RAILGUN_

/// WHITEPAPER

MMXXI

RAILGUN PROJECT

JULY 14, 2021

TABLE OF CONTENTS

RAILGUN PROJECT
JULY 14, 2021

RAILGUN

RAILGUN is a robust, zero-knowledge substrate built on Eth//reum, enabling pr1vacy and an0nymity for users and on-chain applications.

RAILGUN: the first time users can maintain privacy while interacting with DeFi smart contracts on Ethereum - without any trade-off from the full security of Ethereum hash-power.

Vision

Privacy is something all humans instinctually value. It is a well-recognized human right, and even those who deny it to others expect it for themselves. Privacy and anonymity should be the default, not the exception. Your consent should be needed before your personal or financial details are revealed to any would-be voyeur.

A small community of passionate and skilled privacy enthusiasts are developing RAILGUN, a privacy and anonymity system built directly on-chain on Ethereum, from which you can interact directly with DEXs, lending platforms, and popular smart contract applications. RAILGUN keeps your actions secret, protecting your privacy, and allows you to keep your identity secret - thus giving you anonymity. RAILGUN does this without you ever having to leave the safety and liveliness of Ethereum and its booming ecosystem. Not only that, RAILGUN will bring its revolutionary advantages to other blockchain ecosystems in rapid succession.

If you ever need to be transparent, RAILGUN can generate a verifiable report of your actions and balances (for an auditor or compliance officer, for example), with a privacy preserving Zero Knowledge method. This means your funds will still be hidden from the public, but you can provide evidence of the sources to your chosen colleague or recipient. The goal of RAILGUN is not to strip away the third-party verifiability of actions taken on-chain, but rather to give back to users the power to choose who sees what, when, and why.

Introduction

Public blockchains are precisely that: public. On the most popular blockchains of today, every bit of your on-chain activity is available for the whole world to see, and will be even a decade later. With commercial on-chain analysis systems improving to a point where you and all your transactions, ever, are trivially traceable, this presents an obvious problem for the present day blockchain user. This is before even taking into consideration any potential efforts and capabilities of professional electronic surveillance organizations that exist in more than few nations these days. This presents a fundamental challenge for large-scale adoption by individuals, NGOs and even corporations. Imagine a world where paying for your morning coffee tells the coffee shop, its barista, and its customers how much you have in your bank account, how much you get paid, and where else you have been spending your money. If you use most cryptocurrencies today, this is exactly the situation you are likely to end up in.

Ethereum is far and away the leading chain for trading and other decentralised financial applications, but interacting privately and anonymously with Ethereum has proven to be difficult. Until recently, the necessary tools have not been available. The solutions that do exist require specialized blockchains that do not have direct access to the DEXs, applications, and liquidity that lives on Ethereum. They also lack the trust the main Ethereum network provides. Many users begrudgingly resort to centralized solutions — which track all kinds of personal data — as a 'lesser evil' compared to sharing their trading history with the entire world, in perpetuity.

What Is RAILGUN?

RAILGUN is a collection of smart contracts that verify zero-knowledge proofs, allowing users to make, send or receive transactions without revealing any assets, amounts, or identities. In much the same way, it also allows users to interact with smart contracts, such as those used for DEX trading, yield farming and other dApps (decentralised Apps). Layered on top of this on-chain system is a suite of adapters called Adapt Modules for existing applications on Ethereum that anyone can deploy. Such a system has two fundamental advantages:

1. Firstly, it increases the size and noise of the anonymity pool. All transferring, swapping, lending, borrowing, and in general, transacting with all kinds of decentralised applications greatly increases the throughput and variation of interactions with RAILGUN, making it progressively more difficult to correlate withdrawals from RAILGUN to deposits into RAILGUN. This means the users who have deposited into RAILGUN are able to achieve privacy and anonymity faster than any of the other systems available on-chain.

2. Secondly, it allows users to keep their assets in their original form, without converting to a different token, within the RAILGUN system longer term. True privacy and anonymity is achieved when assets can not only be privately transacted but also privately stored. With assets in RAILGUN, users can still do everything they can do with assets outside of RAILGUN, reducing the incentive to ever move these assets out. This helps to increasing the size and noise of the anonymity pool, offering a much better level of privacy and anonymity.

By creating an ecosystem where privacy, anonymity, transfers, trades, and any other activity all happen in one place, all participants benefit from an increasingly large and noisy anonymity pool. All users of the system are able to piggyback off the activity of others, increasing their own privacy and anonymity.

How Does RAILGUN Work?

RAILGUN is composed of these key functions:
ADD transfers assets into RAILGUN and creates a new zero-knowledge note, representing all of the assets and their owner. This action is not in itself private (because it originates from outside the system, but it is the first step towards creating privacy). The note is then added to the live pool.

