Akash: Token Model & Mining Economics

Greg Osuri (Akash Network)

September 2, 2019

This paper covers the economics of the Akash Network and introduces the Akash Token (AKT). In this paper, we present various incentives for ensuring the economic security of the Akash ecosystem as well as drive adoption. We propose an inflationary mechanism to achieve economic goals along with estimates for mining rewards and inflation rates. Additionally, we also present mechanisms for allowing a multitude of fee tokens, which will improve the user experience of using the blockchain.

> NOTE: This paper is a work in progress and intended for PRIVATE distribution only.

1 Introduction

Akash is a permissionless marketplace trade compute cycles. In this paper, we present the Akash Token (AKT) model that's designed to a) maintain ecosystem sovereignty, b) provide economic security, and c) encourage early adoption. Some definitions:

Akash Token (AKT) AKT is the native token of the Akash Network. The core utility of AKT is to act as a staking mechanism to secure the network and normalize compute prices for the marketplace auction. The amount of AKTs staked towards a validator defines the frequency by which the validator may propose a new block and its weight in votes to commit a block. In return for bonding (staking) to a validator, an AKT holder becomes eligible for block rewards (paid in AKT) as well as a proportion of transaction fees (paid in any of the whitelisted tokens).

Validator Validators secure the Akash network by validating and relaying transactions, proposing, verifying and finalizing blocks. There will be a limited set of validators, initially 64, which are required to maintain a high standard of automated signing infrastructure. Validators charge *delegators* a commission fee in AKT.

Delegator Delegators are holders of the AKT and use some or all of their tokens to secure the Akash chain. In return, delegators earn a proportion of the transaction fee as well as block rewards.

Provider Providers offer computing cycles (usually unused) on the Akash network and earn a fee for their contributions. Providers are required to maintain a stake in AKT as collateral, proportional to the hourly income earned; hence, every provider is a delegator and/or a validator.

Tenants Tenants lease computing cycles offered by providers for a market-driven price (set using a reverse auction process).

1.1 Marketplace Overview

Akash provides a novel spot market to lease containers. A container is a unit of computing ($U \equiv \{CPU, Memory, Disk\}$), which can run any type of cloud application. With Akash, early

adopters can enjoy over 8x lower cost than the current market.

All marketplace transactions are persisted on the Akash blockchain. To lease a container, the tenant (developer) requests a deployment by specifying the type(s) of unit(s), and the quantity of each type of unit. To specify a type of unit, the tenant specifies attributes to match, such as region (e.g. US) or privacy features (e.g. Intel SGX). The tenant also specifies the maximum price they're willing to pay for each type of unit.

An order is created in the order book (upon acceptance by a validator).

The provider(s) that match all the requirements of the order then place a bid by competing on price using a `SubmitFulfillment` transaction. The provider that bids the lowest amount on the `Order` wins, upon which a `Lease` is created between the tenant and the provider for the order.

1.2 Proof of Stake using Tendermint Consensus

Akash employs a blockchain secured by a *Proof-of-Stake* consensus model as a Sybil resistance mechanism for determining participation in its consensus protocol and implements the Tendermint (Buchman, Kwon, and Milsosevic, n.d.) algorithm for Byzantine fault-tolerant consensus. Tendermint was designed to address the speed, scalability, and environmental concerns with Proof of Work with the below set of properties:

a) Validators take turns producing blocks in a weighted round-robin fashion, meaning the algorithm has the ability to seamlessly change the leader on a per-block basis.

b) Strict accountability for Byzantine faults allows for punishing misbehaving validators and providing economic security for the network.

Anyone who owns an Akash token can bond (or delegate) their coins and become a validator, making the validator set open and permissionless. The limited resource of Akash tokens acts as a Sybil prevention mechanism.

Voting power is determined by a validator's bonded stake (not reputation or real-world identity). No single actor can create multiple nodes in order to increase their voting power as the voting power is proportional to their bonded stake. Validators are required to post a "security deposit" which can be seized and burned by the protocol in a process known as "slashing".

These security deposits are locked in a bonded account and only released after an "unbonding period" in the event the staker wishes to unbond. Slashing allows for punishing bad actors that are caught causing any attributable Byzantine faults that harm the well-functioning the system.

The slashing condition and the respective attributable Byzantine faults and punishments are beyond the scope of this paper.

