AKT: Akash Network, Token & Mining Economics

Greg Osuri*
Akash Network
(Dated: November 11, 2019)

Akash is a marketplace for cloud compute resources which is designed to reduce waste, thereby cutting costs for consumers and increasing revenue for providers. This paper covers the economics of the Akash Network and introduces the Akash Token (AKT). We describe an economic incentive structure designed to drive adoption and ensure the economic security of the Akash ecosystem. We propose an inflationary mechanism to achieve economic goals. We provide calculations for mining rewards and inflation rates. We also present mechanisms for allowing a multitude of fee tokens.

ACKNOWLEDGMENTS

We thank Morgan Thomas (*Co-Founder, Kassir*), and Brandon Goldman *(Frm. Lead Architect, Blockfolio)* for providing valuable comments that significantly improved the manuscript.

I. INTRODUCTION

Akash is a permissionless marketplace for trading compute cycles. It aims to create efficiencies in the cloud hosting market through algorithms for allocating compute resources which go to waste in the current market. In this paper, we present the economics of the Akash Network by introducing the Akash Token (AKT) model, which is designed to a) maintain ecosystem sovereignty, b) provide economic security, and c) encourage early adoption by offering *10x lower* cost than current market solutions. Here are some definitions:

Akash Token (AKT): AKT is the native token of the Akash Network. The core utility of AKT acts 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 and service 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

* greg@akash.network

to the hourly income earned; hence, every provider is a delegator and/or a validator.

Tenant: Tenants lease computing cycles offered by providers for a market-driven price (set using a reverse auction process described in below section).

A. Marketplace Overview

A unit of computing (*CPU, Memory, Disk*) is leased as a container on Akash. A container [1] is a standard unit of software that packages up code and all its dependencies, so the application runs quickly and reliably from one computing environment to another. A container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries, and settings.

Any one with a physical machine (ie, computer, server) can slice the machine's resources into containers using a process called virtualization. Docker is a company that provides widely adopted container virtualization technology, and it is common to refer to containers as "docker images." The relation between a physical computer and a container is illustrated in fig. 1).

All marketplace transactions are 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 are 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. 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.

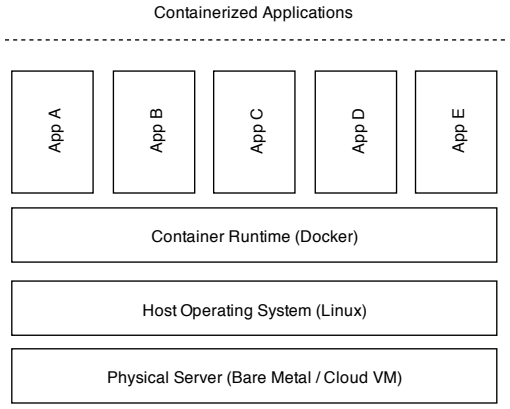

Figure 1: A simple illustration of containerized applications in relation to the physical servers

II. PROVIDER ECONOMICS

In this section, we present economics for a provider offering computing services on Akash.

First, let us examine the current pricing model on popular cloud infrastructure providers, Amazon Web Services (AWS), Google Compute Platform (GCP), and Microsoft Azure. Most providers now offer virtualized computing packaged as *containers*.

The unit cost for a container at a given time t can be calculated using:

$$UnitCost(t) = \frac{ServerCost(t)}{UnitYield \times Utilization(t)} \quad (1)$$

where $ServerCost$ is the capital and operational expenditure for the server and $Utilization$ is the amount of resources consumed (without the operating overheard).