SPLIT turns one or more zero-knowledge notes into two zero-knowledge notes. The input notes are moved from the live pool to the dead pool, and the output notes are added to the live pool. All of this is done in zero-knowledge, where the user proves that they own the input notes, and that the input notes have not been used before, without revealing the notes themselves. Splitting can also be used as a way to transfer funds by setting a different owner in one of the newly created notes. This helps to hide the precise intent of the split.

REMOVE transfers assets from RAILGUN to an arbitrary address by destroying a note. Again, this is done in zero-knowledge, where the user proves ownership of the destroyed note without revealing the note or themselves. However, because the receiver is outside the system, this action reveals the address to which the funds are transferred, and the amount of funds. But not the actual user it came from, only that it came from the RAILGUN system.

All actions will also come with varying levels of privacy and anonymity and therefore the user can select the type of action to reduce gas costs where necessary. Users will also be able to batch multiple actions into one proof to reduce gas costs.

Technical

RAILGUN was designed to make it possible for you to:

- Build a fully private store of your cryptocurrency assets, at rest
- Trade and participate in DeFi platforms with complete privacy & the full security of the blockchain
- Swap tokens privately with other users, without an on-chain activity trace
- Produce a proof of your source of assets - for compliance purposes, for example

In addition to that, a RAILGUN DEX will be released shortly (provided the DAO votes in favor of doing so) to facilitate peer-to-peer trading that is invisible to external parties yet built within this same on-chain technology that makes RAILGUN possible.

RAILGUN Core System & Protocol

The core of the Railgun protocol is a JoinSplit transaction, which operates on the (U)TXO – that is (Unspent) Transaction Output, model. Here, the U is in brackets because any outside observers are unable to determine which TXOs (Transaction Outputs) have been spent and which haven't. Each UTXO is an encrypted note of a public key that establishes who can spend it, an amount, a token ID (for example USDT, WETH, WBTC etc., represented by their respective ERC-20 contract address), and also a randomness field to randomize the note commitment (i.e. hash). We internally use efficient implementations of batch incremental Merkle trees (a tree data-structure that allows efficient and secure verification of the contents of large data structures, using cryptographic hash functions) to maintain this information, on-chain.

We use Nullifiers, which is a particular type of hash that is generated using the users' private keys, that aren't linkable to the TXOs by outside observers, but are deterministically generated in transactions to eliminate any potential for double spends. Nullifiers are calculated by using the hash of the Spending Key (spender's private keys for the UTXOs in question) and path indices of Merkle root, which ensures that each note will always generate unique Nullifier. Since the Nullifier can only be generated by the spender, this allows RAILGUN to use them as double spend elimination. The spender is the only one able to link which Nullifier belongs to which UTXO.

RAILGUN uses zero-knowledge proofs heavily within the system. Zero-knowledge transactions can be thought of as small programs with some public inputs and some private inputs. A prover program therefore can generate a proof with both public and private inputs and then give the public inputs to a verifier program, along with a ZKP (Zero-Knowledge Proof). With this information, the verifier program can now verify what the prover program has successfully executed. The public inputs exist as a part of the sufficient information to prove that the private inputs were what the verifier expected, and are not forged values. The Merkle tree root of our UTXO set ensures very efficiently that the prover is not able to fraudulently claim 'I have a UTXO with 10000000000 ETH'.

In our zero-knowledge program we have the following public inputs:

- Adapt ID (more on this in a later section)
- Deposit amount
- Withdrawal amount
- Merkle root
- Nullifiers
- Output UTXO hashes
- Output UTXOs encrypted, which are only decryptable by receiver

And also the following, optional, public inputs:

- TokenID which must be public if deposits or withdrawals must be made, but private if both deposits and withdrawals are zero
- An address to withdraw from which must be public for non-zero withdrawals

The private inputs are:

- Input amounts
- Spending Keys which hare the private keys for UTXOs being spent
- Merkle proofs of membership
- Recipient public keys
- Output amounts

With these inputs, the zero-knowledge program verifies the following:

- Deposit amount + input amounts = withdrawal amount + output amounts, so no one can create tokens out of thin air
- Input notes exist in the Merkle tree by using Merkle root and Merkle proofs of membership
- The spending keys are valid for the input notes since only the private key of the notes can spend them
- Nullifiers are correctly calculated

The contract checks to make sure:

- The zero-knowledge proof for the transaction is valid
- That it hasn't seen the nullifiers before, eliminating any double spends
- The Merkle root is the current or a previous Merkle root, to prevent users from making up UTXOs
- Then the smart contract transfers tokens from the user wallet to itself, if a deposit amount is specified, or transfers tokens from itself to the specified user wallet address in the public inputs, if a withdrawal amount is specified instead. The hashes of the output UTXOs are then added to the Merkle tree so they can be spent in the future.