1.2.1 Limits on Number of Validators

Akash's blockchain is based on Tendermint consensus which gets slower with more validators due to the increased communication complexity. Fortunately, we can support enough validators to make for a robust globally distributed blockchain with very fast transaction confirmation times, and, as bandwidth, storage, and parallel compute capacity increases, we will be able to support more validators in the future.

On Genesis day, the number of validators V_i is set to $V_i(0) = V_{i,0} = 64$ and the number of validators at time t *year* will be:

$$V_n(t) = |\log_2(t+1) \cdot V_{i,0}| \tag{1}$$

So, in 10 *years*, there will be $V_n(10) = 442$ validators as illustrated in fig. 1

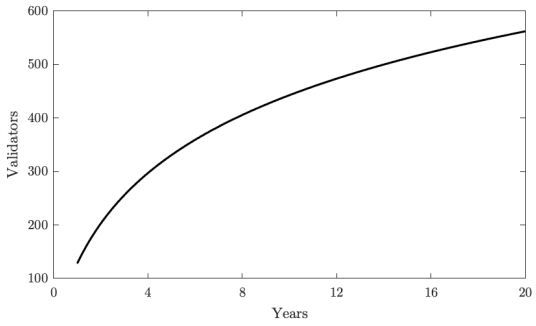

Figure 1: Validators limit over the years

2 The Akash Token Model

2.1 Native Token (AKT)

The primary functions of AKT are in staking (which provides security to the network) and in acting as a unit of measure for pricing all currencies supported by the marketplace. AKT is expected to have very low liquidity because of the high earning potential of staking rewards. Although AKT can be used for settling transactions in the marketplace, it is not intended to be used to pay a fee or to be used as a currency, because of its highly illiquid nature. However, transaction fees and block rewards are denominated in AKT. The income stakers earn is proportional to the tokens staked and length of staking commitment.

2.2 Settlement & Price Volatility Protection

The lease fees are denominated in AKT, but they can be settled using any whitelisted tokens. There is an option to lock in an exchange rate between AKT and the settlement currency. This way providers and tenants are protected from the price volatility of AKT expected to result from its low liquidity.

For example, suppose a lease is set to 10 $AKTs$ and locks an exchange rate of 1 $AKT = 0.2\ BTC$. If the price of AKT doubles, i.e., 1 $AKT = 0.4\ BTC$, the tenant is required to pay 5 AKT. Conversely, if the price of BTC doubles while keeping the price of AKT same, i.e., 1 $AKT = 0.1\ BTC$, then the tenant is required to pay 20 AKT.

2.3 Fees Using a Multitude of Tokens

In order to avoid issues of network abuse (e.g. DOS attacks), all transactions and leases on Akash are subject to fees. Every transaction has a specific associated fee, `GasLimit`, for processing the transaction, as long as it does not exceed `BlockGasLimit`.

The `GasLimit` is the amount of gas which is deducted from the sender's account balance to issue a transaction. All leases (purchases) require the tenant (buyer) to pay `TakeFee` and the seller (provider) to pay a `MakeFee`.

Unlike most other blockchain platforms, such as Ethereum (Wood, n.d.), Bitcoin (Satoshi, n.d.), and Neo ("NEO Whitepaper," n.d.), Akash accepts a multitude of tokens for fees, whereas the mentioned platforms require fees to be paid in the platform's native cryptocurrency. Each validator and provider on Akash can choose to accept any currency or a combination of currencies as fees.

The resulting transaction fees, minus a network tax that goes into a reserve pool, are split among validators and delegators based on their stake (amount and length).

2.4 Transaction Ordering using Consensus Weighted Median

In order to prioritize transactions when multiple tokens are used, validators need a mechanism to determine the *relative value* of the transaction fee. For example, let us assume we had a perfect oracle to inform us that the relative value of BTC is 200 AKT, and that of ETH is 0.4 AKT. Suppose we have two transactions of equal gas cost, and the transaction fees on them are 10 BTC and 6000 ETH, respectively. The first transaction's fee is equivalent to 2000 (10 x 200) AKT and the second

transaction's fee is equivalent to 2400 (6000 x 0.4) AKT. Then the second transaction will have a higher priority.

In order to get these relative values without using an oracle, we can employ a *Consensus Weighted Median using Localized Validator Configuration* (Aggarwal, n.d.) mechanism.

In this method, each validator maintains a local view of the relative values of the tokens in a config file which is periodically updated, and the relative value is achieved by using a weighted mean, meaning they submit their "votes" for the value of each token on-chain as a transaction.

