Whitepaper: 0.1.1

Yield Gold

Non-Sovereign crypto-collateralized yield gold issued as mutually exchangeable digital crypto-tokens and physical notes.

GLT Protocol

mint@glt.gold

mint@physicals.crypto

gltmint@protonmail.com

Web: https://glt.gold

Decentralized Web: physicals.crypto

Contents

1. Introduction

1.1 What's GLT?

GLT is a non-sovereign trustless anonymous crypto-backed yield note linked to average global price of physical gold being issued as both digital tokens and physical tokenized crypto notes.

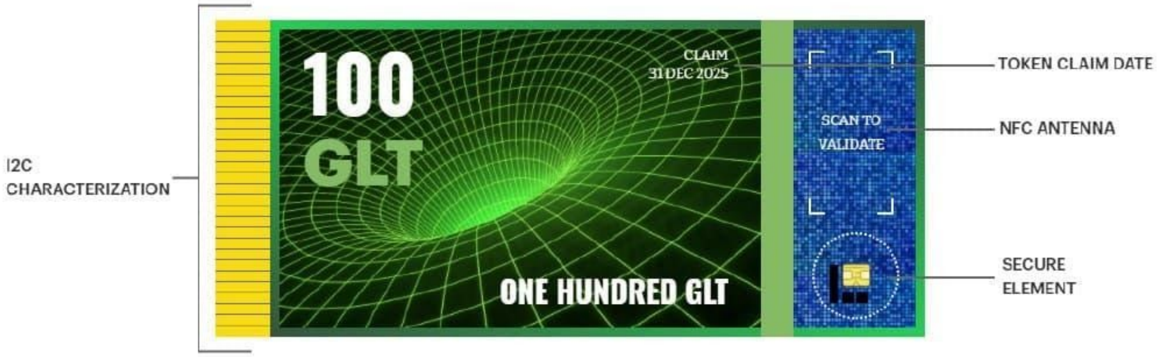
Each physical and physical GLT gold-linked yield note is fully backed by equivalent BTC, ETH, USDT, USDC, wBTC and DOTs, all highly liquid cryptocurrencies and stablecoins having multibillion dollar marketcaps. The collateralized cryptoassets backing GLT yield notes are escrowed via an ERC29 smart contract dynamically as per the physical gold price data feeds of a decentralized Oracle.

Transactions in physical GLT yield notes are free, anonymous, and instantaneously final. No need to pay any transaction or mining fees to anyone. GLT is the 21st century yield gold issued in the form of digital cryptotokens and physical cryptonotes embedded on a flexible circuit board, specialized secure element chips, independent NFC interface capable fo powering the secure element and a kill-switch which will be activated in the event of following three triggers:

> Event#1: Someone tries to probe it physically with an electron microscope and other instruments.
> Event#2: When a hodler/holder decides to convert the physical notes into equivalent digital GLT cryptotokens (ERC20 tokens) and store in his ERC20 compatible cryptowallet.
> Event#3: If and when anyone tries out any hardware attack on a physical GLT yield note.

Each secure element stores an internally-generated ECDSA key pair that is associated with a unique smart contract on the Ethereum blockchain. This smart contract escrows the amount of GLT tokens printed onto the note until a note-specific claim date (e.g. 31st December 2025). Upon expiry, a cryptographic signature created by the note's secure element unlocks a function within the smart contract to transfer the digital token. Restrictions on the format of this signature guarantee that only the person who owns the physical note after expiry is capable of successfully calling the function.

The balance and code of each GLT note's associated smart contract is publicly visible and verifiable on the Ethereum blockchain. The coupling of a GLT note with a smart contract wallet is the first example of a **Physically**-**Locked** Contract, or **PhyLo**. GLT is strongly opinionated that crypto cash should be locked for an extended period of time – ideally years at a minimum – however, one can envision other assets which are immediately transferable from a PhyLo or locked up permanently. Besides hodler/holder may convert the physical GLT yield notes into equivalent digital GLT cryptotokens (ERC20 tokens) and store in his ERC20 compatible cryptowallet anytime he wants and the **kill-switch embedded on the note will be activated automatically and destroy the note and its electronic circuit irreversibility to prevent misuse, double spending and fraud.**



FRONT OF A 100 GLT CRYPTO-BACKED GOLD LINKED YIELD NOTE

The security of physical GLT notes thus crucially depends on the security guarantees provided by the embedded hardware. Specifically, the security of each individual note is reliant on whether the marginal benefits of breaking the note’s secure element (and extracting the key material) exceed the marginal costs. This marginal cost thus puts an upper bound on the value that each note can hold before attempts to break its security become economically worthwhile (see 2.4).

