The Wealth of Crypto Networks

First Economic Model Proposal for MultiversX Network

MultiversX Team - v0.3.2

Last update: 7 Oct. 2020

Disclaimer

Nothing in this paper or elrond.com website is an offer to sell, or the solicitation of an offer to buy, any tokens. MultiversX is publishing this paper solely to receive feedback and comments from the public. Nothing in this paper should be treated or read as a guarantee or promise of how MultiversX's business, services or the token will develop or of the utility or value of the token.

This paper and elrond.com website outlines current plans, which could change at its discretion, and the success of which will depend on many factors outside MultiversX's control, including market-based factors and factors within the data and cryptocurrency industries, among others. Any statements about future events are based solely on MultiversX's analysis of the issues described in this paper or elrond.com website. That analysis may prove to be incorrect.

MultiversX eGold (eGLD) incorporates no connection to physical gold or gold derivative instruments. eGLD is not a "stablecoin" and may be volatile and/or may lose value. No recommendation is made herein as to the advisability of purchasing eGLD; notwithstanding, do not purchase eGLD if you cannot bear the loss of the entire purchase price.

Preface

Why is capitalism under stress now? Why are the central banks looking so fragile? Is it possible to move beyond capitalism and find a better approach? Can capitalism short itself? [3] Can we build a more robust, or even antifragile alternative, where the "too big to fail" systems are no longer present? [21]

Through MultiversX we propose a bold vision for a post-capitalist world, providing a new economic model and language specifically designed for the information age.

This paper outlines how the native currency of the MultiversX public blockchain, will be created and algorithmically minted, to maintain the incentives aligned with the long term health and security of the network. This paper offers a temporary snapshot of the economic principles governing the MultiversX Network, as they stand at the time of writing.

The MultiversX token, eGold (eGLD), will have an expected bootstrapping duration of roughly three-to-five years. The MultiversX token is inseparable from the MultiversX Network, and thus intrinsic to it. Some of eGold's intended use cases include staking, delegation, payments, fees for storage rent and for smart contracts deployment, as well as rewarding the validators that contribute to the Network's performance, stability and security.

During the first few years, our focus will be to establish MultiversX as a global public utility within the internet ecosystem, offering a highly scalable, efficient, and interoperable blockchain architecture, with a growing economy built on its native eGold tokens. All activities within the network, such as processing transactions, running smart contracts, providing services like staking or running a validator node will be fueled by our native token. Both startups and large scale enterprises will be able to build decentralized applications on top of MultiversX's Network or to integrate MultiversX as part of their infrastructure solution for products and services.

In this first phase, gaining access to a recurring value stream generated by the network is conditioned by owning the eGold token, as the native asset of the MultiversX Network.

Following this first period, we expect that eGold will naturally temporarily lend itself to becoming a currency or payment token as well, complementing conventional currencies thanks to its flexible programmatic mechanism. This means that eGold will likely become an efficient medium of exchange for various goods and services, since its owners will be able to send and receive eGold directly, globally, and inexpensively via transactions.

Once MultiversX becomes a thriving global ecosystem and public utility, one might expect the token to become a robust store of value, owing to compounding programmable incentives and strong underlying network effects governing blockchain architectures. Its quality as a store of value will be a function of the underlying economic incentives amplified via real world adoption, defined conditional transition to a deflationary economic model, and accrued trust in the MultiversX Network.

The MultiversX Network, on the other hand, is a proof-of-stake based blockchain platform where a set of validators, who have staked eGold, produce blocks by reaching consensus. Validators are rewarded for their work and staked eGold. However, if a validator decides to intentionally depart from protocol instructions, they stand to lose part of their staked eGold due to slashing. The set of nodes elected as validators and their assignment to shards changes constantly (in each epoch, i.e. around once a day, based on an auction process that will be activated after the launch of Mainnet), and this number is limited depending on the current needs of the network in terms of security and throughput.

Any number of eGold holders can participate in staking indirectly by delegating their eGold to existing validators, usually professional validators (staking-as-a-service providers), that choose to accept delegations. An eGold holder indicates which validator candidates they trust, and puts some eGold at stake to support their delegation. If one or more of their candidates are elected as validators in an epoch, they will share with them any economic rewards or punishments, proportional to their delegated stake. Delegating eGold is a way of investing one's eGold, and contributing to the security of the system. The larger the total amount of eGold staked, the higher the system security, thanks to the increasing amount of stake needed by an adversary to get any nodes elected as validators.

We therefore aim to have above 50% of the circulating supply staked at all times.

How to contribute and give feedback

This paper is the first public draft of the MultiversX economic model. The individuals and companies contributing to this paper operate in a dynamic environment where new ideas and risk factors emerge continually. Thus, we are constantly looking for feedback, with new assumptions that could challenge and improve parts of our model. We encourage those who want to contribute, to provide their feedback on the MultiversX's forum**.**

Table of contents

1. Context

About 70,000 years ago, early modern humans went through a significant evolutionary leap known as the Cognitive Revolution. It is this revolution that enabled Homo Sapiens to develop uniquely sophisticated thinking and communication skills, which perhaps surprisingly, lead them to become the dominant and most fearsome predator on earth. [1]

The development of language was, undoubtedly, one of the most instrumental factors in Homo Sapiens' ascent. Language helped create a common understanding between members of a group, facilitating communication and exchange of information and ideas. Consequently, trust, cooperation and coordination emerged as increasingly useful and necessary tools, scaling primitive communities for the first time. Thus, tribes created villages, which later transformed to cities, and then some transformed to nation states. Today nation states have gradually been superseded by the modern hyper-connected global village.

Humans are social animals, and throughout history we've always lived in communities. At first, trust and transfers were more social, personal and direct. Then, it transitioned to institutional, impersonal and indirect (think middlemen) both at local and global levels. But for each transition, new tools and structures had to be invented and applied, as old ones showed their limitations with each change in scale.

The dawn of the technological revolution announced a rapid increase in the rate of progress. Hardware, especially transistors and microprocessors, became the first major landpost, with Moore's law underscoring an exponential trend we would experience. Software, mostly proprietary in the beginning, captured our imaginations next, as it became clear that it was eating the world. The open source movement took software to the next level, creating novel tools and standards that could scale collaboration globally.

Furthermore it is clear now that we are on the cusp of a major paradigm shift with respect to data and privacy. As people wake up to discover its surprising value and utility, we expect novel tools will soon enable everyone to collect, manage, and monetize their private data as they wish. New laws will grant us ownership to what should have been inalienable rights, and this will mark the transition from data feudalism, to open data markets, enabling productive exchange built on sovereign data ownership.

If the Internet was all about collaboration and digitalization of content, the next major technology wave will have to bring novel coordination, and economics mechanisms that can scale globally, enforce digital ownership of data and goods, and offer a working model to govern all of this. This is precisely where MultiversX comes in.

1.1 Programmable money

Humans invented money and writing to facilitate the exchange of value and information. Thus, it became easier to conduct economic transactions, and potentially harder to commit economic fraud. Economic exchange enabled communities to grow, but as they grew, it became increasingly difficult to coordinate them. Thus we developed laws to regulate behavior, and institutions to ensure compliance with them.

Today, new technologies have emerged that enable us to scale economic exchange at a global level in a different and far better way. Cryptographically secured decentralized networks have introduced a form of programmable money with particularly valuable properties that are gaining considerable traction.

Most important properties among those are:

- Capital asset
- Medium of exchange: unstoppable, cheap and fast
- Store of value: seizure-resistant and censorship resistant, non-sovereign
- Privacy: on-demand anonymity, pseudo-anonymity and confidentiality
- Programmable through smart contracts, enabling a range of (decentralized) financial services (DeFi): financial instruments for derivatives, securitization and tokenization of assets, lending, escrow, mortgage, insurance, staking, delegation, collateralization and many others.

Given the above properties we believe that programmable money is a multi-trillion dollar market still in infancy. Programmable money will facilitate better alignment of incentives, and will enable new mechanisms for capturing value, but we should be careful to avoid making the same mistakes done in traditional economies.

Zooming out, MultiversX's impact will aim to go beyond money, gradually enabling data, identity and property to be transformed to digital assets via tokenization.

1.2 Cryptoeconomics

Definitions

Cryptoeconomics can be aptly described as the use of incentives and cryptography in designing distributed networks. It is not a subfield of economics, but rather an area of applied cryptography that takes game theory into account.

Game theory is the mathematical modeling of strategic interaction among rational (and irrational) agents. Mechanism design, on the other hand, is a subfield in game theory, often referred to as reverse game theory, because we start with a desired outcome in mind, and work backwards to design a game promoting it. A game where rational self-interested players will produce a desired outcome.

So, if game theory is about choosing the best moves in a given game, mechanism design is about creating a game which accounts for the moves you desire.

To sum up, cryptoeconomics consists of two components: cryptography which is the part of the mechanism that ensures the integrity of past moves, and economics which is the part of the mechanism that ensures all actors take the proper future moves.

The economic security guarantees of any crypto network depend in part on the strength of its assumptions, about how people react to economic incentives. However, it is worth noting that mechanism design is not a panacea, and cryptoeconomics cannot be applied in a vacuum. There is a limit to how much we can rely on incentives to predictably shape future behaviours.

Creating a model

Several aspects are taken into account when creating a crypto economic model:

- desired behavior of all actors
- economic incentives such as rewards and fees for the well behaving actors, but also penalties for any actor that may have misaligned incentives relative to desired behaviors
- economic rules (like rating, penalties or slashing) that discourage specific behaviours: invalid protocol messages, failure to produce, omission of protocol messages, equivocation and others.

The economic aspects of the incentives being implemented must take into consideration that, regardless of the monetary value of a particular token, there are factors that influence the wellbeing of the system, namely:

- the inflation should be small enough to not "tax" the token holders, but large enough to cover their staking costs (running nodes)
- the monetary supply being staked should be large enough so that there are enough distinct entities that collusion is unlikely, but small enough so that money velocity is not affected (MV=PQ)

As can be seen, cryptoeconomics are the rules of the game, but how does one change the rules after they have been put in motion? The answer is governance**.** Governance is the power to change the rules, and as the game becomes more valuable, governance becomes the metagame that can sustain or destroy that value.

1.3 Governance

Cryptographically secured distributed networks provide a neutral layer of decentralization, immutability, privacy and trust. Smart contracts can thus be used to both run provably fair electronic elections, or to buy them.

Given their significant and far reaching implications designing an effective governance mechanism for decentralized systems is a strenuous task. Mere extrapolations of real-world governance models are proving naive, and many cryptonetworks will likely die due to flawed governance once their network will reach a sufficiently high value for a range of decisive attacks to be warranted.

Thus, governance requires separate, in-depth consideration. The MultiversX governance model will be outlined in a future paper, to be released at a later stage, after the official launch of the MultiversX Network. Prior to that point, MultiversX will use a robust off-chain governance approach to ensure maximal speed and efficiency.

1.4 Terms and Organizational Components

A clear definition is necessary, for the terms used to describe various actors, and actions, inside the MultiversX Economic Model.