Lets say for example, there are five validators $\{A, B, C, D, E\}$ with powers $\{0.3, 0.3, 0.1, 0.1, 0.2\}$ respectively. They submit the following votes for their personal views of each token:

$\mathbf{A} : \mathsf{AKT} = 1, \mathsf{BTC} = 0.2$

$\mathbf{B} : \mathsf{AKT} = 2, \mathsf{BTC} = 0.4$

$\mathbf{C} : \mathsf{AKT} = 12, \mathsf{BTC} = 2$

$\mathbf{D} : \mathsf{AKT} = 4, \mathsf{BTC} = 1$

$\mathbf{E} : \mathsf{AKT} = 1.5, \mathsf{BTC} = 0.5$

These values are stored on-chain in a ordered list along with their validator that placed the vote.

$\mathsf{AKT} : [1_{\mathbf{A}}, 1.5_{\mathbf{E}}, 2_{\mathbf{B}}, 4_{\mathbf{D}}, 12_{\mathbf{C}}]$

$\mathsf{BTC} : [0.2_{\mathbf{A}}, 0.4_{\mathbf{B}}, 0.5_{\mathbf{E}}, 1_{\mathbf{D}}, 2_{\mathbf{C}}]$

The proposer takes a weighted mean (by stake) of the votes for each whitelisted token to determine a consensus relative value of each token, where $\bar{w}(x_n) = WeightedMean(x_n)$:

$\mathsf{AKT} : \bar{w}([1, 0.3], [1.5, 0.2], [2, 0.3], [4, 0.1], [12, 0.1])$

$\mathsf{BTC} : \bar{w}([0.2, 0.3], [0.4, 0.2], [0.5, 0.2], [1, 0.1], [2, 0.2])$

which give us the relative value for each token: $\mathsf{AKT} = 2.8$ and $\mathsf{BTC} = 0.58$ respectively.

2.5 Maker and Taker Fee Schedule

Leasing compute is either zero-fee or for a small fee, depending on the user's activity in the last 30 days. Lease fees have a distinction of a `Maker` fee or a `Taker` fee. A ***taker fee*** is paid when you add computing capacity to Akash network by fulfilling an order in the order book when the lease is created. A ***maker fee*** is paid when you request computing capacity from Akash by placing an order in the book when the lease is created.

For a lease l, the *aggregate trade activity factor* of the a stakeholder of the lease for a given time t is defined by:

$$\kappa_l = \frac{\sum_{t=1}^{T_a} l(1-t)}{\sum_{t=1}^{T_a} L(1-t)}, \tag{2}$$

where $L(t)$ is the aggregate trade activity of the network and $T_a = 30\ days$.

Assuming that $\mathcal{P}_l = 0.5$ is the *fee distribution factor*, the maker fee $R_{mk}(l)$ will be:

$$R_{mk}(l) = \frac{10^{1-4\mathcal{P}_l}}{\log(10^{4\mathcal{P}_l})} \cdot \log\left(\frac{\mathcal{P}_l}{\min(\kappa_l, \mathcal{P}_l)}\right), \tag{3}$$

and the taker fee $R_{tk}(l)$ will be:

$$R_{tk}(l) = \frac{10^{1-\frac{7\mathcal{P}_l}{2}}}{\log\left(10^{\frac{7\mathcal{P}_l}{2}}\right)} \cdot \log\left(\frac{\mathcal{P}_l}{\min(\kappa_l, \mathcal{P}_l)}\right). \tag{4}$$

3 Token Economics and Incentives

Providers earn income by selling computing cycles to tenants who lease computing services for a fee. However, in the early days of the network, there is a high chance the providers will not be able to earn a meaningful income due to a lack of sufficient demand from the tenants (consumers of computing), which in turn hurts demand because of lack of supply.

To solve this problem, we will incentivize the providers using inflation by means of block rewards until a healthy threshold can be achieved.

In this section, we describe the economics of mining and Akash Network's inflation model. An ideal inflation model should have the following properties:

- Early providers can provide services at ex-

ponentially lower costs than in the market outside the network, to accelerate adoption.
- The income a provider can earn is proportional to the number of tokens they stake.
- The block compensation for a staker is proportional to their staked amount, the time to unlock and overall locked tokens.
- Stakers are incentivized to stake for longer periods.
- Short term stakers (such as some bear market participants) are also incentivized, but they gain a smaller reward.
- To maximize compensation, stakers are incentivized to re-stake their income.