The GLT yield gold token is distributed in two novel ways: (1) minted onto physical GLT notes and

(2) via a perpetual recurring yieldrop. A yieldrop is a means of distributing tokens by locking up a fungible cryptoasset e.g. ETH, wBTC, USDC, USDT and DOT for a period of time. GLT's recurring yieldrop model is akin to mining rewards in other cryptocurrencies. Instead of the proof of work, however, the yieldrop represents the opportunity cost of how the participating ETH/wBTC/USDC/USDT/DOT may have been otherwise invested, spent or sold. The GLT recurring yieldrop is elaborated in section 4.2.

1.2 Motivation

The investment legend of last century Warren Buffet often claimed that gold has no yield and that's why he is opposed to holding gold as an investment. Well GLT is changing that fundamental about gold irreversibly for the first time in the history of human civilization. GLT crypto notes are available as both physical and digital tokens and will be issued at discount to their face values. GLT 5-year Yield Notes will carry 50% discount and hence 50% yield at the time of expiry/claim (YTD:50%) and GLT 10-year Yield Notes will carry 75% discount and hence 75% yield at the time of expiry/claim (YTD:75%). Their value will converge to actual gold price on the expiry date gradually as additional collaterals will be added to the GLT ERC20 smart contract every month periodically as per GLT protocol governed by a decentralized DAO. GLT physical and digital crypto-notes are convertible to each other anytime at the option of the holders/hodlers.

Our primary motivation is to transform both the universally recognized ancient asset(gold) without counterparty risks and newest cryptoasset without counterparty risks blend into each other as a form of hybrid digital/physical currency and a zero-coupon bearer bond making both useful for exchanging physical goods and services as well as for passive investment over the counter anywhere in the world with or without an internet connection.

2. GLT Physical Yield Notes and Cryptotokens

GLT physical notes require several important assumptions in order to function as sound crypto cash and yield note:

1. The ability to internally self-generate an asymmetric public-private key pair
2. The private portion of the key pair can be used to create proofs of authenticity which are verifiable against the public portion; in the case of an ECDSA

public-private key this means signing data presented to the device

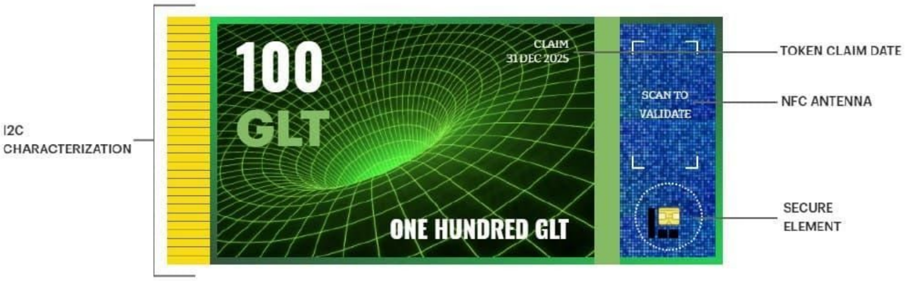
3. The private key is never revealed and cannot be derived at any point in the operation or life cycle of the device
4. The device should represent the simplest embodiment of a system capable of carrying out key generation and signing behavior as described in (1), (2) and (3), given the state of the art
5. The physical notes should be non-clonable to prevent fraud and double spending as well as should be convertible to digital tokens at the option of the note holder/hodler anytime even much before the expiry or claim date.

In order to achieve these characteristics, GLT notes use a secure element chip and integrated kill-switch. Unlike many secure elements available today, however, GLT notes require a variant with limited functionality and a published interface.