As of the writing of this paper, the current prices to lease a *container-unit* (1 GB Memory and 1 CPU) for an hour are as follows:

- AWS[1] : 0.045
- GCP[2] : 0.036
- Azure[3] : 0.054

Lets us now determine a provider's cost-benefit to offer containers by considering two possible scenarios a) provider wants to recover capital by offering excess capacity (at-cost) b) provider wants to offer a bare-metal cloud instance for profit (at-margin)

A. At Cost: Capital Recovery on Server Cabinet

In this scenario, consider $Provider_A$ has a fleet of underutilized servers and intends to recover deployed capital and subsidize operating expenditure. These servers (also referred to as "rack units") are enclosed in a *cabinet* hosted at a datacenter. A server cabinet (also referred to as a "rack") is a standardized enclosure for mounting multiple servers. Datacenter providers typically charge by cabinet for electricity, real estate, and high-performance internet connectivity.

A typical cabinet costs around USD *559,176* in capital expenditure and USD *1,800* in operating expenditure as illustrated in fig. 2 with a shelf life (time for performance deprecation to sub-optimal levels) of *3 years*, the provider needs to earn: $559,176 + 1800 \times 3 \times 12 = 688,776$ USD (or 229,592 USD / Year) for full capital recovery.

Considering a unit yield of *3,192 per Cabinet*, at 100% utilization, the provider can offer a container for no less than *0.013* USD and, at 40% utilization, the price will be *0.034* USD.

Cabinet Cost			
Cabinet CapEx			
Component	**Rack Size (U)**	**Count**	**Cost**
Router	1	1	$599.99
Switch	1	1	$1,225.00
PSU	1	1	$599.99
Servers (C)	38	19	$555,801.68
Servers (M)	0	0	$0.00
Power			
30 amps, 208 volts, single phase		1	$950
CapEx			***$559,176.66***
OpEx (Monthly)			
42U 15amp 120V Cabinet		1	$600.00
30 amps, 208 volts, single phase		1	$1,200.00
OpEx / Month			***$1,800.00***

Other Components		
Component	**Type**	**Cost (Retail)**
APC Metered Rack PSU	PSU	$599.99
CISCO WS-C3560X-48P-S Switch	Switch	$4,787.12
Cisco ISR 4321 Integrated Services Router	Router	$1,225.00

Figure 2: Server cabinet specification with capital and operating expenditures for a unit that can enclose 19 cloud-grade servers illustrated in fig. 3.

B. At Margin: Mid Tier Cloud Provider

In this scenario, $Provider_B$ leases a bare-metal server from a cloud provider such as Packet and intends to offer containers for a

[1] Amazon Web Services on-demand pricing for ECS (Fargate) in US East (https://aws.amazon.com/fargate/pricing)
[2] Google Compute Engine on-demand pricing in US Central (https://cloud.google.com/compute/all-pricing)
[3] Microsoft Azure on-demand pricing in US Central (https://azure.microsoft.com/en-us/pricing/details/container-instances)

Property	Units
Model	C1
Cores (Hyper Threaded) / Processor	28
vCPU	168
Memory	320
Disk (GB)	800
Size (U)	2

Component	Type	Units
PowerEdge R830 Server	Base	1
Intel Xeon E5-4650 v4 2.2GHz,35M Cache,9.6 GT/s QPI,Turbo,HT,14C/28T (105W) Max Mem 2400 MHz	Processor	4
16GB RDIMM, 2400MT/s, Dual Rank, x8 Data Width	Memory	20
400GB Solid State Drive SAS Write Intensive MLC 12Gbps 2.5in Hot-plug Drive, PX05SM	Hard Drive	2
NEMA 6-15P to C13 Wall Plug, 250 Volt, 13 AMP, 6 Feet (1.8m), Power Cord, North America	Power Chord	2
Dual, Hot-plug, Redundant Power Supply (1+1), 1600W, 250 Volt Power Cord Recommended	Power Suppy	1
Fresh Air Cooling	Cooling	1
Cost		***$29,252.72***

Figure 3: Retail pricing and specification for a high performance, cloud-grade server.

profit. In the case of $Provider_B$, a bare-metal server, such as `c2.medium.x86` on Packet with 30 CPUs and 64GB Memory can yield $\approx$ 30 container units and costs 1 USD/Hr, for which the *price per unit-hr* at 100% utilization will be $1/(30 \times 1) \approx 0.033$ USD and at a more realistic 40% utilization to be $1/(30 \times 0.4) \approx 0.83$ USD respectively.

