nodes on validating and propagating transactions. When a fork arises due to rejected snips, the rest of the block time and its allocated rewards are slashed. Meanwhile, when a block is fully minted, the rest of its allocated rewards are distributed to the next block producers of the epoch. This encourages nodes to attest to VDF proofs for receiving snips at the earliest. While Tax outputs are attached as zero input transactions within the snip it contains, the rewards and fee outputs when accounted as a separate snip mark the end of a block.

During staking, producers announce their accepted tokens for which they will directly withdraw the commission. For other tokens, delegators can stake with a condition that their stake in bitcoins or the accepted stake token will be traded for the collected fees. During the commission withdrawal of non-accepted tokens, the producer will deposit collected fees to delegators and inflate the stake 1:1 ratio to withdraw the collateral. Users can pay in any token, delegators incur the risk, and producers get paid in tokens of their choice to validate transactions.

2.7 Renting

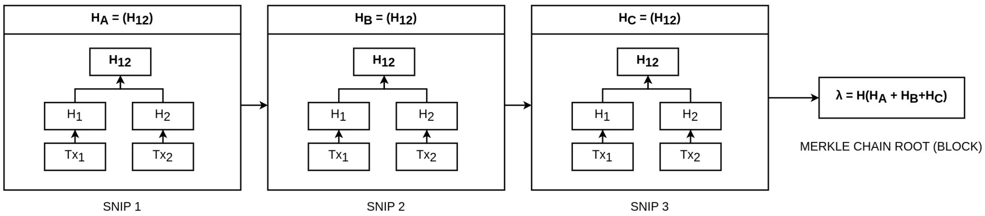


Figure 3: Merkle Chain

Instead of taking the Merkle roots of all the transactions inside a block, a snip's Merkle root is taken and linearly hashed to find the Merkle Chain root. Since snips can be rejected by validators, it is unsure to predict a Merkle chain root giving it a purely randomized value. Inside a snip contains parsed transactions whose hashes are taken to find the snip's Merkle root can be pruned if the UTXOs are spent, burnt, or expired. Each UTXO expiry block height is embedded in its script, and can be scanned by nodes, and pruned to optimize their data storage. Client Wallets can store each of their users' transaction history and can be audited onchain using Merkle chain roots. Renting rates can be given in market price independently voted by producer nodes for every epoch per byte per block. Users cannot directly pay for rent, but rather each new UTXO created is charged a transfer fee in the range of 0.05% - 0.005% decided based on the total volume of all transactions settled on previous epochs.

$$5 \cdot 10^{-4} \leq \Delta f \leq 5 \cdot 10^{-5} == \sigma[\Psi(\epsilon_{n-3}, \epsilon_{n-2})] > 0.75$$

$$f' = f + x \begin{cases} x = +5 \cdot 10^{-6}, & \text{if } \Psi(\epsilon_{n-3} > \epsilon_{n-2}) \\ x = -5 \cdot 10^{-6}, & \text{otherwise} \end{cases}$$

$F = \text{UTXO value } \times f'$

Transfer fee charges more fees for higher value utxos and less for lesser value utxos bringing ease to transact for retailers. According to how much each utxo pays for a transfer fee, an expiry date is set. UTXOs doing state updates will not be charged and can combine UTXOs to a single balance holding UTXO with increased expiry value. This encourages users to store a single UTXO per wallet reducing transaction fees and also incentivizes nodes, clients, etc by saving disk space.