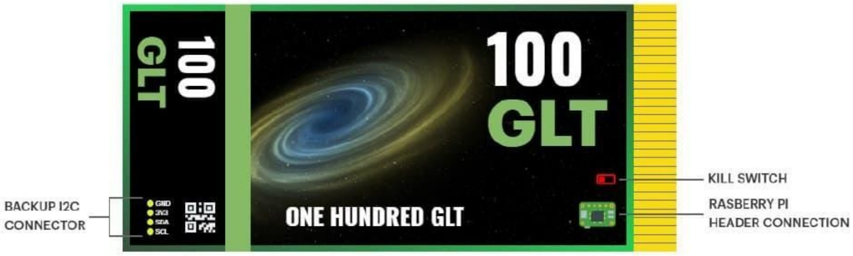
2.1 Form Factor

The GLT note includes many of the characteristics found in traditional paper currencies including a durable, flexible substrate, full color print indicating the denomination and claim date as well as novel visual security features. Like paper money, a GLT note can be folded and unfolded many times. The word elements for each GLT note are distributed across both the printed and copper layers. This paper does not detail all design and security elements in full, but instead focuses on those which allow for cryptographic verification of the note's authenticity.

GLT notes have been designed with an embedded NFC interface alongside the secure element, as shown in figure 2. This interface allows the holder of a GLT note to challenge the secure element via smartphone without any additional hardware. The NFC chip is isolated from the secure element and thus its removal or destruction does not impact the secure element or its ability to sign payloads required to verify and claim the GLT token. Likewise, attacks against the NFC chip do not impact the key material generated and stored by the secure element.



FRONT OF A 100 GLT CRYPTO-BACKED GOLD LINKED YIELD NOTE



BACK OF A 100 GLT CRYPTO - BACKED GOLD LINKED YIELD NOTE

In the case of a damaged NFC antenna or chip, a GLT note can still be audited by one of three electrical interfaces which directly communicate with the I^2C interface on the secure element. This I^2C interface is a direct connection with the secure element chip and bypasses the NFC interface entirely. The backup connector shown in figure 3 lies directly beneath the secure element chip such that only a fragment of the note needs to remain intact for the token to be claimed.

2.2 Secure Elements

Secure elements are a special class of chip that are designed to generate and store cryptographic keys, as well as carry out basic cryptographic operations such as encryption and authentication. Some secure elements provide a full fledged instruction set like ARMv6 which allow for general computation. Broadly, secure elements can be considered to be a type of a HSM (or hardware security module), wherein all functions of the HSM are reduced to a single chip.

The goal of GLT is to maximize the cost of private key extraction relative to the state of the art in secure elements. GLT notes utilize a third party secure element chip with immutable key generation, storage, and signing functions that conform with the assumptions above. Any secure element wherein these functions are created or modifiable via a firmware update is not desirable as this creates additional layers of trust. GLT notes are the first instance of a cryptocurrency wallet where keys are generated and securely stored entirely within the confines of a non-programmable secure element chip and the wallet functions are openly auditable in a non-custodial smart contract. The printer of a GLT note never has access to this private material, nor can they duplicate it onto other notes.

An additional feature of the secure element embedded within GLT notes is the capability to sign data which is internal to the chip using an independent ECDSA key pair. This "internal" signing function is crucial as it can be used to generate signatures which attest to the nature of the chip – most importantly the fact the ECDSA key pair was self-generated, as well as signed random numbers which could have only originated from the chip itself.

2.3 Hardware Attacks

As noted in section 1.2, security guarantees of a given GLT note only extend as far as the costs required to extract a private key from the secure element chip. At the time of publishing, there are no demonstrated or known attacks against the secure element used within the GLT note, however, it's worthwhile to consider several kinds of attacks that might be possible:

- Fault Injection: deliberately introducing faults to the chip at an interface or electrical level such that the device leaks private key material through unintended operation. Fuzzing style attacks which reveal a hidden interface to extract private key material may require anywhere from $100 to 10,000+ USD of equipment. Inducing electrical faults may require costly equipment such as the NewAE ChipSHOUTER or actively modelocked infrared lasers.[8, 9]

- Side Channel Attacks: measuring the chip while it's performing a given operation, notably ones like encryption or signing. During these operations the chip might leak key material via detectable electromagnetic emissions. The cheapest attacks might be feasible with $100 USD of equipment ranging to thousands of dollars for sophisticated spectrum analyzers.