3.1 Motivation

Akash Network intends to gain early adoption by offering exponential cost savings as a value proposition for tenants, and the efficiency of a serverless infrastructure as an additional value proposition for tenants and providers. These value propositions are extremely compelling, especially for data and compute intensive applications such as machine learning.

3.2 Stake and Bind: Mining Protocol

A provider commits to provide services for at least time T and intends to earn service income r every compensation period $T_{comp} = 1\ day$. Providers stake Akash tokens s and specify an unlock time t_1, where minimal lock-time $t_1 - t$ should not be less than $T_{min} = 30\ days$. Additionally, they delegate (voting power) to validator v by bonding their stake via `BindValidator` transaction.

A staker is a delegator and/or a validator to whom delegators delegate. Every provider is a staker, but not every staker is a provider; there can be stakers who are pure delegators providing no other services, and there can be stakers who are pure validators providing no other services.

At any point, a staker can: a) Split their stake (or any piece of their stake) into two pieces. b) Increase their stake l by adding more AKT. c) Increase the lock time T, where $T > T_{min}$.

Stakers choose to split their stake because the compensation is dependent on lock time L which will be addressed in later sections.

3.3 General Inflation Properties

3.3.1 Initial Inflation

If we assume Akash will have the same number of tokens locked as NuCypher (Egorov, Wilkinson, and, n.d.) and DASH (Duffield and Diaz, n.d.): $\lambda = 60\%$, then $1 - 40\%$ of the supply of AKT will be in circulation. The adjusted inflation rate for inflation, I will be:

$$I^* = \frac{I}{1-\lambda}, \tag{5}$$

Considering that ZCash ("ZCash Emmission Rate," n.d.) had $I^* = 350$ (turn around point during the overall bull market), which makes $I = 140\%APR$, it is reasonably safe to set the initial inflation to be $I_0 = 100\%APR$ (meaning 1/365 per day).

3.3.2 Inflation Decay

Assume that all miners have the maximum compensation rate. We define the inflation decay factor (time to halve the inflation rate) to be $T_{1/2} = 2\ years$ in this case. Inflation depending on the time passed from the Genesis t, then looks like:

$$I(t) = I_0 \cdot 2^{-\frac{t}{T_{1/2}}} = I_0 \exp\left[-\ln 2 \frac{t}{T_{1/2}}\right], \tag{6}$$

In this case, the dependence of the token supply on the time t is:

$$M(t) = M_0 + \int_0^t I(t)\,dt = M_0 + \frac{I_0 T_{1/2}}{\ln 2}\left[1 - 2^{-\frac{t}{T_{1/2}}}\right] \tag{7}$$

If we let I_0 be the relative inflation rate, then $I_0 = i_0 M_0$. For 100% APR, $i_0 = 1$ and $I_0 = M_0$, which gives us the maximum number of tokens which will ever be created (as illustrated in fig. 2):

$$M_{\max} = M(\infty) = M_0\left(1 + \frac{i_0 T_{1/2}}{\ln 2}\right) \approx 3.89\, M_0, \tag{8}$$

where M_0 is initial number of tokens.

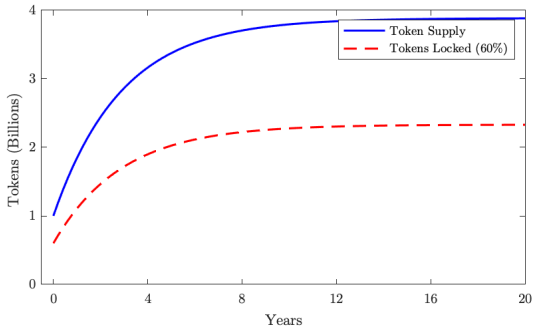

Figure 2: Token supply and tokens locked over years with an initial inflation of 100% APR that is halving every 2 years

3.3.3 Staking Time and Token Creation

We will reward the full compensation ($\gamma = 1$) to the stakers who are committed to stake at least $T_1 = 1$ *year* (365 days). Those who stake for $T_{\min} = 1$ *month* will get close to half the compensation ($\gamma \approx 0.54$). In general,

$$\gamma = \left(0.5 + 0.5\frac{\min(T_i, T_1)}{T_1}\right), \tag{9}$$

$$T_{i,\text{initial}} > T_{\min}, \tag{10}$$

where the unlocking time T_i means the time left to unlock the tokens: $T_i = t_1 - t$. t_1 is the time when the tokens will be unlocked, and t is the current time. The initial T_i cannot be set smaller than $T_{min} = 1$ *month*, but it eventually becomes smaller than that as time passes and t gets closer to t_1.