2.8 Forks

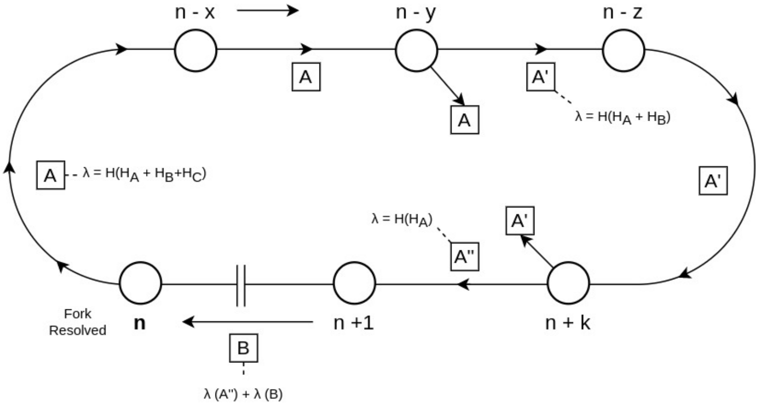


Figure 4: Resolving Forks

Confirmed blocks as snips are streamed with an included path that propagates across every producer in a backward ring manner. The snips are attached with VDF proofs by producers which provide proof of block time. Forks arise when different producers reject extra snips, this can be resolved by the very next block, providing instant finality to users. Additionally, negative weights are added if the fork arises after the first destination node of the path with VDF evaluation and also if a producer's VDF proof isn't attested which directly denotes offline activity. In such cases, producers construct new paths to deliver the snips. Block size and time are decided on consensus per epoch to prevent forks, which denotes a healthy propagation. Bandwidth upgrades are needed for the nodes which initiate the fork and directly instruct the origin block size should be decreased.

$$\begin{aligned}\lambda(B_A) &= H(H_A + H_B + H_C) \\ \lambda(B_{A'}) &= H(H_A + H_B) \\ \lambda(B_{A''}) &= H(H_A) \\ \lambda(B_B) &= H(\lambda(B_{A^?}) + \lambda(B_{B^?}))\end{aligned}$$

For a block, possible Merkle Chain Roots are constructed and forks are resolved. The next blocks will have the input hash value of the previous block's Merkle Chain Root. To avoid influence by the first destination node, path planning can be randomized but must include the next block producer $(n + 1)$ as the final destination.

2.9 Oracles

Oracle data provide real-time market rates to unlocking scripts for every x interval of blocks. Various oracle protocols [5] fall to freeloading and duplication issues on trustless oracle systems. A per-feed subscription-based price-oracle protocol is introduced with participants - exchanges, nodes, and users. Exchanges provide real-time market rates along with traded volume for weighted aggregation of data. Nodes subscribe per request to exchange price feeds for each token id. Each exchange's feeds will have a reputation score similar to prediction markets maintained by users to avoid faulty data. Exchanges are incentivized by subscription fees, nodes provided with positive weights for the next elections, and users profit from removing faulty data oracles. Freeloading is avoided by exchange participation and equally distributing positive weights per feed to producer nodes.

3 Treasury

For the active development and sustainability of the project, a DAO Treasury is set up to fund developers and the community. A minimal commission is imposed on producer fees and deposited to a treasury script. Memberships are non-fungible and non-transferable 1. Temporary (Core-Developers), 2.Permanent (Investors, Community), 3. Contracts (Employees, Operations). Each has its privileges provided to developers, investors, and community members.

Votes can decide to add or remove contracts stating x amount per y term for z period withdrawn during payouts. Votes can also decide to kick a temporary member due to any circumstantial issues or failure in active contribution. Permanent members cannot be kicked out due to their external contribution to building the project. Permanent and Temporary members cannot be added after the genesis epoch but can appoint heirs to their membership. Contracts are paid out initially and the rest is provided to other memberships as dividends according to their weight. Core-Developers are only rewarded for their active contribution, not indefinitely like holding a fungible token. A decentralized open-sourced organization structure is maintained with decisions involving votes bringing forth sustainable growth for the future of project blink.

4 Future

The future of Bitcoin includes building L1 applications such as exchanges, bridges, lending & borrowing, insurance, and mirrored wallets with current banking. Since these applications are developed inside an unlocking script, it requires preimage construction off-chain and settles onchain - inheriting the security of Bitcoin that centralized applications don't offer. Privacy can be improved by obscuring amounts and including ring signatures similar to Monero assisting regulators and masking financial information of a specific country. Emphasis on Zk delivery and validation of snips can reduce influence attacks in the future.

Since Layer 1 is dynamically scalable, the need for an offline-based off-chain digital cash-payment system will be developed with suitable hardware wallets. To provide a decentralized programmable environment, EVM [6] will be deployed as a layer that can update its global-state & contract state by providing a gas limit through a receipt-proof paid in any token. These EVM contracts shall not contain balances and EOAs, but rather purely for logic such as building DApps without a financial scope. Room for further improvement resides in sharding and size-optimized scripts upgraded to Bitcoin and EVM Layer. Moreover, research will be conducted to merge various bitcoin and altcoin chains for a clean experience on the whole of finance and computing.