Users or Network Participants
Any party, individual, entity, enterprise, blockchain or network that uses, develops, creates or interacts with any aspect of the MultiversX Network. Users are identified by a unique account address (derived from their master public-private key pair stored in a wallet).

Token Holders
Users that are holders of native eGold tokens, to be used on the MultiversX Network to submit signed transactions for value transfers, smart contract execution or to provide liquidity.

Application Developers
Users that develop smart contracts and/or applications that rely on smart contracts to provide services. Developers need an account to deploy smart contracts on the Network.

Consensus group
In order for a block to be proposed and committed, a specific number of nodes (*numNodesConsensus*) are randomly selected from all eligible nodes (*eligibleNodesPerShard)* assigned to a shard to form the consensus group during each round (*blockTime*). The consensus group has the responsibility of committing blocks in that shard, during each round. At the beginning of each round, a new consensus group is selected. The consensus group in the Metachain shard is configured so that *numNodesConsensus = eligibleNodesPerShard*, effectively making the entire Metachain shard the consensus group. This is motivated by the high security requirements of the Metachain.

Nodes
Devices (computers or servers) running the software (the MultiversX client) and relaying messages received from their peers. They can be either Validators (actively participating in securing the network) or Observers (passive members of the network that can act as a read & relay interface) and can be either Full (have the entire history of the blockchain) or Light nodes (only keep 2 epochs

of blockchain history). A node is on the eligible nodes list if several requirements are met: a rating above a specific threshold, won a node slot in the selection auction (when auction will be enabled), assignment to a shard, etc.

Node Type	**Participating**	**Non-Participating (Observers)**
Full	Node that keeps a record of every transaction in the network and also stakes eGold to participate in the consensus mechanism (validator as well)	Node that keeps a record of every transaction ever to occur in its shard. Does not stake and therefore does not propose or sign blocks.
Light	Node that has a stake (so is a validator) and only keeps the records of transactions in the most recent epoch(s)	No stake and only keeps 2 epochs of blockchain history

Validators

Validators are nodes — computers on the MultiversX network that process transactions and secure the network by participating in the consensus mechanism, while earning rewards from the protocol and transaction fees. In order to become part of the MultiversX network, a validator needs to commit a collateral in the form of eGold tokens, which are staked to align the incentives of the validators with the correct functioning of the network. Validators stand to lose some, or all their stake if they deviate from the protocol instructions, or otherwise collude to disrupt the network. In order for a node to be able to become a validator it needs to be on the list of eligible nodes.

Block proposer

The block proposer role is designated to the first selected (through an unbiasedly, random process) validator node in the consensus group. The block proposer is the validator who proposes the next block, which the rest of the consensus group must verify and approve.

Block rewards

The blockchain will reward the validator nodes for their staked eGold. The reward might consist of two types: a part of the transaction fees, and new emission of eGold (also called minting or inflation). MultiversX holders who do not put their eGold at stake by being a validator or delegating their eGold to a validator will not receive any of the block rewards.

Shards

At any given time, the network consists of a number of shards, with each shard containing a subset of all addresses and their associated state, including user account addresses and smart contract addresses. Each shard runs its own blockchain, but all shards are connected through the Metachain.

Metachain
A blockchain running in parallel and synchronously with the shards, used for notarizing the blocks committed by the shards and also for cross-shard communication. All eligible validators in the Metachain participate in its consensus. Instead of choosing a consensus group, the randomness source is used only to choose a block producer. The Metachain block producer composes a Metablock which consists of shard headers info and miniblock headers, each of which must be confirmed by at least one shard block in its relevant shard. The Metachain block proposer also creates the "start-of-epoch" block when needed. Metachain is also responsible for the stake/unstake/unjail (changes in the validator configuration) and slashing.

Protocol Sustainability
The protocol sustainability has the purpose of increasing the security and value of the Network on the short, medium and long term. The specifics of governance and management of protocol treasury will be presented in the governance paper. Until then, the protocol treasury will be under the control and supervision of the MultiversX Core Team.

Protocol Governance Body
A self-organized decentralised autonomous organization, overviewed by a non-profit foundation.

More details about the technical aspects are outlined in the whitepaper (https://elrond.com/assets/files/elrond-whitepaper.pdf) which describes the architecture of the MultiversX protocol in detail.

2. Validators

In order to secure the network, MultiversX will utilize a Proof of Stake model.

Unlike Proof-of-Work (PoW) systems, MultiversX does not require machines to solve any puzzles. Instead, all the computational power of the network is used for actual transactions, hence, with Proof-of-Stake, the energy saving is substantial. Additionally, MultiversX does not require any GPUs or specialized chips in order to support the network: you can contribute to and support the network using the hardware you already have (if it meets the minimum requirements: dual CPU, SSE4 and x64 capable CPU, 4 GB RAM, 80 GB HDD).

In Proof-of-Work (PoW) systems where a miner takes everything (block reward + transaction fees), there is only one way to improve your chances of being successful: increase your hash power. This leads to three outcomes: i) it becomes uneconomical for small/low power devices to participate, ii) mass pooling of resources becomes desirable, and iii) specialisation of hardware becomes necessary.

In contrast, Proof-of-Stake (PoS) does not rely on rewards for securing the network, but rather on penalties. Validators put money ("security deposits") at stake, and are compensated for locking up their capital and incurring costs for maintaining the node. Most of the cost of acting against the rules comes from penalties that are hundreds or thousands of times larger than the rewards an attacker could get in the meantime. So if in PoW the miners are competing with each other, in PoS validators are collaborating with each other.

In this way, a PoS network allows for a much more resource-efficient, scalable, and inclusive way of maintaining a permissionless blockchain network. Looking at the numbers gathered in early PoS networks and the two largest PoW networks (Bitcoin and Ethereum), we can see that the money spent on infrastructure is an order of magnitude smaller in PoS compared to PoW (around 10% of the rewards instead of 100%).

So, in MultiversX there is no mining. Instead, validators earn tokens for doing useful work. One of the most important aspects we had in mind when designing the MultiversX Network was achieving guarantees of fairness for all the network participants. In the case of validators, we designed MultiversX to be resistant to concentration of resources and to ensure an equal and fair distribution of rewards based on the work done by all validators, whether big or small.

Network participants bring value to the network. The more validators, the more eGold staked, and the greater the security and decentralization of the network. Given that sharded networks such as MultiversX require a necessary number of validators to form several well secured shards, we have developed a validator client that runs on average consumer hardware without any requirements for complex setups and lengthy configurations.

2.1 Validators selection

One of the main goals we had in mind when we designed the MultiversX protocol was high scalability. We are achieving this by partitioning the network in shards, which enables parallel processing of blocks. More validators means more shards can be created, so the network can process more transactions, thus is scalable. With this in mind we have to take into consideration that the number of validators and shards should closely match the current needs of the network (including — up to a degree — a sudden increase in usage). Since too many shards means the protocol is under-using the resources and the costs are higher than needed, we should aim that all shards have a load of roughly 50% (*targetShardLoad*).

That's why we are planning a phased launch of the mainnet, where the number of nodes is limited to a specific number (*numNodes*). This limit may increase, both with the phase progression and with the needs of the network, keeping a balance between security, decentralization, efficiency and the expected needs of the network, especially in terms of throughput, data availability and storage.

There will be a limited number of *nodesPerShard*, so *numNodes* will grow proportional with the number of shards that the network requires for processing and storage. Thus, a minimum number of 3 shards (plus metachain) are formed so that the reorganization of shards at the end of epoch makes sense.

There will be a minimum reserve node price, predefined, so that the *nodePrice* cannot go lower than this. The minimum reserve node price can be a fixed amount in eGold or pegged to a fixed amount in USD.

In order to bootstrap the MultiversX Mainnet, at Genesis, we have deployed a closed staking and delegation system. This meant a temporary no-in and no-out for validators or delegators. The bootstrapping process was designed to achieve escape velocity and gather a sufficiently large community around the MultiversX Network. Another goal we had in mind was to create prohibitive economic deterrents against network attacks, ensuring that the larger the supply locked for staking, the larger attack costs for malicious actors.

At Genesis, the Mainnet was bootstrapped with a fixed stake per node, 2500 eGLD, and fixed number of validators: 2169, forming 3 shards and a metachain.

The transition from this bootstrapping period to a sustainable growth model will be done in phases.

- Phase 1 and 2 will enable validators and delegators queues so that the number of nodes remains fixed or above a certain threshold, while allowing new community members to join the queue by delegating or staking their eGLD tokens and reserving a spot in the queue. These queues will also allow existing delegators and validators to withdraw their stake if they wish to, thereby replacing them with the first that have been reserved in the queue.

- Phase 3 and 4 will include features like: increasing the total number of nodes, the possibility of staking more than 2500 eGLD per node, open delegation with a new system delegation Smart Contract through which anyone can receive and accept delegations, and a new and improved (soft) auction system. The transition to Phase 3 and 4 will very likely also include our first on-chain community voting.

2.2 Validators ratings

As with any decentralized and permissionless network, we are expecting to see participation from many validators, from different locations, using different hardware specs, infrastructure setups, internet connections, bandwidth, etc. This will lead to different performances in terms of up-time, response time, computation time, etc. While these variations are acceptable and expected, the more decentralized the network is, the clearer it becomes that specific actions are more desirable, while others actions are not. Keep in mind that when discussing rating, we are referring in general to the up-time and hardware/setup performance (manifested as the amount of blocks successfully proposed and signed), and not behaviour and actions against the protocol, which are covered by the slashing section (double-signing, equivocation, etc.).

Through the rating mechanism, we are rewarding desired performance (such as uptime and correct proposal of a block), but we're also penalizing undesirable actions affecting the performance of the network (such as missing block proposals). The higher the rating of a node, the higher the chance to be selected as a consensus validator in a round (which implies having the opportunity to earn rewards). Conversely, the lower the rating (but above a configured value *ratingThreshold)*, the lower the chance to be selected as a validator. The reward or penalty is performed merely through an increase or decrease of the node rating, so no slashing is involved.

The rating of a node is an integer value between 0 and 100, inclusively. All ratings are stored by the metachain, which tracks the activity of the nodes round by round, and at the end of an epoch, the metachain adjusts the ratings accordingly. Each node joins the network with the same initial *startRating,* which is carried on and adjusted from epoch to epoch.

Table 1 quantitatively presents how the rating of a node increases or decreases its chance to be selected as a consensus validator (subject to change in time when more data is available).

Table 1

Rating interval	**Chance modifier**
0-10	-100%
10-20	-20%
20-30	-15%
30-40	-10%
40-50	-5%
50-60	0%
60-70	+5%
70-80	+10%
80-90	+15%
90-100	+20%

A validator node can increase its rating in two ways:

1) Maintaining a good record of signing proposed blocks. Whenever a node is chosen to be a consensus validator, its rating will be implicitly increased by the value *validatorRatingIncrease*, given a good block signing record.
2) Proposing a valid block when selected to be the block proposer (i.e. consensus leader). A valid block will cause the rating of the block proposer to be increased by the value *proposerRatingIncrease*.