At 100% utilization, the cost savings are marginal ($< 2x$), which is not a very compelling reason for a user to switch to Packet from AWS. Additionally, when adjusted for utilization risk, Packet needs to sell the container for close to 2x the AWS price point to break even.

The utilization risk combined with a low-value proposition is a reason why providers like Packet refrain from offering container service to their customers.

C. Akash Advantage

By capitalizing on AKT's staking incentives along with employing *re-stake mining strategy* described in sec. VI E 2, both $Provider_A$ and $Provider_B$ can offer to compute at/up to ***10x*** lower cost than AWS while earning a profit as illustrated in fig. 4

				Utilization			
Provider	Stake Income (re-staking 1 yr cost for 3 yrs)			100%		40%	
	Stake	Income (3 Yr)	Cost (3 Yr)	*Unit Price*	*Savings*	*Unit Price*	*Savings*
Provider A		790,294	623,917	0.007	83%	0.019	59%
Akash	*207,972*	*Net Income*	*166,377*	*0.0045*	*90%*	*0.0045*	*90%*
Provider B		33,288	26,280	0.033	26%	0.083	-85%
Akash	*8760*	*Net Income*	*7,008*	*0.0045*	*90%*	*0.0045*	*90%*

Figure 4: Staked income for providers over three years when re-staking on the initial stake for three years

By staking a year's cost, and by re-staking the mining compensation for three years, $Provider_A$ will earn 166,377 USD, and $Provider_B$ will earn 7,008 USD in net profit respectively.

There is however an implicit limit on earnings that is imposed by the market value of AKT.

III. 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 [2] algorithm for Byzantine fault-tolerant consensus. Tendermint was designed to address the speed, scalability, and environ-

mental 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. (For more information on these, please review Akash Network Technical White paper).

A. 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 $V_i(0) = V_{i,0} = 128$ and the number of validators at time t *year* will be:

$$V_n(t) = |\log_2(t+1) \cdot V_{i,0}| \tag{2}$$

So, in 10 *years*, there will be $V_n(10) = 442$ validators as illustrated in fig. 5

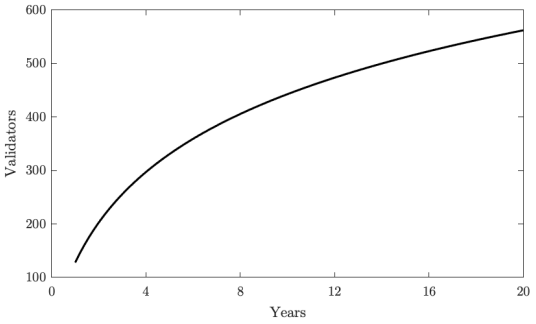

Figure 5: Number of validators over the years

IV. AKT: THE AKASH NETWORK TOKEN

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. That said, AKT performs three main functions: Resolve, Reward, and Reserve.

A. Resolve

Akash relies on a blockchain where a set of validators vote on proposals. Each proposal is weighed by the proposer's voting power, which is the total tokens they staked and the tokens bonded to them (stakers can delegate voting power to validators).

B. Reward

Providers on Akash stake tokens to subsidize operating and capital expenditures. Stakers are rewarded proportional to the number of tokens staked, the length of lockup time, and the overall tokens staked in the system. Lock up times can vary anywhere from one month to one year. Flexibility in lockup encourages stakers that stake for shorter periods (bear markets), in a self-adjusting inflationary system that is designed to optimize for lower price pressure during bear markets.

C. Reserve

Fees on Akash can be settled using a multitude of currencies, however, the market order book uses Akash Token (AKT) as the reserve currency of the ecosystem. AKT provides a novel settlement 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. In this section, we also present a mechanism "Transaction Ordering using Consensus Weighted Median" to establish exchange rates without the need for an oracle.

V. SETTLEMENT

A. 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.

B. 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 [3], Bitcoin [4], and Neo [5], 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).