- Physical Probing: typically conducted by exposing the raw die ("decapping") such that optical or electrical probing can be carried out. The probing could be used to directly assess the state of the memory where the key is stored or extract any seed material used for generating entropy. Physical probing attacks typically require expensive equipment such as electron microscopes, ranging from $1,000 to 100,000+ USD.

It should be noted that these attack costs do not take into account the time or expertise required to carry out the attack. An attack requiring multiple hours per note could be significantly more expensive than the equipment costs noted above.

Even with the existence of a successful attack, attackers would still need physical access to any given note they wish to extract the key material from. Physical probing attacks like chip decapping would significantly alter the look of the secure element chip, decreasing the ability of an attacker to re-circulate a compromised note.

See section 5.1 for further discussion on secure elements today and how their trustworthiness might be quantified.

In order to eliminate and mitigate the above mentioned attack vectors, we have integrated a special kill-switch which shall be automatically activated if anyone tries out any hardware attack on a physical GLT yield note.

3. Escrowing Token for GLT Yield Notes

Physical GLT notes are backed by digital GLT tokens. Depending on the configuration of a GLT note, the tokens can either be escrowed in individual smart contracts or available for minting. See section 4 for more information on the GLT `Entropy`, `Registry` and `Yieldrop` contracts which relate to the issuance and minting of the GLT token. For brevity, this description mostly focuses on interface and intended behavior rather than the full implementation.

The complete set of GLT contracts, including additional documentation and tests, will be made available on our Github repository soon.

3.1 Goals and Design

For each physical GLT note, one `Escrow` contract may be deployed on the Ethereum blockchain. The purpose of each of these contracts is straightforward:

1. Hold digital GLT tokens in escrow for one specific GLT note.
2. Allow whoever holds the physical note to transfer it to an Ethereum account of choice once the escrow period has expired.

3. Attest to the origin and nature of the secure element embedded in the GLT note.

The `Escrow` and `Registry` contracts are designed to achieve these objectives under the restrictions imposed by the secure element, the public nature of the Ethereum blockchain, and security considerations.

3.2 Escrow Contract

Upon deployment, the `constructor()` function of `Escrow` sets the following variables, none of which are mutable after deployment:

- `uint256 publicKeyX;`
- `uint256 publicKeyY;`
- `address eccAddress;`
- `address tokAddress;`
- `address lockAmount;`
- `uint256 unlockTime;`
- `uint256 constant BLOCK_DELAY = 240;`

The first two variables, `publicKeyX` and `publicKeyY`, are the coordinates of the public key stored in the secure element in the physical GLT note. This public key is the primary characteristic that uniquely identifies each note.

The next two variables, `eccAddress` and `tokAddress`, set the addresses of two external contracts that each instance of `Escrow` interacts with. The first variable refers to the external `EllipticCurve` contract. GLT notes rely on ECDSA parameterized with curve secp256r1. Because Solidity provides no pre-implementation of secp256r1, we rely on a separate contract to verify secp256r1 signatures in Solidity, represented by `EllipticCurveInterface`. The second variable, `tokAddress`, refers to the GLT ERC20 token contract, represented by interface `GLTERC20Interface`. `Escrow` calls this contract both to get its own GLT token balance and to transfer this balance to a new address (see below).

The constructor also sets the variable `unlockTime`, a UNIX timestamp that controls the date at which the function to transfer digital GLT tokens to another account is unlocked.

The constant `BLOCK_DELAY` controls how recent a signature must be to be acceptable to the `transferTokens()` function. We set this value to 240 so that `transferTokens()` only accepts block hashes from the last 240 blocks or about 1 hour at a block time of 15 seconds (240 * 15s = 3600s = 1h). The `BLOCK_DELAY` is inteded to ensure that only the most recent holder of a GLT note can claim the token off of it while considering factors like Ethereum network congestion.

3.3 GLT Registry

The GLT `Registry` contract identifies secure element chips usable for GLT notes. The `Registry` serves two purposes for GLT notes: resolution of the address of an associated `Escrow` contract for a given GLT note and an attestation of the model of a given secure element embedded in a GLT note.