In order for a node to be eligible to receive *validatorRatingIncrease* upon its selection for consensus, it needs to have signed a minimum percentage of blocks out of the last continuous sequence of *numValidatedBlocksRange* it has been a validator for (counting into the past epoch is allowed). The percentage of signed blocks must be equal to or greater than the value of *signedBlocksThreshold*. The reason behind this approach is the fact that for a proposed block to be validated, only ⅔ +1 signatures are needed. We do expect that a node will have its signature present on at least some blocks on a specific, long enough, time frame (or number of blocks), in order to increase its rating for validating. This limit needs to be high enough, so we don't encourage free-riding nodes which don't actually sign blocks, but only propose blocks when they happen to be block proposers. On the other hand, nodes that are consistently slow and which are not able to send their signature for blocks in the required time will, at some point, stop receiving a rating increase for being selected in a consensus group, because their percentage of signed blocks will drop below *signedBlocksThreshold*. Moreover, in this situation, they might start losing rating points (to be implemented at a later stage).

The rating model is thus designed around encouraging productive nodes as much as possible, either as validators or as proposers. A primary design concern is how long does an ideal node need to reach the maximum rating possible, after joining the Network. This duration is named *HoursToMaxRatingFromStartRating*, and it is the expected number of seconds needed by a node to gradually reach the maximum possible rating (*maxRating*) in ideal conditions, starting from the initial rating value, *startRating*. The value of *HoursToMaxRatingFromStartRating* will likely be configured to equal a few days.

Starting from *HoursToMaxRatingFromStartRating*, the model defines the functions *avgValidatorRatingPerRound(·)* and *avgProposerRatingPerRound(·)*, which express the average number of rating points gained by an ideal node, per round, when selected as a validator and as a proposer, respectively. These functions depend on the aforementioned *HoursToMaxRatingFromStartRating*, as well as on the shard topology, consensus group configuration and on the *importanceRatingRatio*, a fixed proportion defined as:

$$importanceRatingRato = \frac{avgProposerRatingPerRound(\cdot)}{avgValidatorRatingPerRound(\cdot)}$$

This proportion balances the amount of rating points gained by validators versus proposers, a necessity given the fact that it is far more likely to be selected as validator in a round, instead of as proposer. For block proposers, it is desired that the overall contribution to rating of the total *proposerRatingIncrease* awarded in one epoch should ideally be the same with the total of awarded *validatorRatingIncrease*. The exact definitions of *avgValidatorRatingPerRound(·)* and *avgProposerRatingPerRound(·)* are currently in development, as they depend on the model chosen for the consensus selection algorithm.

The values for *validatorRatingIncrease* and *proposerRatingIncrease* can be expressed as follows:

$$validatorRatingIncrease = \frac{(maxRating - startRating)}{avgValidatorRatingPerRound(\cdot)}$$

$$proposerRatingIncrease = \frac{(maxRating - startRating)}{avgProposerRatingPerRound(\cdot)}$$

The current intention is to keep both *validatorRatingIncrease* and *proposerRatingIncrease* to constant values. To achieve this, the definitions of *avgValidatorRatingPerRound(·)* and *avgProposerRatingPerRound(·)* must be adjusted accordingly, because they alter the rating of a node, which in turn alters its probability of being selected for consensus. As explained earlier, this then affects the rating, forming a controlled feedback loop. Non-constant alternatives for *validatorRatingIncrease* and *proposerRatingIncrease* are being considered by the team as well.

Apart from having its rating increased, a validator will have its rating decreased if it fails to propose a valid block when selected as a block proposer, regardless of the reason of the failure. Every time a block proposer fails to fulfill its role, its rating will be adjusted with the following penalty:

$$proposerRatingDecrease =- 4 \cdot proposerRatingIncrease$$

A validator that is offline or not able or willing to produce new blocks or sign blocks, will have its rating decrease a lot faster than the rate is increasing. This can be further accelerated if more nodes have their rating below *ratingThreshold.*

It is up to the implementation phase to avoid penalizing honest block proposers after a malicious round took place (1 block before) by delaying the block.

The metachain shard selects its consensus group differently. While the selection of the block proposer is indeed the same as in the rest of the shards, any node that isn’t the block proposer automatically becomes a consensus validator. This happens because the consensus group in the metachain shard is configured to have the size of the entire shard, increasing security. In order to maintain consistency with the other shards, the metachain replaces the definition of *validatorRatingIncrease* with *validatorRatingIncreaseMeta*:

$$validatorRatingIncreaseMeta = \frac{(maxRating - startRating)}{avgValidatorRatingMetaPerRound(\cdot)}$$

Rating awards and penalties for block proposers in the Metachain remain the same as in the shards:

$$proposerRatingIncreaseMeta = \frac{(maxRating - startRating)}{avgProposerRatingPerRound(\cdot)}$$

$$proposerRatingDecrease =- 4 \cdot proposerRatingIncreaseMeta$$

In order to accelerate the elimination of potentially offline nodes, we will implement a penalty that increases with every consecutive failure to propose a block when chosen to be the proposer. This value is the *proposerPenaltyGrowth* (in the Genesis config it is called "consecutiveMissedBlocksPenalty" and can be set differently on shards and meta; default right now is 1.1, 10% increase of proposerRatingDecrease)*,* configured so that a node which constantly fails to propose blocks when selected as a block proposer will have its rating decreased below the *ratingThreshold* in around 10 hours, making it ineligible to participate in the next auction or validator selection process. Furthermore, a node who fails to propose all blocks (when selected as a block proposer), during one epoch, will not be eligible for any rewards during that epoch.

For a node with rating below the *ratingThreshold* to be re-considered to be put back on the list of eligible validators, a special unJail transaction (*resetRating*) has to be sent to the metachain, and validated. In order to incentivize the inclusion of the *resetRating* transaction, the node must include a *resetRatingFee* as part of their transaction, which will be awarded the block proposer that includes it. The current line of thinking is to make the *resetRatingFee* amount to be equal with at least the average rewards earned by a validator in the last epoch. At Genesis this was configured at 0.1% of the nodePrice and there is no actual reset transaction, but through unJail (if only the node is already jailed) can be reseted.

Note that we are not slashing a validator that had its rating drop below *ratingThreshold*. Still, nodes with a rating below *ratingThreshold* are no longer earning rewards and are no longer considered eligible for being part of consensus groups. Moreover, at the end of the epoch, they will also be kicked out automatically from the rollover pool for the next validator selection if the minimum number of nodes per shard has not been reached.

The incentive for a validator to keep a node rating above *ratingThreshold,* is that failing to do so will cost them the opportunity of being part of the validator pool, and that will result in losing rewards for at least 2 epochs. Moreover, keeping a high rating increases the chances of a validator to be selected in a consensus group.

The exact definition of the statistical model for rating is currently being developed, and is strongly connected to the consensus selection algorithm, which, as described earlier, uses the rating of a node to increase or decrease its probability of being selected.

Under the current implementation, the consensus selection algorithm can be modelled using a distribution based on the multivariate central hypergeometric distribution. Alternative algorithms and their implementations are currently being researched, as well.

A simulation, based on some initial assumptions and configured values can be seen here: https://docs.google.com/spreadsheets/d/1DzeeJvLvS5H7XrH24QURyYQJ9QqyaUG5yzUDwyl5yY4/edit#gid=267148288.

2.3 Slashing

Actions taken by validators, such as running other clients or modified code from the official client, can be detrimental to the operation of the network, and so require some punitive measures to be taken in the context of a PoS system. The security of a PoS system is held together through incentives in the form of reward and penalty. By requiring validators to put skin in the game via a locked eGold stake, they will have a strong disincentive to act maliciously due to their economic value being at risk.

In the MultiversX Network, the whole slashing process can be described through a process of triggers that can have different actors:

1. Detection
2. Reporting
3. Verification
4. Effect

The **detection (1)** is done by a node that has access to the block that a malicious validator creates/signs, and can verify the correctness of this block. This can be any node in the shard where the malicious action was made. Since all nodes in the shard are processing all the produced blocks, it means that any node in a shard (validator or observer) can detect the defined adverse actions. This enables for any node that runs the official MultiversX code, to detect and provide proofs for the observed adverse action, verify the validity of provided proofs, and be rewarded in case such a proof was validated. The nodes that detect and provide proof of wrongdoing are called **fishermen** (or challenger).

There will be an additional option switch for validators, to enable or disable the fisherman/challenger role, which will require the configuration of a valid private key associated with a wallet that holds funds.

The **reporting (2)** of an observed malicious action will be made through a special transaction (it could need 2 transactions with a commit reveal scheme in order to prevent front running attacks). The transferred value in such a transaction will be non-trivial, and prohibitive for generating false challenges. The reason for using transactions as challenges is two fold: the mechanism should prevent spamming, and to ensure a reward, given that validating such challenges will require non-trivial bandwidth consumption (data transfers for the evidence) and processing time.

The structure of such a transaction is similar to normal smart contract call and detailed below:

- **sender** – wallet address of the node operator
- **destination** – fixed address for the slashing protocol smart contract
- **gas price** – the gas price
- **gas limit** – the gas limit
- **value** – fixed non-trivial value for any challenge (to be decided later)
- **data** – parameters for the verification SmartContract – which function to call and its parameters. The called function should be the proving function for the observed malicious action, and the parameters the required data for the verification (e.g header data, block data, merkle proofs etc.)

The gas price and gas limit should be fixed for the protocol smart contract to account for the longest path (most complex proof of a scenario).

As mentioned earlier, the reporter of an adverse/malicious situation will be called “Fisherman” - as it fishes for malicious activities, or “Challenger” - as it challenges any adverse situation it finds. The fisherman, as previously mentioned, can either be a validator in the MultiversX Network, or simply an observer node.

This means that the fisherman does not require any stake in the network, but still requires a wallet and sufficient MultiversX tokens in order to issue challenges. In case the challenge is proven valid according to the provided evidence, the transferred value through the challenge transaction is returned to the sender, together with the 50% of the slashed amount from the found malicious actor(s). The rest of 50% of the slashed amount is being considered to be burned in order to deter possible attacks and prevent collusion.

The **verification (3)** of any challenge will be done by the metachain nodes. The challenge is issued through a transaction that transfers an amount of eGold, so it will be executed inside a shard and included in a block. The challenge transactions are also referenced in the block header, so that metachain can do the verification.

The same challenge may come from multiple fishermen in the system at the same time, so there will be a way of identifying the same challenge coming from multiple reporters. This could be done according to the challenge data field which is unique for each different challenge.

There should be no two different types of challenges that can be verified on the same block, if there are, then the most damaging one should be considered for giving back rewards.

The notarization of such a challenge will produce a slashing **effect (4)** for one or more validators if the challenge is indeed validated, depending on the type of adversarial action that was reported: in some cases all signers of an invalid block are slashed, in other cases only the block producer, or a subset of validators from a consensus group. If the challenge is not validated by the Metachain nodes, then the challenger has lost the value transferred through the challenge transaction and the associated gas for validating the challenge.

Once a shard processes a Metachain block that has notarized a validated challenge transaction, the challenger would receive the transferred value back and a percentage (amount to be decided) of the slashed stake(s), while another portion could be given as rewards to the metachain nodes. For the challenges marked invalid by the Metachain, the shard would not need to do anything else, as the cost of the challenge was already transferred.