C. 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* [6] 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.

D. 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 stakeholder of the lease for a given time t is defined by:

$$\kappa_l = \frac{\sum_{t=1}^{T_a} l(1-t)}{\sum_{t=1}^{T_a} L(1-t)}, \tag{3}$$

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:

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

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

$$R_{tk}(l) = \frac{10^{1-\frac{7\mathcal{P}_l}{2}}}{\log\left(10^{\frac{7\mathcal{P}_l}{2}}\right)} \cdot \log\left(\frac{\mathcal{P}_l}{\min(\kappa_l, \mathcal{P}_l)}\right). \tag{5}$$

VI. 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 exponentially 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.

A. 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.

B. 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.

C. General Inflation Properties

1. Initial Inflation

If we assume Akash will have the same number of tokens locked as NuCypher [7] and DASH [8]: $\lambda = 60\%$, then $1 - 40\%$ of the supply of AKT will be in circulation. The adjusted inflation rate for inflation, I will be:

$$I^* = \frac{I}{1-\lambda}, \tag{6}$$

Considering that ZCash [9] 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).

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:

$$I(t) = I_0 \cdot 2^{-\frac{t}{T_{1/2}}} = I_0 \exp\left[-\ln 2 \frac{t}{T_{1/2}}\right], \tag{7}$$

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

$$M(t) = M_0 + \int_0^t I(t)\, dt = M_0 + \frac{I_0 T_{1/2}}{\ln 2}\left[1 - 2^{-\frac{t}{T_{1/2}}}\right] \tag{8}$$

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. 6):

$$M_{\max} = M(\infty) = M_0\left(1 + \frac{i_0 T_{1/2}}{\ln 2}\right) \approx 3.89\, M_0, \tag{9}$$

where M_0 is initial number of tokens.

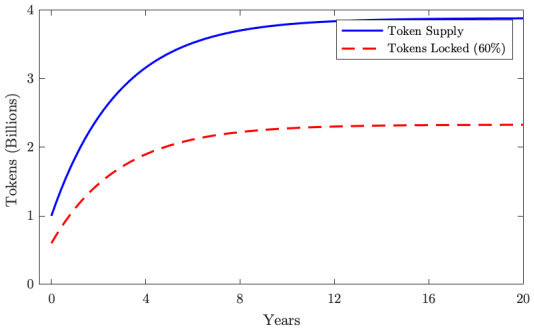

Figure 6: Token supply and tokens locked over years with an initial inflation of 100% APR that is halving every 2 years

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,

$$\gamma = \left(0.5 + 0.5\frac{\min(T_i, T_1)}{T_1}\right), \qquad (10)$$

$$T_{i,\text{initial}} > T_{\min}, \qquad (11)$$

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 that miners will most likely stake for short periods during a bear market, we can expect token emission to decline during a bear market, which will help to boost the price. Therefore we can expect this mechanism to support price stability.

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. 8) at $\gamma^* \neq 1$ will then look like:

$$M(t) = M_0\left[1 + \frac{i_0\gamma^* T^*_{1/2}}{\ln 2}\left(1 - 2^{-\frac{t}{T^*_{1/2}}}\right)\right]. \qquad (12)$$

D. Delegate Pool Distribution

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

$$I(t) = \frac{\ln 2}{T_{1/2}}\left(M_{\max} - M(t)\right) \qquad (13)$$

$$dM = I(t)\,dt. \qquad (14)$$

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. The amount of mined tokens for the validator pool p in the time t can be calculated according to the formula:

$$\delta m_{v,t} = \frac{s_v}{S}\frac{\ln 2}{T_{1/2}}\delta M(t), \tag{15}$$

$$\delta M_t = \sum_v \delta m_{v,t}, \tag{16}$$

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 v is:

$$\kappa = \frac{1}{2}\left(\frac{\gamma}{\gamma_v} + \frac{s}{S_v}\right), \tag{17}$$