Implementations can be contributed openly to `https://github.com/projectblink`

References

[1] S. Nakamoto, "Bitcoin : A peer-to-peer electronic cash system," *Whitepaper*, p. 9, 2008.

[2] "scrypt — a high level smart contract language for bitcoin sv." `https://scrypt.io/`.

[3] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," 2016.

[4] A. Yakovenko, “Solana: A new architecture for a high performance blockchain v0. 8.13,” *Whitepaper*, 2018.

[5] S. Ellis, A. Juels, and S. Nazarov, “Chainlink : A decentralized oracle network v1,” *Whitepaper*, 2017.

[6] G. Wood *et al.*, “Ethereum: A secure decentralised generalised transaction ledger,” *Ethereum project yellow paper*, vol. 151, no. 2014, pp. 1–32, 2014.

Appendix

A Zk Circuits

A.1 Bandwidth Proof

A.2 VoC Vote (Ownership Proof)

The trapdoor and nullifier are private values of the semaphore identity. It uses Poseidon hash function to create the identity commitment using these signals and generate a binary Merkle tree that stores the public identity commitment of each member

A.3 IHR Proof

B Onchain Protocols

B.1 Election

B.1.1 Partcipation

For every epoch before the allocation of blocks, the joining requirement is increased by eliminating certain nodes' bandwidth proof by conducting a vote with current producers.

$$|\mathbb{R}| = \{P(B)_1, P(B)_2, ...P(B)_n\}$$

Vote Participants,

$$\forall|\mathbb{R}| \begin{cases} |\mathbb{R}|\backslash\{\{P(B)_n\}\}\wedge \in |\mathbb{R}'|, & \text{if } P(B)_n < \bar{x}(|\mathbb{R}|) \\ P(B)_n \in |\mathbb{R}|, & \text{otherwise} \end{cases}$$

After Seed,

$$\forall\mathbb{R}'k = \sum_{P(B)'_n \in|\mathbb{R}|} P(B)'_n$$

$$\forall\mathbb{R}' \begin{cases} |\mathbb{R}'|\backslash\{\{P(B)_n\}\}, & \text{if } k > \bar{x}(|\mathbb{R}|) \cdot (51 \cdot 10^{-2}) \\ P(B)_n \in |\mathbb{R}'|, & \text{otherwise} \end{cases}$$

Results,

$$|\mathbb{R}| = |\mathbb{R}|\cup|\mathbb{R}'|, i = \bar{x}(|\mathbb{R}|), |\mathbb{R}| = \emptyset$$

$\lvert\mathbb{R}\rvert$	= Block Producers	$\lvert\mathbb{R}''\rvert$	= Producers to be voted
$P(B)_n$	= PubKey's Bandwidth in bps	$P(B)'_n$	= Voted Pubkey's bps
$\bar{x}$	= Mean value	i	= Joining requirement in bps

B.1.2 Producer Allocation

Blocks are allocated for a whole epoch from a definite randomized seed where nodes construct the same predictable result. Each producer's bandwidth proof $> i$ will provide the production rate in the number of blocks.

$$\text{Currently } |\mathbb{R}| = \emptyset, \text{If } \epsilon_{(n)}\{P(B)_n...\} > i \in |\mathbb{R}|$$

Block Size/sec,

$$\epsilon_n(j) = min(|\mathbb{R}|) \cdot l$$

Weights addition,

$$\forall|\mathbb{R}|, P(B)_n = \sum w(P) + P(B)_n = P(w)_n \in |\mathbb{R}''|$$

Allocation,

$$\forall|\mathbb{R}''|, \lfloor\Phi_n\rfloor = \frac{x \cdot P(w)_n}{10^4 \cdot \sum_{P(w)_n \in |\mathbb{R}''|} P(w)_n} \in |\mathbb{N}|$$

x	= No. of blocks per epoch	Φ	= Allocated no. of blocks per producer
l	= Latency	$\lvert\mathbb{R}''\rvert$	= Producers with weight
w	= Individual weights	$\lvert\mathbb{N}\rvert$	= Producers with no. of blocks allocation

B.1.3 Random Seed

To randomize the election, between a block range, a seed(k) - Merkle Chain Root from a confirmed block is found. This seed should lie between a selected range in 2^{256} base10 possibilities. The difficulty rate is adjusted every epoch to seed entropy. If a seed is not found, the last block of the range's MCR is taken.