The `Registry` stores several pieces of information about a given secure element in a GLT note. Importantly this includes sha256 hashes of the secure element manufacturer, model, manufacturer serial number and associated ECDSA public key. The values stored are as follows:

- `bytes32 primaryPublicKeyHash;`
- `bytes32 secondaryPublicKeyHash;`
- `bytes32 tertiaryPublicKeyHash;`
- `address contractAddress;`
- `bytes32 hardwareManufacturer;`
- `bytes32 hardwareModel;`

- `bytes32 hardwareSerial;`
- `bytes32 hardwareConfig;`
- `uint256 GLTAmount;`
- `bool mintable;`

From Registry, any holder of a GLT note can verify that it integrates a secure element which conforms to the assumptions discussed above. They can also look up the associated Escrow contract for a given GLT note to verify the face value matches the stored GLT token. If the GLT note is unminted, it may not have an associated Escrow contract, but rather allows for the GLT tokens to be minted at a future point in time. The additional values stored in the Registry for each GLT note relate to GLT minting described in section 4.1.

3.4 Unlocking Escrowed GLT Yield Notes and Tokens

The function transferTokens() takes the following arguments:

- address _to: The address to receive the contract's GLT balance.
- uint256 _block: The number of the block whose hash was signed by the secure element.
- uint256[2] calldata _rs: The r and s values of the signature.

When the function is called, the contract first checks three timing conditions:

```
require(block.timestamp >= unlockTime);
require(block.number >= _blockNumber);
require(block.number <= _blockNumber +
BLOCK_DELAY);
```

The first condition ensures that transferTokens() can only be called after the claim date printed on a given note. The second and third condition limit the validity of block hashes to those of the previous 240 existing blocks.

Next, the contract verifies that the provided signature is valid:

```
require(_validateSignature(sha256(abi.encodePacked(_to,
   blockhash(_blockNumber))), _rs) == true);
```

The contract expects _rs to be a signature that signs the sha256 hash of the tightly packed arguments _to and blockhash(_blockNumber). We include the recipient's address as part of the expected signature format to ensure that _to is the recipient of the locked balance. By enforcing the age condition for the block hash and including it in the signature, we ensure that only the current owner of the note is capable of calling the function. The function called to verify the signature, _validateSignature(), is an internal function that calls the signature verification contract and returns true if the signature is valid for the public key stored in the contract.

Once all of these conditions are verified, the contract obtains its own GLT balance from the ERC20 contract and transfers the entire balance to `_to`.

4. Issuance

GLT crypto cash is an experiment which intends to test the thesis that a physical form factor for cryptocurrency can achieve the hallmarks of fiat currency like price stability against goods and services as well as broad adoption by non-technical users. To test this question, GLT is the first cryptocurrency issued primarily in a physical, note form. At launch, GLT is distinct from both existing cryptocurrencies, which are not desirable to spend for goods and services due to speculative pressure, and stable tokens that are pinned to inflationary fiat currencies.

There are seven GLT denominations: 1, 5, 10, 20, 50, 100 and 500 GLT. These denomina- tions are intended to mirror a range of typical fiat currency notes and allow for change to be made in a variety of ways depending on a given transaction.

A small amount of purely digital GLT is available via a recurring GLT yieldrop. The GLT yieldrop grows the total GLT by up to 1% annually. The digital GLT emitted via yieldrop compensates for the significant amount of GLT notes that will be destroyed prior to the claim date. Any token or minting right associated with a destroyed note is permanently locked or unusable, and therefore no longer in circulation. The recurring yieldrops also ensure that some digital GLT will be available for development purposes prior to the first claim date; for instance, to integrate GLT in a decentralized finance application. Section 4.2 details how the recurring GLT yieldrop functions.

4.1 Minting GLT

At the point of manufacture, GLT notes may either have associated `Escrow` contracts which are unloaded, or they may instead confer minting rights to the holder of the note to mint GLT tokens. In the latter case, the holder of a GLT note must carry out a minting function against the note which consists of signing a recent block hash after a given amount of time. The initial GLT `Registry` contract stores four variables which control the availability and functionality of this minting function:

- `bytes32 tertiaryPublicKeyHash`: A sha256 hash of the provisioning key pair.
- `uint256 GLTAmount`: The face value of a given GLT note.
- `uint256 mintableTime`: UNIX timestamp after which point the note is mintable.
- `bool mintable`: Boolean determining whether or not the note is mintable.