We define misbehavior or malicious behavior the actions that can be proven cryptographically:

- Double signing a block at the same height

- Signing a block with an invalid post state root (i.e. invalid state transition)

There are two additional approaches that are left for the implementation phase and for future research:

- We might consider gradually increasing the amount slashed, as the time pass and the network becomes more mature
- We might consider increasing the amount slashed based on the number of other validators slashed at the same time, so that we further discourage coordinated actions by multiple malicious actors

2.4 Staking rewards

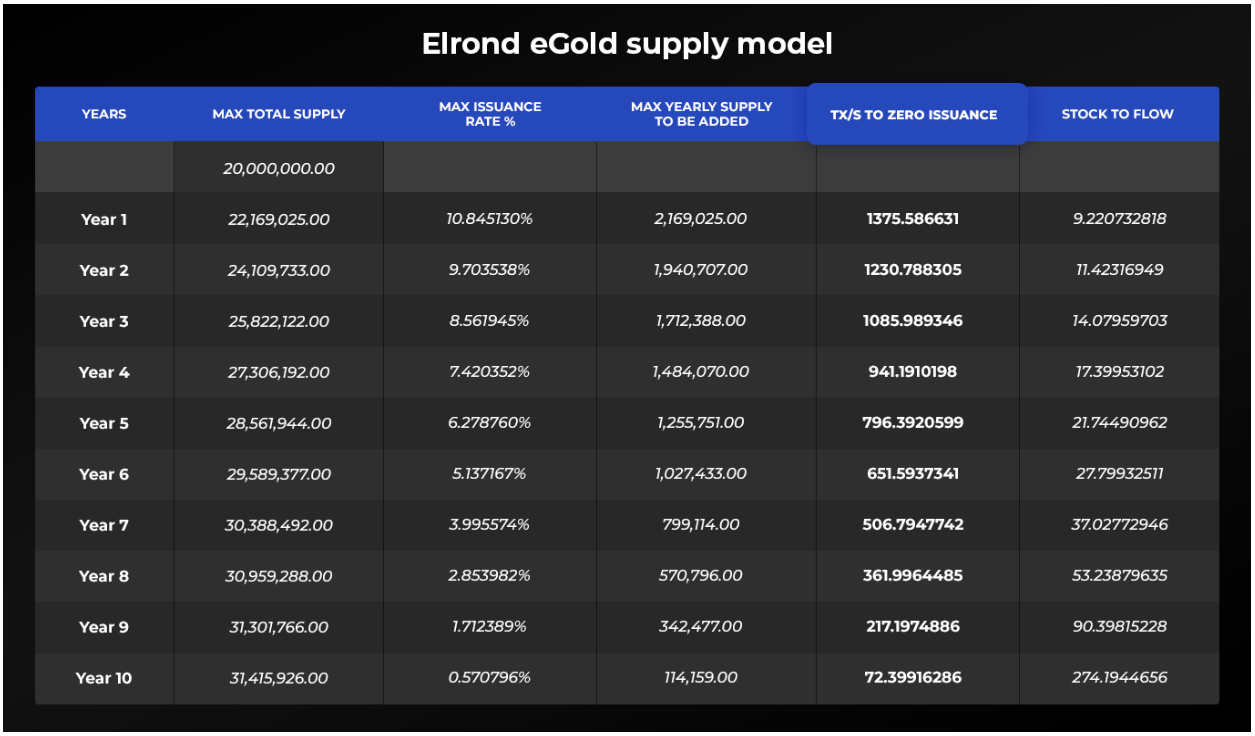
Staking rewards, possibility of slashing, or increasing/decreasing a node rating, are a set of incentives that encourage token holders and validators to secure the MultiversX Network. In return for security, the validators can increase their relative share of token holdings in the network.

We believe that staking rewards do not exist to provide an income stream per se to the token holders. In fact, the economic rationale for staking is not to receive a reward ("yield"), but instead to clearly assert to the validators that staking increases their relative interest (through the amount of eGold owned) in the network, and also contributes to significant token appreciation.

With this in mind, it is better to look at the inflation rate as a token holder dilution rate instead. As such, staking is the best way to grow your token holdings and interest in the MultiversX network.

Here is how rewards will be paid in MultiversX:

There will be a minimum guaranteed reward amount per year. The minimum guaranteed reward amount will come from fees, while the rest will come from inflation. So the maximum inflation rate per year, if fees are 0, is:



Elrond eGold supply model

YEARS	MAX TOTAL SUPPLY	MAX ISSUANCE RATE %	MAX YEARLY SUPPLY TO BE ADDED	TX/S TO ZERO ISSUANCE	STOCK TO FLOW
	20,000,000.00				
Year 1	22,169,025.00	10.845130%	2,169,025.00	1375.586631	9.220732818
Year 2	24,109,733.00	9.703538%	1,940,707.00	1230.788305	11.42316949
Year 3	25,822,122.00	8.561945%	1,712,388.00	1085.989346	14.07959703
Year 4	27,306,192.00	7.420352%	1,484,070.00	941.1910198	17.39953102
Year 5	28,561,944.00	6.278760%	1,255,751.00	796.3920599	21.74490962
Year 6	29,589,377.00	5.137167%	1,027,433.00	651.5937341	27.79932511
Year 7	30,388,492.00	3.995574%	799,114.00	506.7947742	37.02772946
Year 8	30,959,288.00	2.853982%	570,796.00	361.9964485	53.23879635
Year 9	31,301,766.00	1.712389%	342,477.00	217.1974886	90.39815228
Year 10	31,415,926.00	0.570796%	114,159.00	72.39916286	274.1944656

If the cumulative sum of fees during one year is higher than the minimum guaranteed rewards, inflation rate becomes zero and the rewards distributed will be higher than the minimum guaranteed rewards. Otherwise, total fees will just decrease the inflation by the corresponding amount. By adopting this approach, we have created the premises for the transition to a deflationary monetary system.

Since the rewards are fixed at the beginning, the amount distributed to each validator will be proportional to his total number of nodes and their rating. While the rating is more under the validator's control, the number of nodes is under the control of the protocol's governance. At genesis time, MultiversX Network will be bootstrapped with 2169 nodes that will form one metachain and 3 shards (this includes the waiting lists of the shards, containing 142 nodes each). This setup will be sufficient to reach around 15 000 TPS and the level of security and decentralization desired.

We acknowledge that in time the number of shards and nodes might need to be increased, in order to balance the load on the shards and to create more infrastructure support for a higher throughput.

We expect that when the above needs will arise, the additional rewards needed for a new shard with 400 eligible nodes + waiting nodes, will be partially "financed" by an increase in fees (already happening and further accelerated by the new shard), so as to eliminate the need for inflation increase. Thus, we have capped the inflation rate, so as to prevent the increase above the maximum defined rate per year. Ideally, any new shard should be enabled when the amount of fees exceeds the minimum guaranteed rewards by a ratio of $1/N_{sh}$, where N_{sh} is the total existing number of shards.

At a fee of 0.00005 eGold per transaction, it seems that for a TPS between 5000-7000 enough fees are generated to justify an additional shard with no effect on inflation. For each requirement of additional 2000 TPS, an additional shard can be added with no effect on inflation, while keeping the load on the shards below 50%. While further real world testing and more data are required, this is how things have been modeled at the moment of writing.

Here is the calculator for validators that we used for the launch of the MultiversX Network (Genesis): https://docs.google.com/spreadsheets/d/1moHSRVAPeFyVnnx6psHmsUbTUrIBibXyopJAZ5o4zWs/edit#gid=1905747724

2.5 Rewards calculation and distribution

Rewards are distributed at the end-of-epoch by the following rules: 10% of the fee from one block is received by the block proposer, while the rest of 90% goes into a fees pool, *TotalFeesToBeDistributed.* Please refer to the fees section 3 for more information about fees.

At the end-of-epoch a calculation will be done to establish how many new tokens have to be minted. This number is established calculating the *TotalRewardsToBeDistributed* according to *maxPossibleInflation,* and number of blocks produced by each shard, minus the *TotalAccumulatedFees* by all the shards during that epoch. From the amount of *TotalRewardsToBeDistributed,* 10% will be transferred to the Protocol Sustainability Address. See Section 4 for details on this fund.

When the number of shards is changed, the rewards per block is calculated according to the new shard number. If the round time is changed, then the rewards per block is calculated according to the new round time. The calculation of the *RewardPerBlock* is done at end-of-epoch and added to the start-of-the-epoch block by the block proposers, verified by all the validators.

At the end of each epoch:

a. *MinTotalRewardsToBeDistributed* is equal to the total number of blocks produced by all the shards + metachain multiplied with *RewardsPerBlock.*
b. For each block that is produced in each round in each shard, 10% of the sum of that block's transaction fees go directly to that block's proposer, but only after 10% goes to Protocol Sustainability
c. The other 90% of all transaction fees from all shards are aggregated and added to a pool, called *TotalFeesToBeDistributed* denominated in the number of eGold tokens.
d. *TotalAccumulatedFees* is equal to *TotalFeesToBeDistributed* + all the fees which go directly to the block proposers.

    i. If *TotalAccumulatedFees* < *MinTotalRewardsToBeDistributed* then *MinTotalRewardsToBeDistributed* - *TotalAccumulatedFees* tokens are minted and added to the validator compensation pool for a total of *TotalRewardsToBeDistributed* = *MinTotalRewardsToBeDistributed*
    ii. If *TotalAccumulatedFees* > *MinTotalRewardsToBeDistributed* then no additional tokens are minted and *TotalRewardsToBeDistributed* = *MinTotalRewardsToBeDistributed* + (*TotalAccumulatedFees* - *MinTotalRewardsToBeDistributed*).
    iii. From the value of *TotalRewardsToBeDistributed*, an amount of 10% is transferred to the Protocol Sustainability Address.
    iv. The remaining 90% of *TotalRewardsToBeDistributed* is split among all validators (across all shards, including the metachain validators) who acted as consensus group members

e. From the *TotalRewardsToBeDistributed* we calculate the RewardsPerBlock and RewardsPerBlockPerNode according to the number of the eligible validators in that epoch and the number of total blocks produced in that epoch.

f. The new Metachain block proposer of the new start-of-epoch block distributes the rewards (transaction fees and the minted tokens, if any) in the start-of-epoch metablock.

g. The distribution process is a deterministic one, all the metachain validators create the same rewards and must reach to the same conclusion:
    i. Iterate the validator statistics trie and export the following data for each BLS public key: number of times selected in successful blocks, number of times being leader, total accumulated fees and the reward address.
    ii. When iterating all the BLS public keys the process adds the RewardsPerBlockPerNode * NumSelectedInSuccesfullBlocks + TotalAccumulatedFees to the RewardAddress for that BLS public key.
    iii. For each Reward address a reward transaction is created from the metachain to the shards.
    iv. The shards will add the value from the rewards transactions to the accounts balances.