$$|\mathbb{S}| = \{1, ..2^{256}\} \text{ where, } k \in |\mathbb{S}|$$

Range Identify,

$$|\mathbb{F}| = \{[(\frac{p}{d} \cdot q) + h], ...(p = 2^{256})\} \equiv |\mathbb{S}|$$

$$h = 0,\ h++;\ q = 0,\ q++ \text{ every } \frac{p}{d}$$

Value k,

$$|\mathbb{X}| = \{[(\frac{p}{d} \cdot q) + h], ...[(\frac{p}{d} \cdot q) + h]\},$$

$$\text{where } q \leq k' \leq q \in |\mathbb{X}| \subset |\mathbb{F}|$$

Difficulty level,

$$k_{10} \text{ is taken} \begin{cases} d = 1, k = k', & \text{if } k' \text{ is ¿}, \exists k \in |\mathbb{X}| \\ d++, k = k', & \text{if } \exists k \in |\mathbb{X}| \\ d-- \geq 2, k = k', & \text{if } \neg\exists k \in |\mathbb{X}| \vee \exists k \in |\mathbb{F}|\{p\} \end{cases}$$

Block height allocation,

$$Z_{16} = MD160_{16}((SHA256_{16}(k_{16} + b_h + \epsilon_h))$$

$$Z_{16} \begin{cases} b_h++, P(\Phi_n) \in |\mathbb{L}|, \Phi_n = \Phi_n - 1 \in |\mathbb{N}|, & \text{if } |\mathbb{N}| \neq \emptyset \wedge \forall\Phi_n \in |\mathbb{N}| < Z_{16} \\ MD160_{16}(Z_{16}), & \text{if } |\mathbb{N}| \neq \emptyset \wedge \forall\Phi_n \in |\mathbb{N}| > Z_{16} \\ |\mathbb{N}|\{\Phi_n\}\backslash|\mathbb{N}|, & \text{if } \Phi_n = 0 \\ \{n(|\mathbb{L}|) + 1, ..x\} = |\mathbb{L}|\Phi_n = \{1, ..\}, & \text{if } |\mathbb{N}| = \emptyset \wedge x - n(|\mathbb{L}|) = \mathbb{N} \\ |\mathbb{L}|, & \text{if } |\mathbb{N}| = \emptyset \wedge n(|\mathbb{L}|) = x \end{cases}$$

Block Time,

$$\epsilon_n b(\tau) = \widetilde{x}(\epsilon_{n-2}(b_1^x)) \in |\mathbb{B}|$$

Block Size,

$$Blocksizehere(b_s)$$

b_h = Block height in base_{10}
ϵ_h = Epoch height in base_{10}
$P(\Phi_n)$ = Pubkey in MD160 base_{16} rep

B.2 Network Graph

B.3 Routing Path

B.4 Hash-Reward

Halving of Bitcoin issuance is done every $R \cdot (1.26 \cdot 10^8)$ hashes equals 210,000 10-minute bitcoin blocks. Every epoch (ϵ_n) will have a g amount of bitcoins to be issued. Hence for every epoch a target $g(\epsilon_n)$ new bitcoins is set to mint. For forked blocks, the supply is slashed. For fully minted blocks with remaining hashes, the rewards are distributed to the next hashes equally incentivizing producers.

$$g[R(\epsilon_n)] = \frac{\Delta g(\epsilon_n)}{\epsilon_n(\sum_1^n R)}$$

Rewards per block,

$$g[R(\epsilon_n(b))] = \epsilon_n(b(\sum_1^n R)) \cdot g[R(\epsilon_n)]$$

$$\Delta g(\epsilon_n) = g(\epsilon_n) - [\sum_1^n g[R(\epsilon_n)] \in b(\epsilon_n)]$$

Slashing forks $\Delta\lambda$,

$$\forall \epsilon_n(b_n), \Delta g(\epsilon_n) = \begin{cases} g(\epsilon_n) - g(\epsilon_n(b_n)), & \text{if } \Omega(b_n) = 0 \\ \Delta g(\epsilon_n) + b, & \text{if } \Omega(b_n) = \mathbb{N} \wedge \lambda \\ \Delta g(\epsilon_n) - b, & \text{if } \Omega(b_n) = \mathbb{N} \wedge \Delta\lambda \end{cases}$$

$$\Omega(b_n) = g(\epsilon_n(b)) - [\sum_1^n g[R(\epsilon_n)] \in b_n(\epsilon_n)]$$

B.5 Merkle-Chain

B.6 VDF & Forks

VDF how it works - continuous hash in a single thread, with concated snips data, each snip is bunch of transactions in linear.