Adapt Modules

Adapt modules are separate smart contract extensions to the RAILGUN protocol that can facilitate features such as private trades and NFTs, etc. Such Adapt modules can implement extra functionality without bloating the core RAILGUN code. The Adapt ID interface itself is simple - yet powerful.

The Adapt ID interface consists of two fields, a contract address and parameters.

If a contract address is specified, the RAILGUN core contract will only process the transaction if the transaction has been submitted from the contract address specified via the Adapt ID module interface. Because any relevant proofs are only submittable from the specified contract, they are only valid if they pass both the RAILGUN core protocol's validation rules as well as the Adapt module's validation rules.

The Adapt ID parameters are not validated by the core RAILGUN code, which allows Adapt modules to implement whatever custom logic they wish here (could be a set of interactions with an external DeFi contract like AAVE, for example).

Keeping Adapt modules in separate contracts also ensures that RAILGUN user funds are not at risk from poorly coded or malicious Adapt modules.

Let's look at how and Adapt module could be used by Alice and Bob to swap their tokens:

- Alice wants to sell 100 USDC for 100 USDT, so she generates a note for 100 USDT spendable by herself (let's call this note A)
- Bob wants to sell 100 USDT for 100 USDC, so he generates a note for 100 USDC spendable by himself (call this note B)
- Alice sends note A, to Bob and Bob sends note B, to Alice
- Alice creates a proof that spends to note B with the hash of commitment A as the Adapt ID (call this proof A)
- Bob creates a proof that spends to note A with the hash of commitment B as the Adapt ID (call this proof B)
- Bob sends his proof to Alice. Alice sends her proof to Bob. Either Alice or Bob send both proofs to a common Relayer. In this example Alice will send both
- Alice submits both proofs to the Swap module (via a Relayer). The Swap module checks that the Adapt ID of proof A is equal to one of the note's hash of proof B and the Adapt ID of proof B is equal to one of the note's hash of proof A. If so, both proofs are submitted to the RAILGUN system as an atomic transaction. Entire transaction reverts, if either proof fails

Swap Transactions and RAILGUN DEX

Swap transactions make use of the Adapt ID interface. Each RAILGUN transaction specifies the hash of an output that it wishes to be filled. The validation rules on the Swap Adapt ID module ensures that the transaction is only valid if another transaction is submitted alongside it that outputs to the specified output hash. So, any swaps are performed in an atomic and trustless manner - only pairs of transactions with outputs matching each other's requests are valid and will get executed.

Relayer Network

In RAILGUN, anyone can be a Relayer. Users specify the transaction and gas price they wish to submit. The Relayer then responds with a fee to be paid (to cover the Relayer's ETH gas cost). The user then generates a RAILGUN transaction with one of the outputs to the Relayer's address matching the requested fee. The Relayer checks that one of the outputs is addressed to them with the correct fee, and then sends the transaction both to the network and to the user at the specified gas rate. This prevents the user's internal RAILGUN transactions from being associated with their ETH address.

Potential Technology Roadmap:

After launch, DAO users can vote to:

- Deploy RAILGUN Core with reference frontend
- Deploy RAILGUN on Binance Smart Chain & Polygon (Likely July/August 2021)
- Deploy the Relayer network
- Batch transaction verification capabilities making internal & swap transactions cheaper
- RAILGUN DEX (RAILYSWAP)
- Pay transaction fees in a different token to what's being transacted
- Private NFT support
- Fully Private NFT auctions
- Private voting via NFT'd stakes
- SOL / Solana RAILGUN (SOLRAIL) deployment (Likely November 2021)
- Polkadot RAILGUN (DOTRAIL) deployment (Likely January 2022)

Governance

RAILGUN is not under the control of any one person or team, and it never will be. The guidance will always come from the RAILGUN DAO, a DAO (Decentralised Autonomous Organization) where the governance token holders vote on proposals that define the operations and direction of the project. RAILGUN smart contract code is only deployed or updated after a DAO governance vote. At launch, RAILGUN DAO will go live without any RAILGUN privacy contracts deployed – the version of the code that is deployed will be the one voted on by RAIL token stakers.

RAIL Token & Voting

RAIL is the governance token of the RAILGUN DAO. One token (staked in the voting contract) equals one vote. Users who have not staked, or who are unstaking, will not be able to vote. Once the RAIL tokens are staked, the unstaking period is 30 days, thus there is an effective minimum time of 30 days to hold the token after a vote is cast. This means that voters have to look at least a month in advance when they choose how they vote for protocol upgrades or fees. There can be no "Vote Raiding" and voters will look beyond the next few days.

RAIL Distribution

The distribution of the RAILGUN DAO governance token, RAIL, can be summarized as below:

- 25% allocated to Airdrop
- 25% allocated to the Foundation
- 50% allocated to RAILGUN DAO

Total circulating token supply at launch will be 50 million RAIL tokens. Maximum lifetime total supply of the RAIL token would be 100 million tokens - creating more than 100 million is impossible.