2.6 Unstaking and unbonding

Unstaking

If a validator wishes to unstake, he initiates a transaction that indicates he wants to unstake a number of nodes, including the BLS public key of each node. The transaction is generated by the validator and sent to the metachain.

At the end of the epoch, when nodes are re-shuffled, those who unstaked during the just-completed epoch will be shuffled out first.

- If the node cannot be shuffled out, then the node must “stay and work.” If the node decides to go offline, then their rating decreases and at some point they will be under

ratingThreshold making it ineligible to participate in the next selection or auction process. A node below *ratingThreshold* cannot be un-staked until the rating is above *ratingThreshold (*see *resetRating* transaction).
- If there are more unstaking nodes on a shard than the number of nodes in the waiting list the metachain computes an order for shuffling out and just the first waiting nodes from the list are removed.

If a validator initiates unstaking, and then in the same epoch, decides not to proceed, he can send a re-stake transaction and his unstaking will be canceled.

The unstaking information is saved in the validator staking smart contract. The re-stake transaction is the same as submitting an initial stake, the only difference is that he does not need to send the value again.

Unbonding

The unbond period is set at 10 days, after which the node will be able to retrieve its previously staked funds.

During the unbonding period:

- If the node conducts malicious activity, it is still slashable. This might include attacks such as (see Slashing section):
    - Long-range attacks
    - Non-performance of required validation activities

- It is possible for a node's unbonding period to never end if all the nodes of the system have left, and there are not enough nodes to run one shard. However, this cannot practically happen, as MultiversX will provide nodes for at least the metachain and one shard, at the minimum reserve node price. In this way, we ensure a fail-safe mechanism where MultiversX is the node operator of last resort.

At the end of the unbond period, the validator sends a transaction requesting the unstaked money for each of the nodes that he is choosing to unstake.

The unBond request is processed by the metachain nodes only if the unbond period has concluded for each respective node. If the unbond period has not concluded, then all gas is consumed.

2.7 Delegation

Since not everyone will be able to be a validator and run a node, those who still want to stake, can delegate their stake to other validators or Staking as a Service providers, and split the rewards between them.

At the bootstrapping phase of MultiversX Network, MultiversX as a company will run a number of nodes. Community members will be able to delegate their stake to MultiversX as a staking as a service provider during the first few months. Furthermore, MultiversX has a number of partners who will provide professional services to run large infrastructures; these partners as well as MultiversX will need a delegation smart contract model to start with.

The general requirements for one such contract is to distribute the rewards generated by the validators, towards the community members who staked their token through the contract. The distribution has to take care of when to give the rewards to the registered members, and how much of a service fee is taken out.

More information about delegation in general, delegation at genesis, and the smart contract template for delegation, provided by MultiversX as a guidance, will be later announced in a different paper or medium post.

3. Fees

A sustainable value stream for the network can come from transaction fees and asset inflation. Since the success of the network is reflected by the adoption and usage which will generate transaction fees, the economic model will be able to finance the growth and maintenance of the network without the need of inflation.

The calculation and distribution of rewards and fees is done at the end-of-epoch, and added to the start-of-the-epoch block by the block proposers, verified by all the validators.

For all blocks produced in each round, by each shard, 10% of the block transaction fees go directly to the block proposer. The other 90% of all transaction fees of a block are added in a pool and are distributed to all validators at the end of the epoch. Just the block proposer takes 10% of the fees in the current block.

After reviewing initial simulations, we have decided that transaction fees will start with 0.00005 eGold per transaction.

3.1 Transaction and smart contract fees

Transaction fees are calculated as follows:

1. Value transfer transactions:

$$(moveBalanceGas + storePerByteGas * len(txData\ field)) * GasPrice$$
$$GasPrice >= minGasPrice$$

2. Smart contract transactions:

*(moveBalanceGas + storePerByteGas * len(txData field)) * GasPrice + (actual smart contract processing gas) * GasPrice*

The appropriate block proposer will calculate them directly during the consensus process.

Transaction fees are calculated using a gas model. This takes into consideration: the quantity of resources used per transaction, including:

i. CPU
ii. Bandwidth
iii. Storage

This list provides 584 operations and their associated gas amount that will be used at Genesis (subject to be changed in the future). The 584 operations are Gas only. The denomination comes from gasPrice. There is a minimum gasPrice in the system, under which the transactions are not executed. The gasPrice can be set by the user. The actual fee of the transaction is calculated via gasPrice * gasLimit. The gasPrice contains the actual denomination which is currently 10e-18 eGold. The fee is calculated by the consumedGas * gasPrice.

For any given block in each shard, the transaction fees included in the block are aggregated (see Staking Rewards section). Until the end of the epoch (at which point the pooled transaction fees are distributed to the appropriate agents), the transaction fees are controlled by no agent, and are stored as information in the metaBlock header, inaccessible to nodes in each shard.

Each transaction must specify the amount of gas it needs as part of the transaction data. While a block producer creates a block, it will execute each transaction with the consumed gas being deducted, and remaining gas being refunded. If a transaction specifies enough gas for the execution but insufficient funds for the actual transfer, then the execution will consume the given gas but the move balance function (or smart contract call) will not cause any balance change due to insufficient balance (the account nonce will be increased and the transaction will be added to the blockchain as an invalid transaction).

For any transaction this amount can be calculated by an overestimation (but no more than 10x) of expected gas usage, thanks to the unused amount being returned to the payer, after all transactions are completed. If a transaction doesn’t attach enough gas to execute a required function, the transaction will terminate early and fail, but still charge for spent gas.

For any transaction that specifies less gasLimit as stated in section 3.1, formula 1, the system will reject that transaction thus not notarizing the transaction (not even as a failed transaction).

Future work will explore the possibility to adjust fees based on the load on the entire network, so that, for example, as long as the load is under 50% we have a min. price per gas, but when the load goes above 50%, the gas price increases. In order to avoid manipulation of the gas price by holding transactions, an expiration time will be set to each transaction. Subsequently at the end of each epoch, a reorganization of shards could be triggered, so that smart contracts and dApps are moved to other shards in order to rebalance the load per shards, and return it to under 50%.

3.2 Storage fees

Storage should be considered separately from computation or bandwidth, because each smart contract transaction that will require storage across all validators going forward, is not just subject to a one time fee at the execution of the transaction, but also a storage cost.

MultiversX will introduce a state rent for smart contract transactions, where there will be a fixed price per each byte that needs to be stored (in the future this fixed price can be adjusted via governance), a price that will be paid periodically. The state rent price is applied only to smart contracts and not to normal balance accounts. We will also introduce a mechanism to temporarily clear the state of an account (unable to pay the rent), to hibernate the account, and restore it once needed.

3.3 Developers fees and monetization

In order to significantly accelerate developer adoption, we will provide developers with a built-in protocol monetization solution. Thus 30% of the fees directly associated with a dApp, will go to the developer. So when processing a smart contract transaction, 30% of the fees from that transaction will be added to the smart contract balance.

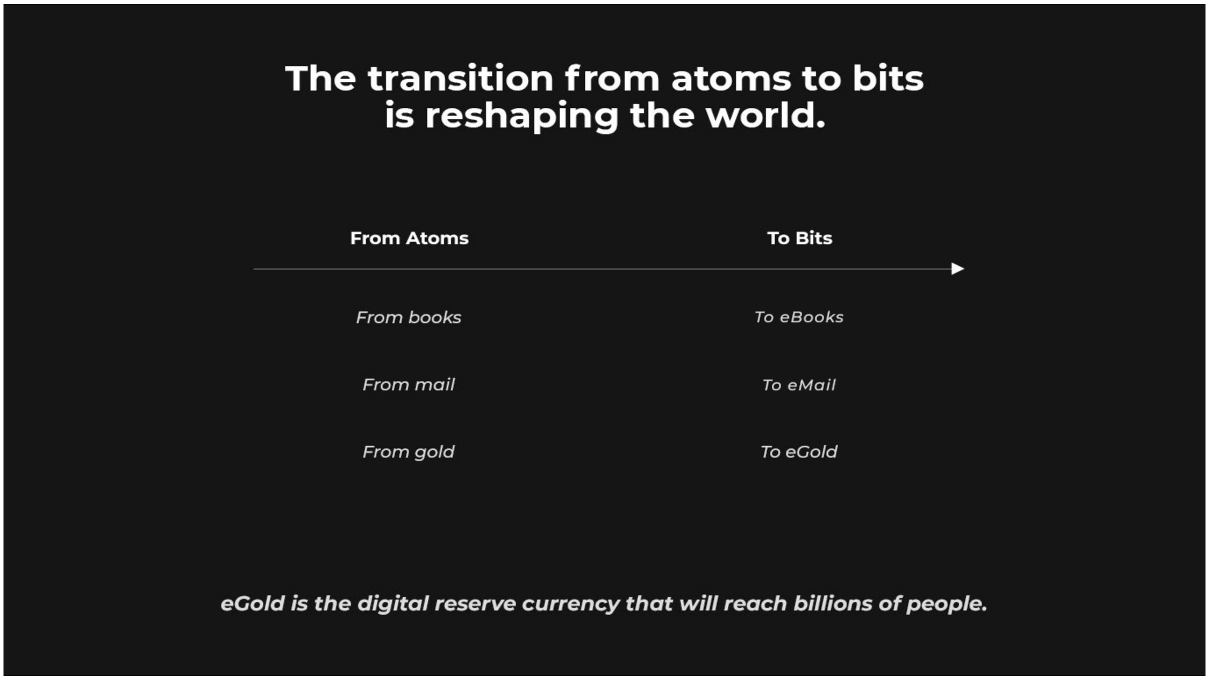
4. eGold

The native MultiversX eGold token is opening a new growth phase for the MultiversX economy. It is a natural step toward enabling native MultiversX services such as staking and delegation, and native DeFi options.

4.1 Overview

Here's an overview of the most important eGold premises:

a) **The eGold currency is designed for simplicity and global adoption**
Complexity is the most important obstacle for real world adoption -- try explaining Bitcoin or Ethereum to normal people and you immediately see what we mean. In order to reach the next billion people, we've completely rethought the MultiversX currency, capturing its essence into a universally appealing and powerful metaphor.



The transition from atoms to bits is reshaping the world.