Snip = pre-image + snip(transactions++)

all should attach preimage

When validating hash continuous with pre-image, if not empty snip, hash continuous - Find sum of hashes

Block time τ in given in number of hashes in global

local/global = intermediate

For each pubkey different intermediate hash

If sum of all snips ¡= intermediate hash - finish block

Each producer will have different snip data with merkle chain

To resolve, previous block with merkle chain root concated with next block.

All producers upgrade it, fork resolved

B.7 Weights

Weights shall increment or decrement the posted bandwidth costing capital to nodes due to dishonest, lower performance or bandwidth.

$$\begin{aligned}
\text{Null Block} &= -0.015 \cdot (\widetilde{x}|\mathbb{R}|)(\text{Blocks with no snips}) \\
\text{Fork}^{1deg} &= -0.005 \cdot (\widetilde{x}|\mathbb{R}|) \text{ by first destination} \\
\text{Forks}^{2deg} &= -0.07 \cdot (\widetilde{x}|\mathbb{R}|) \text{ by other destinations} \\
\text{Fork Proof}^{1deg} &= +0.005 \cdot (\widetilde{x}|\mathbb{R}|) \\
\text{Fork Proof}^{2deg} &= +0.07 \cdot (\widetilde{x}|\mathbb{R}|) \\
\text{Age} &= +0.03 \cdot (\widetilde{x}|\mathbb{R}|) \\
\text{Extras} &= +x \cdot (\widetilde{x}|\mathbb{R}|)
\end{aligned}$$

Null Block weight is added if a producer fails to add any snip. Forks of degree 1 refer to the fork created by the first destination producer. Forks of degree 2 refer to the fork created by further destinations. Both the sender and receiver shall incur the negative weight equally shared during forks. Fork proofs are provided by the producer before the fork, incentivized for their honesty. Age is a positive weight provided to producers with full block production without forks for the epoch. Extras is given for further native protocols such as oracles, etc.

B.8 Public Keys

-governments should not ban -same time governments should impose civil law -public keys should not be reused by criminals -governments should add -users should withdraw between favourable governments -Can have withdrawal period

C Bitcoin Scripts

C.1 Taxes

C.2 Stake Proof

Delegators provide bitcoins or the staking token for accepted token id to nodes. For non-accepted tokens, delegators ($h1 \in D_T$) can create their script along with specifying node (M).

$$\mathbb{SS} = lim, id, R, D_T(am_n, Pub_k, e_i), P(am_n, Pub(M), er_i), F_t(am, er_i). \sum D_T(am_n)$$

When deposited, each stake of delegators (D_T) is identified along with its (k) Pubkey and (er_i) initial exchange rate to deduct gains tax during withdrawal (W). Each script ($\mathbb{SS}$) will have a limit (lim) in bitcoins or staking token along with id for what token (t) it is being staked.

$$W(\mathbb{SS}) \leq \begin{cases} [D_T(am_n) \cdot R] + [F_t(am) \cdot \frac{D_T(Pub_k(am_n))}{\sum D_T(am_n)}], & \text{if } D_T(Pub_k) \\ o = F_t(am) \cdot C & \text{if } P(Pub(M)) \wedge Pub(M) = h1 \in D_T \\ \frac{(\sum D_T(am_n) \cdot er_c(T)) - (o \cdot er_c(t))}{er_c(T)} & \text{if } P(Pub(M)) \wedge Pub(M) \neq h1 \in D_T \end{cases}$$

During the production of blocks, producers $P(M)$ can reject their stake script transactions to freeze the collateral as proof for validation. Each script can only be proved once where the sum of all deposits will be made constant. For accepted tokens, validators create outputs for themselves and deposit fees. For non-accepted tokens producers inflate the ratio (R), to withdraw staked tokens (T) for commissions.

$$y = (F_t(am) \cdot er_c)(C)$$
$$R = \frac{[\sum D_T(am_n)] - y}{\sum D_T(am_n)}$$

C.3 Oracle

Oracle Feed Funding Positive Weight Reputation

C.4 DAO

Anonymous Proposal Voting Protocol Membership Protocol