If the note is `mintable`, then the holder of the GLT note may request that the secure element sign a sha256 hash consisting of the desired recipient address and block hash created after the `mintableTime`. An increment-only counter within the secure element may limit the number of times that the provisioning key pair can be used to generate signatures. The `mintableTime` of a GLT note is similar to the `unlockTime` for notes with associated `Escrow` contracts rather than minting rights.

Within 256 blocks of the `blockNumber`, the sha256 hash created from the block hash corresponding to `blockNumber` and the recipient `address` are then submitted to the `Entropy`

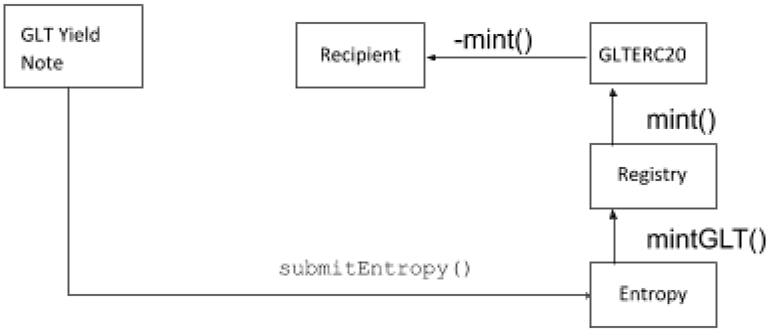


Fig. 5. Minting GLT through submission of random number signed by provisioning key

4.2 Recurring Yieldrop Contract

A yieldrop is a means of distributing token to users which demonstrate that they have assumed the opportunity cost of locking up a crypto asset for a period of time. At the end of the yieldrop period they can then claim back their locked crypto asset as well as a portion of the token in a given yieldrop.

Yieldrops thus far have been singular token issuance events[11]; GLT is available via the first perpetually recurring yieldrop contract – one yieldrop can be initiated once every 30 days. The recurring nature of the GLT yieldrop is designed to mirror mining rewards; users must incur an opportunity cost in order to claim a portion of GLT token in a similar fashion to the way in which miners must incur costs to compete for block rewards in proof of work mining schemes.

GLT is distributed proportionally to those accounts which stake ETH into a given `Yieldrop` instance based on two factors: the amount of ETH staked and the period of time that ETH is staked for, between 30 and 365 days. These two factors equally influence the amount of GLT that will be claimable at the end of a `Yieldrop` instance.

The GLT yieldrop consists of several functions:

- `beginYieldrop()`: Located in `GLTERC20`, this function will allow any account to initiate an instance of `Yieldrop` as long as at least 30 days has past since creation of the previous yieldrop. An amount of token representing approximately 1/12 of 1% (0.08295381%) of the total supply of GLT token, `MINING_REWARD`, is minted into the `Yieldrop` instance. The first account to successfully call the function after the completion of the previous yieldrop deposit period will receive 256 GLT token.
- `stakeETH()`: Once a `Yieldrop` contract has been deployed by `beginYieldrop()`, `stakeETH()` can be called to participate in that instance of `Yieldrop`. It includes two parameters, the value to be staked in ETH and the staking period, `uint256 stakingPeriod`. All value staked is immediately sent to an individual instance of `LockETH`.
- `claimGLT()`: At the completion of the staking period this function can be called against `Yieldrop` to claim the proportional amount of GLT owed.
- `unlockETH()`: Present in each instance of `LockETH`, this function releases the staked ETH at the end of the yieldrop period. The ETH will be returned to the address which deposited it.

The GLT yieldrop does not contain any bonus or “signaling” functions. Users must incur a direct opportunity cost through the locking of ETH in order to participate in the yieldrop.