From Atoms	**To Bits**
From books	*To eBooks*
From mail	*To eMail*
From gold	*To eGold*

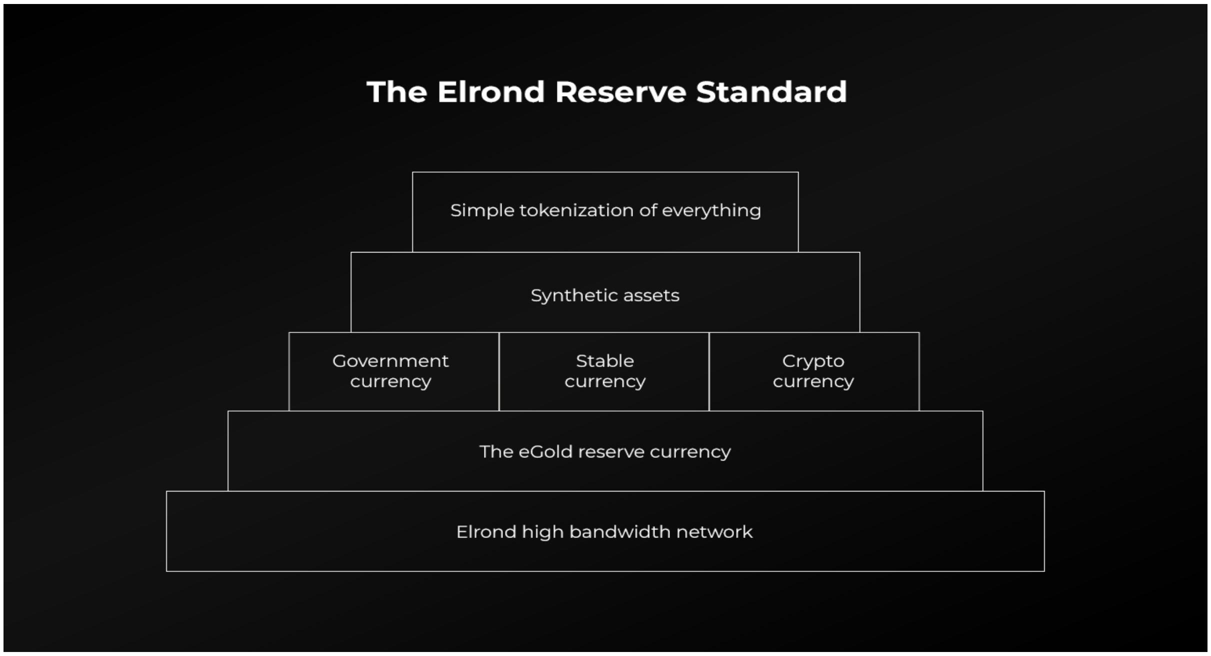
eGold is the digital reserve currency that will reach billions of people.

b) **The eGold currency is designed as a digital reserve standard and robust store of value**

A new economics model has been defined to position eGold as the core network token, fundamental to all of MultiversX's internal usage. This token is designed to optimize parameters that lend themselves to creating a robust store of value, similar to gold, but with mechanics and functionality that go well beyond those of gold.

By enabling a new set of tickers with an e as a prefix, like eGLD, we make things simple and intuitive to understand, but perhaps even better, enable a flexible and coherent derivation path based on the E prefix, compatible with listing an unlimited set of new currencies on top of the eGold reserve.

Embedded in this design is the premise that MultiversX is compatible with both local government currencies and other crypto currencies, which will eventually be able to leverage MultiversX's high bandwidth network, to offer global value transfer to their local communities. In fact, we intend to onboard many new tokens, such as stable coins, synthetic assets, and local fiat currencies.



The Elrond Reserve Standard

Simple tokenization of everything

Synthetic assets

Government currency

Stable currency

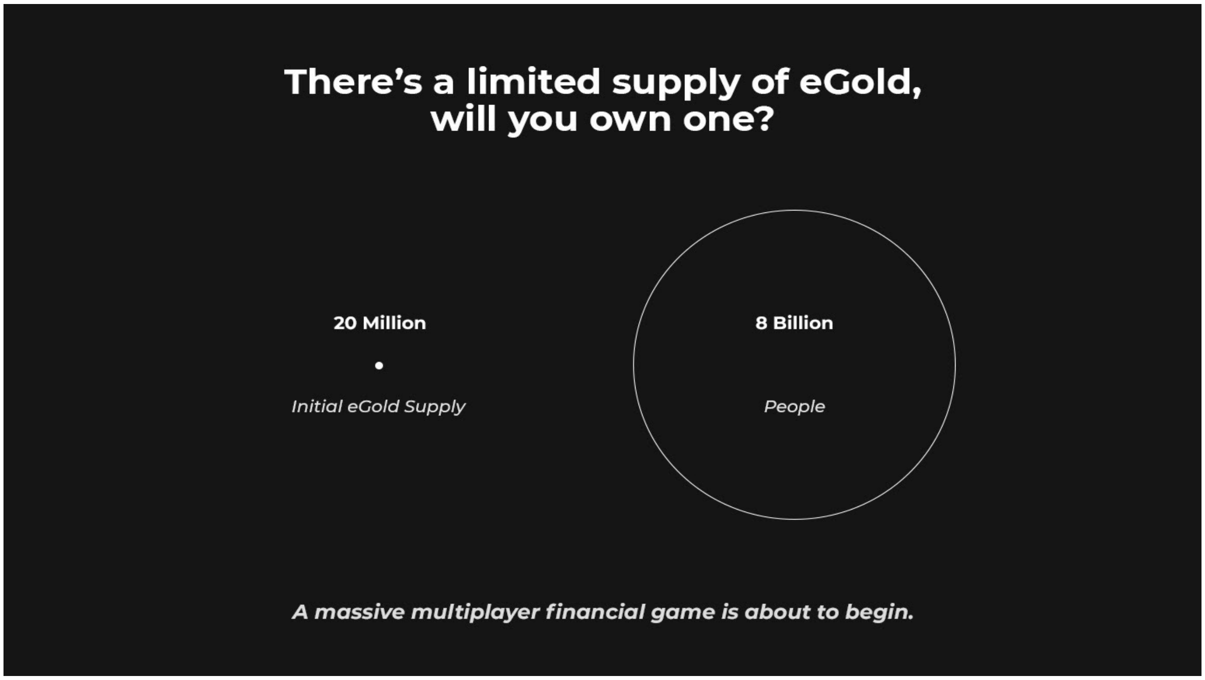
Crypto currency

The eGold reserve currency

Elrond high bandwidth network

c) **Built-in scarcity to reinforce value and demand**

There are only 20Mil initial eGold at Genesis relative to 8 Bil people. This means there is a very limited supply of only 0.0025 eGold per person. This sets an arms race game of accumulation in motion, since owning a few thousand eGold now might be like owning a few thousand Bitcoin in 2010.



There's a limited supply of eGold, will you own one?

20 Million

Initial eGold Supply

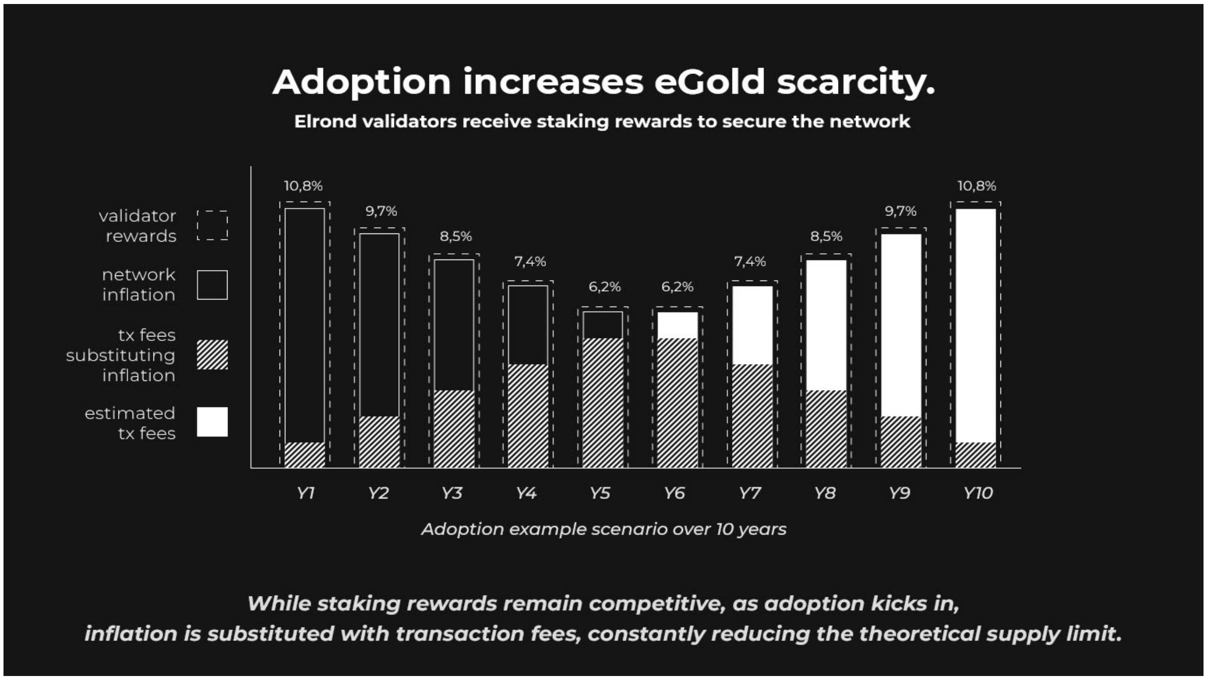
8 Billion

People

A massive multiplayer financial game is about to begin.

d) **Strong staking incentive for validator adoption paired with a max supply limit**

There are strong staking incentives for validators to secure the MultiversX network. At first, these staking incentives come from new supply issued yearly, but as adoption kicks in, inflation is substituted with transaction fees to cover the staking rewards. Furthermore, in contrast to most other blockchain networks where the new issuance is infinite and uncapped, in MultiversX this sum is capped to a theoretical supply limit of 31,415,926 eGold which can be reached over 10 years.



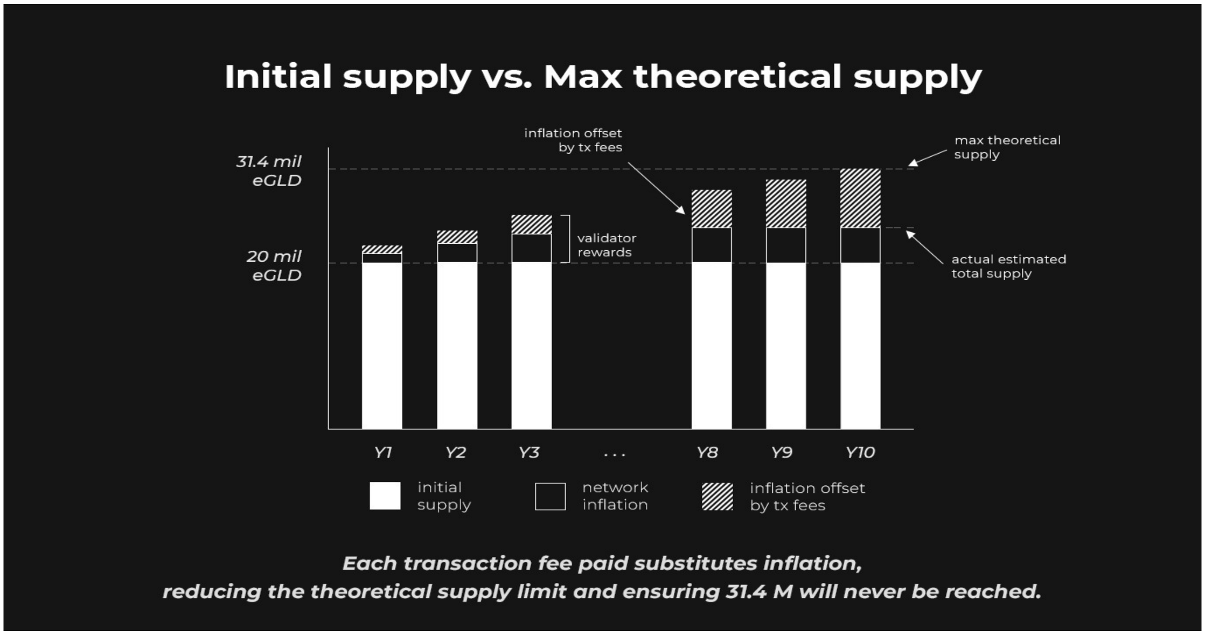
Adoption increases eGold scarcity.