Shorter stake periods (for lower rewards) result in a lower daily token emission. Considering, miners will most likely stake for short periods during a bear market; lower emissions will provide much better price and stability as a result.

The emission half decay time $T^*_{1/2} = T_{1/2}/\gamma^*$, where γ^* is the mean staking parameter, is also prolonged when $\gamma < 1$. $T_{1/2}$ prolongs to 4 years instead of 2 if all stakers have $\gamma^* = \gamma = 0.5$.

The total supply over time (eq. 7) at $\gamma^* \neq 1$ will then look like:

$$M(t) = M_0\left[1 + \frac{i_0 \gamma^* T^*_{1/2}}{\ln 2}\left(1 - 2^{-\frac{t}{T^*_{1/2}}}\right)\right]. \tag{11}$$

3.4 Delegate Pool Distribution

The exponential is a solution of a differential equation where inflation is proportional to the amount of not yet mined tokens:

$$I(t) = \frac{\ln 2}{T_{1/2}}\left(M_{\max} - M(t)\right) \tag{12}$$

$$dM = I(t)\,dt. \tag{13}$$

where $M(t)$ is the current token supply with $M(0) = M_0$ and dt can be equal to the mining period (1 day). Each validator can trivially calculate its dM using few operations using the token supply M from the last period. So, the amount of mined tokens for the validator pool p in the time t will be:

$$\delta m_{v,t} = \frac{s_v}{S}\frac{\ln 2}{T_{1/2}}\left(M_{\max} - M_{t-1}\right), \tag{14}$$

$$dM_t = \sum_v dm_{v,t}, \tag{15}$$

where s_v is the number of tokens bound to the validator's delegate pool v and S is the total number of tokens locked. Instead of calculating all the sum over v, each validator can add their portion $\delta m_{v,t}$.

The distribution factor for a delegate bound to pool p is:

$$\kappa = \frac{1}{2}\left(\frac{\gamma}{\gamma_v} + \frac{s}{S_v}\right), \tag{16}$$

γ_v is the aggregate stake compensation factor for the pool and S_v is the sum of all tokens bound to the pool.

3.5 Mining strategies and expected compensation

In this section, we look at three possibilities: a staker liquidating all the compensation while extending the lock time (Liquidate mining compensation), a staker adding all the compensation to their current stake, and a miner waiting for their stake to unlock after time T. Each of these possibilities could have different distributions of γ. Let's consider $\gamma = 1$ and $\gamma = 0.5$ as the two extreme values of γ. Let's take the amount of tokens locked to be $\lambda = 60\%$, as in DASH.

3.5.1 Liquidate Mining Compensation

In this scenario, all stakers in the pool are liquidating all their earnings every T_{comp} period. The total amount of tokens staked in the network can be expressed as $S = \lambda M$. Assume all the delegators have equal amounts of stake bound to the pool. The amount of stake stays constant in this case, and equal to $m_i = s$, making $m_v = s_v$ and $\gamma = \overline{\gamma_v}$ where, $\overline{\gamma_v}$ is the mean staking parameter of the pool. Then, the pool mining rate (i.e. the cumulative pool reward) is:

$$\frac{dr_p}{dt} = \overline{\gamma_v}\,\frac{l}{\lambda M(t)}\,\frac{\ln 2}{T_{1/2}}\left(M_{\max} - M(t)\right). \tag{17}$$

When we substitute $M(t)$ from eq. 11 and integrate over time, we find total pool compensation:

$$r_p(t) = l\frac{\gamma}{\gamma^*\lambda}\ln\frac{M(t)}{M_0}, \tag{18}$$

If $\Delta r_p(t) = r_p(t) - \mathcal{C}$ where $\mathcal{C}$ is validator's commission, that brings individual staker's compensation to be:

$$r(t) = \kappa \cdot \Delta r_p(t). \tag{19}$$

If $\gamma = 1$ (staking for 1 *year*) and $\lambda = 60\%$ (60% of all AKT are staked). With $\mathcal{C} = 0.1 \cdot r(t)$, staker compensation in AKT starts from 0.45% *per day*, or 101.6% during the first year of staking.

We should note that if other miners stake for less than a year ($\gamma^* < 1$), the inflation rate decays slower, and the compensation over a given period will be higher.