Airdrop tokens (25%): Ethereum addresses who have made donations on the ETH network to privacy charities and nonprofits, such as the TOR Project, the Right to Privacy Foundation, the Free Software Foundation and others, will be given an airdrop of RAIL tokens. The most important part of the forming of any community is the inclusion of members, and members that have already proven their long-term interest in the goals of Railgun are the perfect members to start the DAO. Many of these recipients may not be initially aware that they are receiving the airdrop, as there is no way to directly inform them.

Foundation tokens (25%): The Right to Privacy Foundation issued a grant to develop RAILGUN project and has volunteered to take custody of 25% of the RAIL tokens, in order to support the long-term benefit of the Project. The foundation is a registered charity and is not motivated by profit. These tokens will only be used for incentivizing developers and the promotion of the RAILGUN platform, including future deployments. The Foundation will not be a seller of tokens for the first DAO year.

DAO tokens (50%): The 50 million tokens given to the DAO are locked and unminted. They can only be minted by a DAO vote by RAIL token holders. For example, if the DAO wanted to give a bonus yield to liquidity pool runners of the RAIL token, the DAO will vote to mint and unlock the required number of RAIL tokens from this allocation. The DAO can vote on how to distribute transaction fees from the RAILGUN Treasury to RAIL token holders

Summary

All users should be a meaningful part of RAILGUN, and influence the growth and direction of the project as a community that values privacy in the digital age. The RAIL token will be used in the DAO governance system for this kind of teamwork. Users will be able submit proposals for change, and to vote on changes, privately and securely. In the first weeks, we saw proposals by early supporters, from native desktop and mobile RAILGUN app ideas to the distributing of transaction fees collected in the RAILGUN treasury to staked RAIL token holders.

ECONOMICS

The following is the economic proposal that will be delivered to be voted on by RAILGUN DAO members in the first week after the launch. The RAIL stakers, who will have the DAO voting rights, may propose any alternatives - or vote either for or against this particular proposal. Please refer to the Governance section for further information on the governance model and process.

The Economics of RAILGUN and the RAIL Token

The economic policy of the RAILGUN network will be controlled by those who interact and contribute directly to the RAILGUN system and its anonymity pool. The RAIL token will be used to govern the system, its upgrades, and its parameters. It will also be used to incentivize participation in governance and growth.

RAIL will be continuously emitted to all liquidity pools. The RAIL liquidity pool will be the first to exist, and its liquidity providers will earn governance power proportional to the amount of RAIL deposited. The private and anonymous nature of liquidity pools allow RAIL holders to remain unknown and govern in secrecy, a property unique to the RAILGUN system that protects the voters and contributes to a more robust, independent RAILGUN DAO. The exact emission schedule will be governable by RAIL holders but will start with a 10-year emission schedule.

Fees and the RAILGUN Treasury

Fees will be coded into every interaction: the fees taken are sent into the RAILGUN DAO Treasury address. Only majority votes of the DAO can control the spending of DAO Treasury funds - no other person can move them. All RAILGUN fees will initially be set to zero except for the ADD function fee and REMOVE function fee which will be set at the start to 25 basis points each. This means the RAILGUN privacy smart contract will charge 0.25% of the tokens to desposit or withdraw. With a large volume of privacy-preserving transactions expected to take place through the RAILGUN privacy contract, from the start, the fees collected in the Treasury are expected to grow very large from RAILGUN privacy contract activity.

Liquidity Provisioning for RAIL

RAIL will be released from the DAO to early users and liquidity providers of the early versions and prototypes to incentivize participation in development and testing. In general, RAIL tokens will be released to liquidity providers and active users over the next 10 years, in proportion to their activity on the network. This serves firstly to incentivize active use of the network. And secondly, this makes sure that the governance of RAILGUN goes to users who are actually using RAILGUN and are strongly aligned with its success.

Multiple Chains

As the RAILGUN project deploys RAILGUNs on Binance Smart Chain & Polygon first, and the later on Solana & Polkadot, the new corresponding tokens (POLYRAIL, SOLRAIL, DOTRAIL, etc.) will be airdropped ONLY to those who are staking RAIL and to those holding any LP tokens for RAIL. The alternate-chain deployments like Binance Smart Chain RAILGUN (BRAIL) and Polygon RAILGUN (POLYRAIL) should be airdropped to those providing RAIL liquidity on DEXs (those who hold LP tokens) rather than only to RAIL stakers.

No other parties will receive these new token airdrops and there will be no reserve supply at all. These new non-ETH chain tokens will have a completely new market cap and price, independent of RAIL.

TRAIL

Pr1vacy &
An0nymity

JULY 14, 2021