Elrond validators receive staking rewards to secure the network

While staking rewards remain competitive, as adoption kicks in, inflation is substituted with transaction fees, constantly reducing the theoretical supply limit.

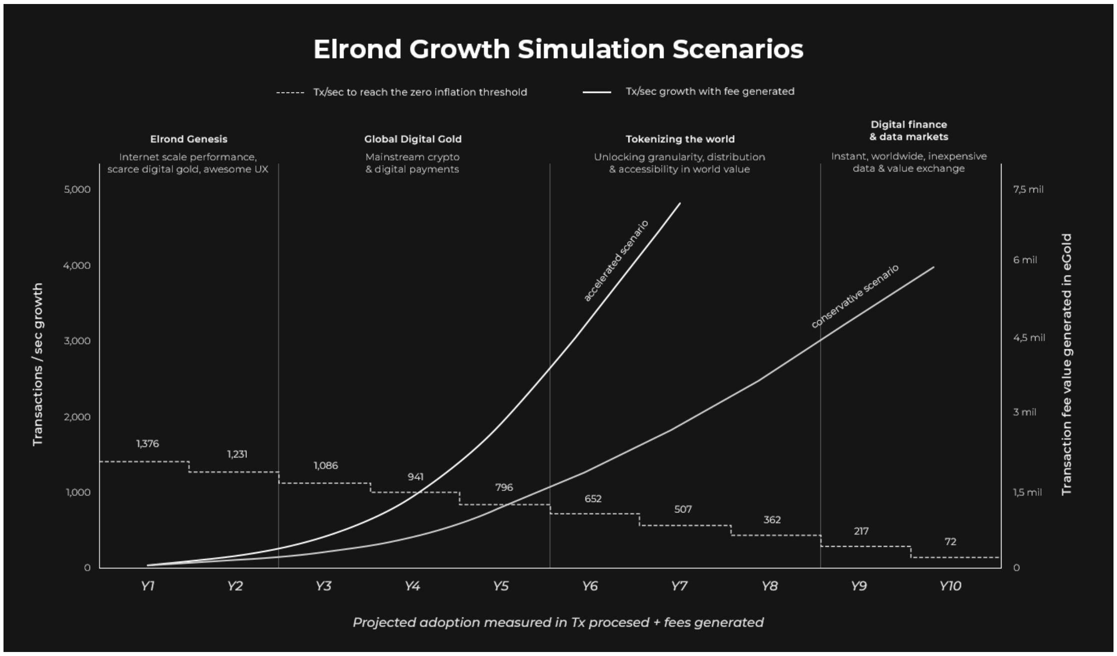
e) Adoption reduces this theoretical inflation and increases scarcity

One of the most powerful features of the MultiversX economic model is that each transaction fee paid reduces the theoretical limit by substituting inflation with fees, thus making eGold more scarce, ensuring that the 31.4M max supply limit will never be reached.



f) A sustainable adoption model growing the entire eGold economy and reinforcing deflation

MultiversX offers arguably one of the strongest adoption models in the blockchain space, thanks to the network being able to immediately transition to a fully deflationary model via any adoption scenario. Indeed, the zero inflation threshold visible in the image below shows that since below 10% of the network capability is needed to cross the threshold, with enough adoption MultiversX can exceed this threshold and create a significant amount of value for all the network participants.

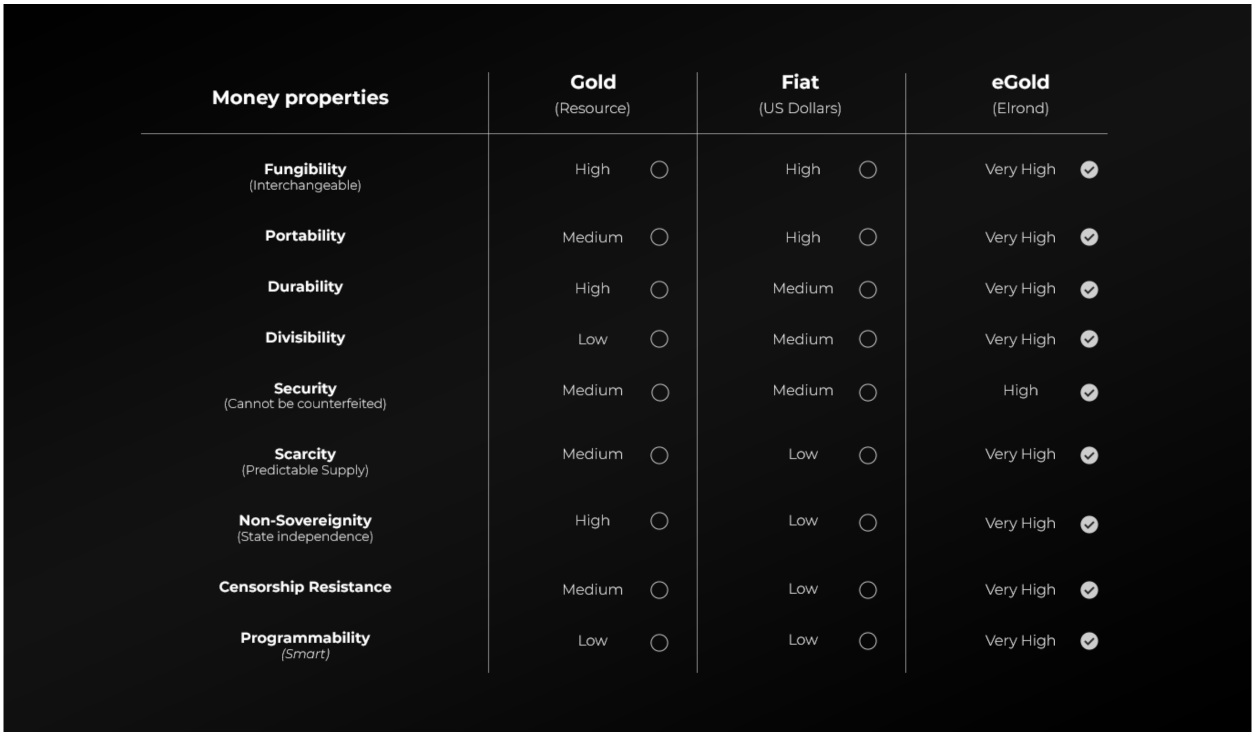


4.2 Properties of money and eGold

There are two types of currencies that have been used recently around the world: representative currencies, where each coin or note can be directly exchanged for a specified amount of a commodity; and fiat currencies, issued by a government, not backed by any commodity, but rather operating by a shared faith between individuals and governments, that the currency will continue to be accepted, and used as means of exchange or payment.

Any currency around the world is counted as a store of value, if it can reliably maintain its relative value over time without depreciating. In addition to being a good store of value, any robust currency must also satisfy certain characteristics related to utility, scarcity, divisibility, transportability, durability, and counterfeitability.

eGold is a new type of digital currency with unique properties that lend themselves to creating a robust digital store of value.



Money properties	**Gold** (Resource)	**Fiat** (US Dollars)	**eGold** (Elrond)
Fungibility (Interchangeable)	High ○	High ○	Very High ✓
Portability	Medium ○	High ○	Very High ✓
Durability	High ○	Medium ○	Very High ✓
Divisibility	Low ○	Medium ○	Very High ✓
Security (Cannot be counterfeited)	Medium ○	Medium ○	High ✓
Scarcity (Predictable Supply)	Medium ○	Low ○	Very High ✓
Non-Sovereignity (State independence)	High ○	Low ○	Very High ✓
Censorship Resistance	Medium ○	Low ○	Very High ✓
Programmability *(Smart)*	Low ○	Low ○	Very High ✓

a) Utility

A currency must have utility in order to be effective. Individuals must be able to reliably trade units of the currency for goods and services. This is a primary reason why currencies developed in the first place: so that participants in a market could avoid having to barter directly for goods. Utility also requires that currencies be easily moved from one location to another. Burdensome precious metals and commodities don't easily meet this stipulation.

Perhaps the biggest advantage of the eGold currency is that it is the native token powering one of the most advanced blockchain architectures, processing more than 15.000 transactions per second at launch, with a capacity able to exceed hundreds of thousands per second. Thus, being digital, eGold is a superior means of exchanging and transferring value, lending itself to fast, worldwide, and cost effective money transfers.

b) **Scarcity**

The key to the maintenance of a currency's value is its supply. A money supply that is too large could cause prices of goods to spike, resulting in economic collapse.

In MultiversX the supply starts at 20,000,000 and exhibits a predictable temporary increase in supply to incentivize network security via staking rewards. The defined maximum supply cannot exceed 31,415,926 over a span of 10 years. However, this theoretical cap will actually decrease with each transaction processed and fees generated. Thus, the stronger the adoption, the smaller the eGold's supply will become.

c) **Divisibility**

Successful currencies are divisible into smaller incremental units. In order for a single currency system to function as a medium of exchange across all types of goods and values within an economy, it must have the flexibility associated with this divisibility. The currency must be sufficiently divisible so as to accurately reflect the value of every good or service available throughout the economy.

MultiversX has a much larger degree of divisibility than most fiat currencies around the world. One eGold is divisible to 18 decimal points. If MultiversX continues to increase in price over time, the large divisibility of MultiversX ensures that with tiny fractions of a single MultiversX, people can still take part in everyday transactions.

d) **Transportability**

Currencies must be easily transferred between participants in an economy in order to be useful. In fiat currency terms, this means that units of currency must be transferable within a particular country's economy as well as between nations via exchange.

In contrast to fiat currencies, where the process of transferring money can take days and have significant fees, as long as there is internet, eGold can be transferred anywhere in the world, in an instant, and at a 100x less cost than current available options. Thanks to being listed on the largest exchanges, eGold can be easily exchanged to almost any currency.

e) **Durability**

Durability is a major issue for fiat currencies in their physical form. A dollar bill, while sturdy, can still be torn, burned, or otherwise rendered unusable.