4.3 Example GLT Yieldrop

Juanita initiates a GLT `Yieldrop` by calling `beginYieldrop()`. At that moment, 12,054,902 GLT token have been minted across GLT notes, the genesis token and previous yieldrops. She receives 256 GLT token. The `Yieldrop` instance she created is available for staking as follows:

5. At the point at which the yieldrop is initiated, approximately 1/12 of 1% of the outstanding GLT token, or 10,000 GLT, are minted into the yieldrop contract. This is the `MINTING_REWARD`.
6. For a period of 30 days any account can lock ETH through the `Yieldrop` instance by calling `stakeETH()`.
7. Each account can only participate in a given `Yieldrop` instance with a single stake.

There are two participants, Lee and Enzo, who decide to lock ETH on the last day the `Yieldrop` instance is open for staking:

- Lee sets a `stakingPeriod` of 30 days and locks 10 ETH.
- Enzo sets a `stakingPeriod` of 60 days and locks 5 ETH.

Lee and Enzo's weights in the yieldrop can be calculated by multiplying each of their `stakingPeriod`'s by amount of ETH each staked. In this case that is 10 * 30, or 300 for Lee, and 5 * 60, also 300 for Enzo. Each participant will receives a portion of the total GLT token in `Yieldrop` in relation to their weights; 300/600 or 50% for Lee and 300/600 or 50% for Enzo.

After his `stakingPeriod` of 30 days, Lee can call `claimGLT()` to receive his proportion of GLT token from the `Yieldrop` instance, 6,000 GLT. Lee may also call `unlockETH()` which will release his ETH to the account he used to deposit it.

Because Enzo staked for 60 days, he must wait another 30 days until his `stakingPeriod` of 60 days is complete at which point he can claim his proportion of the `Yieldrop` instance, also 6,000 GLT. Likewise, he can also call `unlockETH()` at this point in time to release his ETH.

Had either Lee or Enzo locked up ETH at the beginning of the yieldrop period as opposed to the end, then they would have been compensated for that time as well.

4.4 GLT Supply

For the first four years after the genesis of GLT, the following amount of GLT can be issued in a physical note form through the minting process detailed in section 4.1:

- **Year 1**: 2 ** 25 or 33,554,432 GLT
- **Year 2**: 2 ** 24 or 16,777,216 GLT
- **Year 3**: 2 ** 23 or 8,388,608 GLT
- **Year 4**: 2 ** 22 or 4,194,304 GLT

An additional 7,340,032 GLT was issued to the GLT project for discretionary GLT note printing and distribution. As detailed above, each GLT yieldrop increases the GLT supply by approximately 1% annually based on the total supply at the time the yieldrop is started. The frequency at which yieldrops are initiated is dependent on external accounts calling the `beginYieldrop()` function.

The upper bound for GLT token created after five years (the point at which only the recurring yieldrop rewards remain) is roughly 72,700,000 GLT, although this is highly dependent on how many GLT notes are ultimately printed and when yieldrops are created. It's also possible that there will be no increase in the GLT token supply if no account calls the yieldrop function.

The GLT token was deployed August 9th, 2019 at the address `0x177f2ace25f81fc50f9f6e9193adf5ac758e8098`. The first instance of `Yieldrop` took place on the same day.

4. Future Evolution

Eventually GLT physical yield notes will be based on non-clonable quantum cash protected by No-Cloning Quantum theorem of physical nature (https://en.wikipedia.org/wiki/No-cloning_theorem) and digital GLT cryptotokens will be deployed on a galactic-scale infinitely-scalable time-like quantum recursive zkSNARKd blockchain endowed with constant-size blocks providing utmost privacy(balancers, transaction details won't be revealed), security and speed of transactions.

2 Outlook

The GLT project seeks to challenge the nature of cryptocurrencies as abstract, digital-only constructs through the use of novel smart contracts, secure element chips, and token minting mechanisms. The result is a non-custodial physical cryptocurrency that embodies all the psychological touch points of cash that cross cultural and societal boundaries so that anyone can realize cryptocurrency without a prior understanding of elliptic curve math, private keys or addresses.

The first iteration of GLT is strongly opinionated with respect to mirroring the functionality of cash closely, however, it's possible to envision that new financial instruments designed around the same core principles of GLT with distinct operating models may emerge.

At the time of publishing, over 1,000,000 GLT yield notes will be created, and 400,000 of which will be loaded with GLT token immediately.