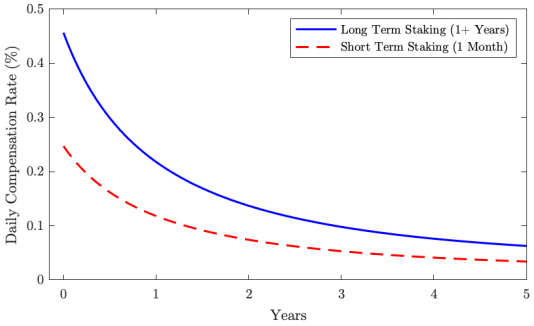

Figure 3: Daily compensation over time assuming 60% tokens locked for lock times of 1 year and 1 month

3.5.2 Re-stake mining compensation

Instead of liquidating mining compensation, it could be re-staked into the pool in order to increase the delegator's stake. In this case, the actual stake s is constantly increasing with time:

$$\frac{ds}{dt} = \gamma \frac{s}{\lambda M(t)} \frac{\ln 2}{T_{1/2}} \left(M_{\max} - M(t)\right). \quad (20)$$

If we substitute $S(t)$ from eq. 11 and solve this differential equation against s, we get:

$$s(t) = s(0) \left[\frac{M(t)}{M_0}\right]^{\frac{\gamma}{\gamma^* \lambda}}. \quad (21)$$

Assuming the validator commission is 1%, if $\gamma = 1$ (staking for 1 *year+*) and $\lambda = 60\%$ (60% of all nodes in the network are staking), delegate compensation in AKT starts from 0.45% *per day*, or $s(1) - s(0) = 176.5\%$ during the first year of staking.

3.5.3 Take mining compensation and spindown

When the node spins down, the staker doesn't extend the time for end of staking t_1, and the compensation is constantly decreasing as the time left to unlock becomes smaller and smaller, effectively decreasing γ gradually towards 0.5. That's the default behavior. To avoid that, the staker should set t_1 large enough, or increase t_1 periodically.

3.5.4 FAQ

How many tokens will ever be in existence? We'll start with 1 *billion* tokens, and the maximum amount of tokens ever created will be 3.89 *billion*.

What's the inflation rate? The inflation rate will depend on how many short-term miners and long-term miners are working in the system. Depending on this, the initial inflation will be between 50% *APR* (if all miners are very short term) and 100% *APR* (if all miners commit for a long term). The inflation will decay exponentially every day, halving sometime between 2 *years*(if all the miners are long term) and 4 *years* (if all the miners are short term) as illustrated in fig. 4

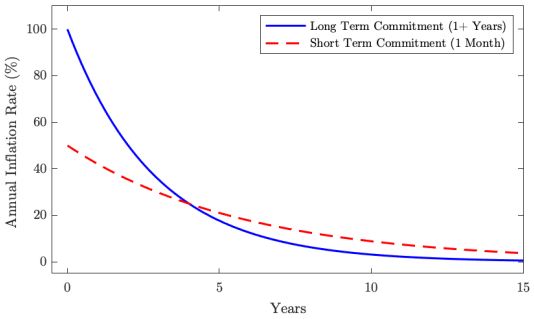

Figure 4: Annual inflation over the years when tokens are locked with long and short commitments

Aggarwal, Sankalp. n.d. "Cosmos Multi-Token Proof of Stake Token Model." https://github.com/cosmos/cosmos/blob/master/Cosmos_Token_Model.pdf.

Buchman, Ethan, Jae Kwon, and Zarko Milsosevic. n.d. "The Latest Gossip on BFT Consensus." https://arxiv.org/abs/1807.04938.

Duffield, Evan, and Daniel Diaz. n.d. "Dash: A Payments-Focussed Cryptocurrency." https://github.com/dashpay/dash/wiki/Whitepaper.

Egorov, Michael, MacLane Wilkinson, and. n.d. "NuCypher: Mining & Staking Economics." https://www.nucypher.com/static/whitepapers/mining-paper.pdf.

"NEO Whitepaper." n.d. http://docs.neo.org/docs/en-us/basic/whitepaper.html.

Satoshi, Nakamoto. n.d. "Bitcoin: A Peer-to-Peer Electronic Cash System." https://bitcoin.org/bitcoin.pdf.

Wood, Gavin. n.d. "Ethereum: A Secure Decentralised Generalised Transaction Ledger." https://gavwood.com/paper.pdf.

"ZCash Emmission Rate." n.d. https://z.cash/technology/.