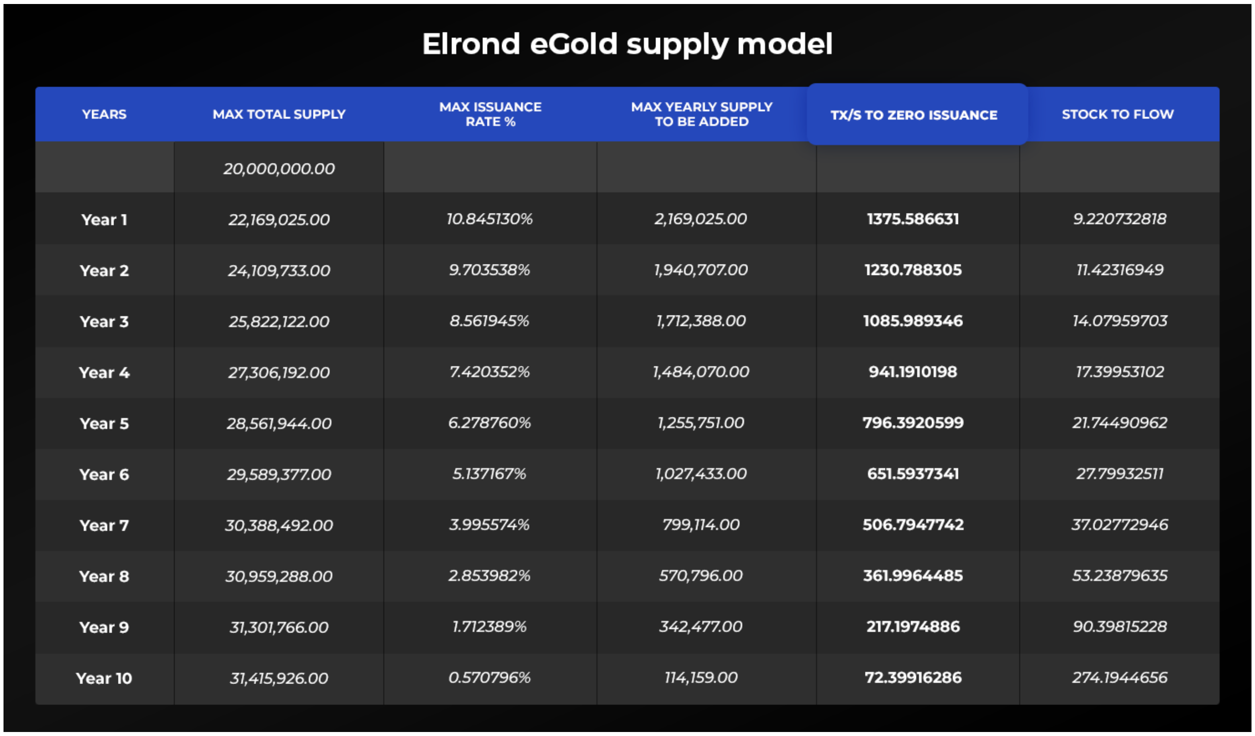
Just as a currency must be durable, it must also be difficult to counterfeit in order to remain effective. If not, malicious parties could easily disrupt the currency system by flooding it with fake bills, thereby negatively impacting the currency's value.

Digital forms of payment are not susceptible to these physical harms in the same way. For this reason, eGold has tremendous value. It cannot be destroyed in the same way that a dollar bill can be, although it can be lost. If a user loses his or her cryptographic key, the eGold in the corresponding wallet may be effectively unusable on a permanent basis. However, the eGold itself will not be destroyed and will continue to exist in records on the blockchain.

f) Counterfeitability

Thanks to the robust built-in security of its decentralized blockchain system, eGold is incredibly difficult to counterfeit. Doing so would essentially require confusing a non-trivial part of the network participants and would require an increasingly large and prohibitive cost. The single way one would be able to create a counterfeit eGold, would be by executing what is known as a double spend attack.

This refers to a situation in which a user "spends" or transfers the same eGold in two or more separate settings, effectively creating a duplicate record. While this is not a problem with a fiat currency note—it is impossible to spend the same dollar bill in two or more separate transactions—it is theoretically possible with digital currencies. What makes a double spend unlikely in MultiversX, is the increasing and prohibitive cost of resources needed to perform it.
Below is a snapshot of the eGold supply model:



Elrond eGold supply model

YEARS	MAX TOTAL SUPPLY	MAX ISSUANCE RATE %	MAX YEARLY SUPPLY TO BE ADDED	TX/S TO ZERO ISSUANCE	STOCK TO FLOW
	20,000,000.00				
Year 1	22,169,025.00	10.845130%	2,169,025.00	1375.586631	9.220732818
Year 2	24,109,733.00	9.703538%	1,940,707.00	1230.788305	11.42316949
Year 3	25,822,122.00	8.561945%	1,712,388.00	1085.989346	14.07959703
Year 4	27,306,192.00	7.420352%	1,484,070.00	941.1910198	17.39953102
Year 5	28,561,944.00	6.278760%	1,255,751.00	796.3920599	21.74490962
Year 6	29,589,377.00	5.137167%	1,027,433.00	651.5937341	27.79932511
Year 7	30,388,492.00	3.995574%	799,114.00	506.7947742	37.02772946
Year 8	30,959,288.00	2.853982%	570,796.00	361.9964485	53.23879635
Year 9	31,301,766.00	1.712389%	342,477.00	217.1974886	90.39815228
Year 10	31,415,926.00	0.570796%	114,159.00	72.39916286	274.1944656

5. Protocol sustainability

The protocol sustainability address will receive 10% from the total generated rewards, in order to provide the necessary resources and funds to further develop, maintain and promote the MultiversX protocol.

Future work

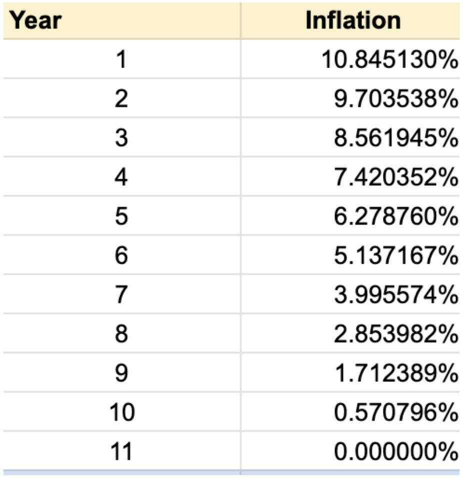
A promising direction for future work will investigate using an algorithmic stable token for fees, and using the stake as a collateral for issuing the stable token.

By enabling a new set of tickers with an **e** as a prefix, like eGLD, we make things simple and intuitive to understand, but perhaps even better, enable a flexible and coherent derivation path based on the E prefix, compatible with listing an unlimited set of new currencies on top of the MultiversX Network.

Moreover, eGold, besides being locked in staking and delegation, could be used to stabilize the value of MultiversX stabilized assets, becoming a reserve component. The reserve might consist of a basket of cryptocurrencies that helps the protocol to reduce the supply of future MultiversX stable-assets.

This paper is the first public draft of the MultiversX economic model. The individuals and companies contributing to this paper operate in a dynamic environment where new ideas and risk factors emerge continually. Thus, we are constantly looking for feedback, with new assumptions that could challenge and improve parts of our model. We encourage those who want to contribute, to provide their feedback on the MultiversX forum.

Constants and formulas

Name	**Value**	**Formula/More info**
initialSupply	20,000,000	
maxPossibleInflation		 Year Inflation 1 10.845130% 2 9.703538% 3 8.561945% 4 7.420352% 5 6.278760% 6 5.137167% 7 3.995574% 8 2.853982% 9 1.712389% 10 0.570796% 11 0.000000%
numNodes	2169	
eligibleNodesPerShard	400	
nodesPerShard	542.5	*Shard 0 will be assigned to take 1 additional node*
waitingNodesPerShard	142.5	*nodesPerShard - eligibleNodesPerShard*
numNodesConsensus	63	
eligibleNodesMeta	400	
numNodesConsensusMeta	400	
targetShardLoad	50%	
epochLength	86,400 seconds	
blockTime	6 seconds	
numBlocksPerEpoch	14,400	*epochLength ÷ blockTime*
validatorPerEpoch	2232 times	*numBlocksPerEpoch ×(numNodesConsensus ÷ eligibleNodesPerShard)*
blockProposerPerEpoch	36 times	*validatorPerEpoch ×(1 ÷ numNodesConsensus)*
nodePrice	2500 eGold	*(1)*

$validatorRatingIncrease$	0,00367	*(4)*
$validatorRatingIncrease_{metachain}$	0,00075	*(7)*
$proposerRatingIncrease$	0,23148 for shard and 0.303030 for meta	*(5)*
$blockProposerRatingNegativePct$	TBC	*(6)*
$importanceRatingRatio$	1	*(3)*
$startRating$	50.00001	
$maxRating$	100	
$ratingThreshold$	10	
$HoursToMaxRatingFromStartRating$	72h for shards and 55h for metachain	
$resetRating$	50.00001	
$resetRatingFee$		*0.1% of the nodePrice*

Appendix

- MultiversX Network Whitepaper
- eGold Token release schedule
- Staking Calculator
- Gas Cost for Operations
- Peer Rating for MultiversX Validators

References

1. *Sapiens: A Brief History of Humankind - by Yuval Noah Harari*

2. *Value Capture & Quantification: Cryptocapital vs Cryptocommodities*
https://www.placeholder.vc/blog/2019/4/26/value-capture-and-quantification-cryptocapital-vs-cryptocommodities

3. *Towards Post-Capitalism*
https://medium.com/econaut/towards-post-capitalism-7679d2831408

4. *Theory of Games and Economic Behavior - by John von Neumann, Oskar Morgenstern*

5. *A Brief Introduction to the Basics of Game Theory - by Matthew O. Jackson*

6. *Mechanism Theory - by Matthew O. Jackson*

7. *Essentials of Game Theory - by Kevin Leyton-Brown*

8. *Game Theory - by Fudenberg, Drew and Tirole, Jean*

9. *Cryptonetworks as Emerging Economies (Done Right?)*
https://a16z.com/2019/02/11/cryptonetworks-economies-governance-capital-access-risk-capital

10. *Crypto, the Future of Trust*
https://a16z.com/2018/12/16/future-trust-crypto-summit-2018/

11. *Programmable money*
https://medium.com/@ElectricCapital/programmable-money-79e16dc7bfca

12. *Voting, Security, and Governance in Blockchains*
https://a16z.com/2019/02/09/voting-blockchains-governance-security-cryptoeconomics/

13. *A Crash Course in Mechanism Design for Crypto Economic Applications*
https://medium.com/blockchannel/a-crash-course-in-mechanism-design-for-cryptoeconomic-applications-a9f06ab6a976

14. *Vitalik Buterin. Blockchain resource pricing*
https://github.com/ethereum/research/blob/master/papers/pricing/ethpricing.pdf

15. *ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER*
https://ethereum.github.io/yellowpaper/paper.pdf

16. *On Inflation, Transaction Fees and Cryptocurrency Monetary Policy*

https://blog.ethereum.org/2016/07/27/inflation-transaction-fees-cryptocurrency-monetary-policy/

17. *The Truth About Staking Yields*
    https://blog.chorus.one/the-truth-about-staking-yields/

18. *MultiversX: A Highly Scalable Public Blockchain via Adaptive State Sharding and Secure Proof of Stake - Technical whitepaper*
    https://elrond.com/assets/files/elrond-whitepaper.pdf

19. *Antifragile: Things That Gain from Disorder - by Nassim Nicholas Taleb*