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

E. 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.

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:

$$\frac{dr_v}{dt} = \overline{\gamma_v}\frac{S_v}{\lambda M(t)}\frac{\ln 2}{T_{1/2}}\left(M_{\max} - M(t)\right). \tag{18}$$

When we substitute $M(t)$ from eq. 12 and integrate over time, we find total pool compensation:

$$r_v(t) = S_v\frac{\overline{\gamma}}{\gamma^*\lambda}\ln\frac{M(t)}{M_0}, \tag{19}$$

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

$$r(t) = \kappa \cdot \Delta r_v(t) = \frac{1}{2}\left(\frac{\gamma}{\gamma_v} + \frac{s}{S_v}\right)\cdot \Delta r_v(t) \tag{20}$$

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.

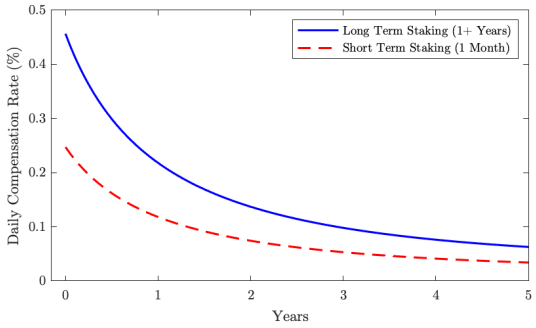

Figure 7: Daily compensation over time assuming 60% tokens locked for lock times of 1 year and 1 month

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:

$$\frac{ds}{dt} = \gamma \frac{s}{\lambda M(t)} \frac{\ln 2}{T_{1/2}} \left(M_{\max} - M(t)\right). \quad (21)$$

If we substitute $S(t)$ from eq. 12 and solve this differential equation against s, we get:

$$s(t) = s(0) \left[\frac{M(t)}{M_0}\right]^{\frac{\gamma}{\gamma^* \lambda}}. \quad (22)$$

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. *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.

4. *FAQ*

How many tokens will ever be in existence? We'll start with 100 million tokens, and the maximum amount of tokens ever created will be 389 million, as illustrated in fig. 6

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). fig. 8

VII. RELATED WORK

The majority of *proof of stake* network such as Ethereum 2.0 [10], Tezos [11], and Cardano [12] all use a single token model. However, there seem to be networks that are experimenting with more novel models. In this section, we will review some of these systems and explore the differences with Akash's token model.

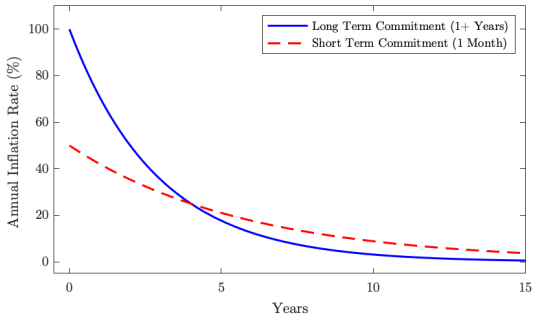

Figure 8: Annual inflation over the years when tokens are locked with long and short commitments

A. Cosmos Hub

Akash and Cosmos Hub use Tendermint [2] Consensus Algorithm and share a core set of values with interoperability and user experience. Similar to Cosmos's Atom [6], AKT's primary utility is to provide economic security to the network. However, Akash's model variously improves the Cosmos' model. Firstly, AKT provides a mechanism to normalize compute prices for the marketplace auction. Secondly, Akash introduces a mechanism to lock in an exchange rate to a reserve currency of choice to mitigate market drive volatility risk of AKT when leasing computing for more extended periods. Finally, Akash's block reward distribution is proportional to the time and amount of a stake, unlike Cosmos' model where the distribution is homogeneous for a fixed time. Cosmos imposes a 21-day "unbonding" period – considered lock up – and there is no incentive to commit for more extended periods. Whereas, stakers in Akash can choose to commit for one month to a year, for which they will receive ~54% and 100% compensation respectively.

B. NEO

According to NEO's white paper [5]:

> NEO network has two tokens. NEO representing the right to manage NEO blockchain and GAS representing the right to use the NEO Blockchain.

At the surface, NEO's primary utility is a staking token and GAS is the fee token. However, after closer observation, NEO's model is very different from Akash's model.

Firstly, NEO is used as a mechanism to determine how many votes each NEO account gets without a requirement to stake tokens. Each account can vote for as many validator candidates as they wish and each validator candidate they vote for receives the number of votes equivalent to the amount of NEO in the voter's account.

With regards to the fee, NEO's chain only supports a single fee token, unlike Akash's multi-token model. Furthermore, unlike Akash, NEO does not provide volatility protection for the GAS tokens.

C. EOS

EOS's *delegated proof of stake consensus* [13] has similarities with Akash's model but are extensively different. In EOS, each token holder can stake their tokens in order to vote for block producer and in return, rewarded in resource units such as CPU, RAM, and NET that can be spent for transactions on the network. However, like in NEO, the staking token EOS is not staked by the block producers, and it is not slashable in the case of misbehavior.

In EOS, staking means, stakers are putting tokens in a lock-up period and not necessarily contributing to the functionality of the network. Stakers earn rewards in CPU, RAM, and NET that are used to purchase computational resources on the network. These resources are not transferrable. CPU and NET are only spendable by the receiver, whereas RAM can be traded with other users in a Bancor-style marketplace [14].

EOS burns these resources upon spending, instead of giving them to block producers. The validator compensation model is unclear, considering transaction fees is not the primary mechanism. EOS is seemingly a single token network, despite having nuances and additional steps.

VIII. CONCLUSION

This paper explains the network and mining economics of Akash Network and presents various incentives and utilities of different tokens in the staking and fees mechanisms. The Akash Token (AKT) acts as staking token and reserve currency for the network while using a multitude of tokens for settlement.

---

[1] "What is a Container?" [Online]. Available: https://www.docker.com/resources/what-container

[2] E. Buchman, J. Kwon, and Z. Milsosevic, "The latest gossip on BFT consensus" [Online]. Available: https://arxiv.org/abs/1807.04938

[3] G. Wood, "Ethereum: A Secure Decentralised Generalised Transaction Ledger." [Online]. Available: https://gavwood.com/paper.pdf

[4] N. Satoshi, "Bitcoin: A Peer-to-Peer Electronic Cash System." [Online]. Available: https://bitcoin.org/bitcoin.pdf

[5] "NEO Whitepaper." [Online]. Available: http://docs.neo.org/docs/en-us/basic/whitepaper.html

[6] S. Aggarwal, "Cosmos Multi-Token Proof of Stake Token Model" [Online]. Available: https://github.com/cosmos/cosmos/blob/master/Cosmos_Token_Model.pdf

[7] M. Egorov, M. Wilkinson, and, "NuCypher: Mining & Staking Economics" [Online]. Available: https://www.nucypher.com/static/whitepapers/mining-paper.pdf

[8] E. Duffield and D. Diaz, "Dash: A Payments-Focussed Cryptocurrency." [Online]. Available: https://github.com/dashpay/dash/wiki/Whitepaper

[9] "ZCash Emmission Rate." [Online]. Available: https://z.cash/technology/

[10] "Ethereum 2.0 White Paper." [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper

[11] L. M. Goodman, "Tezos: a self-amending crypto-ledger." [Online]. Available: https://tezos.com/static/white_paper-2dc8c02267a8fb86bd67a108199441bf.pdf

[12] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A Provably Secure Proof of Stake Blockchain Protocol." [Online]. Available: https://iohk.io/research/papers/#ouroboros-a-provably-secure-proof-of-stake-blockchain-protocol

[13] D. Larimer, "EOS: Technical Whitepaper." [Online]. Available: https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md

[14] "EOS RAM 101: Non-Technical Guidebook for Beginners." [Online]. Available:

https://medium.com/coinmonks/eos-ram-101-non-technical-guidebook-for-beginners-6f971322042e