How Enshroud Compares with Tornado Cash

21 April, 2025

Tornado Cash (TC) is a privacy service on Ethereum (and other chains) which was sanctioned on 8th August, 2022 by the US Treasury Department's OFAC group. These sanctions were clearly inappropriate, and done for largely spurious and political reasons, spurring several lawsuits. As of this writing, the sanctions have been rescinded by OFAC, although the prosecution of TC developer Roman Storm has yet to be resolved. "Sanctions," be it noted, constitute an illegal jump from accusation straight to punishment, leap-frogging normal due process completely. This document compares and contrasts the operational capabilities of Enshroud with TC. We will undertake a point-by-point comparison of similarities and differences, so this writeup will necessarily be moderately technical.

Brief Summary of TC

Eliding the full technical details, a qualitative description of TC's usage is as follows:

1. A user generates a random secret and supplies it to the TC smart contract, along with a desired deposit duration (time *T*) and a selected fixed amount.

2. The TC contract hands back an encoded deposit note (receipt), used to represent a proof-of-ownership of the deposit.

3. The fixed quantum of an asset (such as 0.1 ETH or 1 ETH) is then deposited to the smart contract's corresponding pool (e.g. the 1 ETH pool) with the encoded deposit note, joining a pool of coins supplied by other users.

4. The user waits for the specified length of time *T* (such as 7 days).

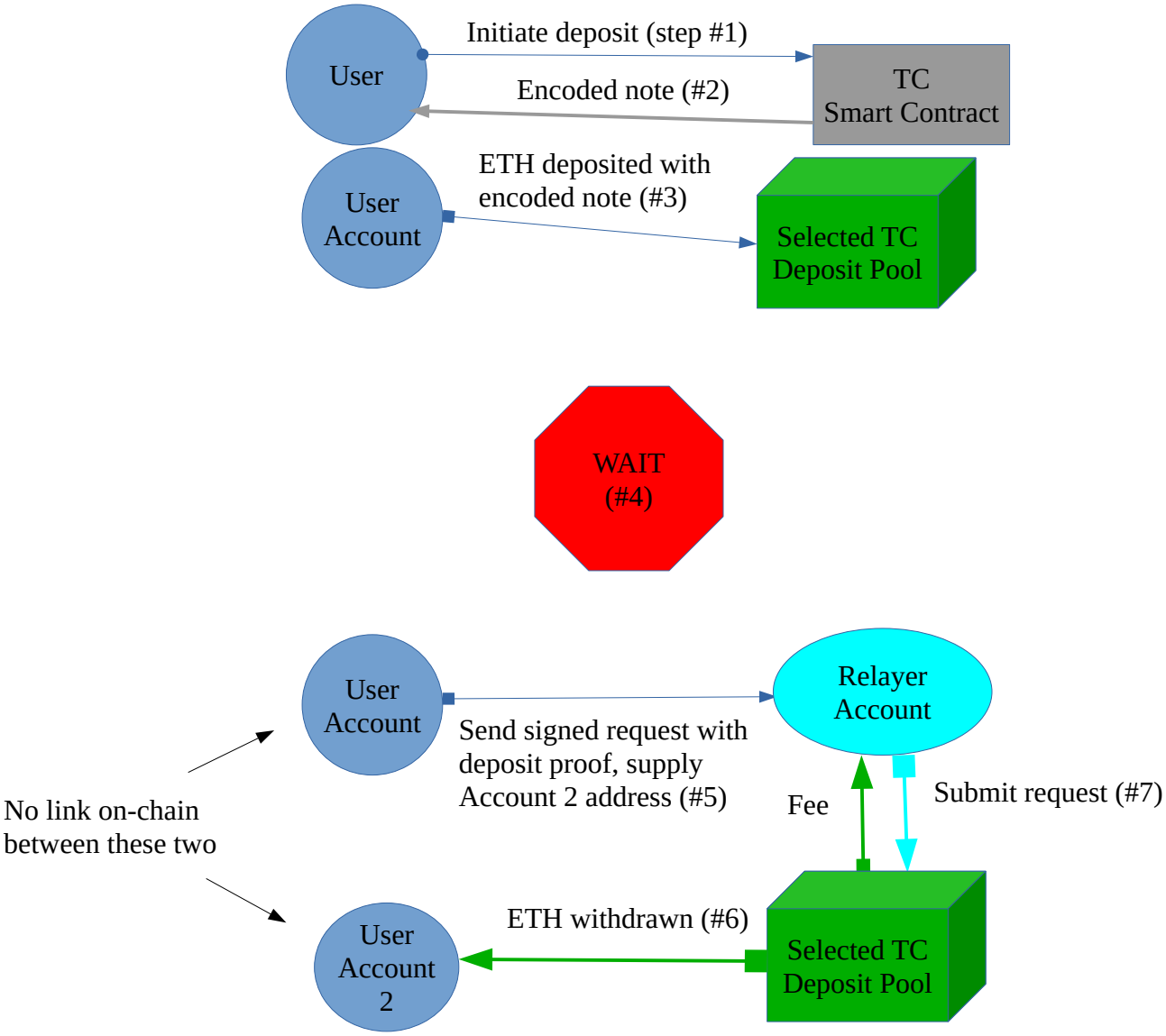
5. The user submits a solved ZK-proof which proves they are in possession of the original secret associated with the deposit, also designating a withdrawal address.

6. The TC contract then permits the designated withdrawal address to withdraw the original quantity of deposited coins from the pool.

7. (Optional) Because the withdrawal address needs ETH (for gas) in order to make the withdrawal from the TC contract, and it's best to use a fresh wallet address for that purpose, a "Relayer" (agent) can also be designated. The Relayer pays the gas cost and performs the withdrawal for an agreed fee, spending the net amount to the withdrawal address.

The net effect is that there is no link on-chain between the depositing address and the withdrawing address. The degree of uncertainty created is directly proportional to the number of other users entering or leaving the same asset pool during the time *T* which elapses between deposit and withdrawal.

The process can be represented pictorially thus:



A Few Notes

- Because pool deposits and withdrawals need to be uniform, arbitrary amounts cannot be used, only those specific amounts and asset types for which a deposit pool exists, as defined by TC.

- The normal assumption is that the depositing User Account and withdrawing User Account (Acct #2) are controlled by the same party. There is no requirement that this be the case. (That is, the withdrawal could be a payment to a third party.) But since the amounts are always fixed quanta, this is unlikely.

- TC only ever supported a small set of assets and pool quanta. Since the USDC pools were frozen by Circle as a result of the sanctions (and the USDT pools could have been frozen), effectively only the ETH pools remain in use.

- Based on this experience, it might be a valid design constraint that no smart contract privacy dApp should support any ERC-20 or other token which features an address blacklisting mechanism. (Possibly, not an update mechanism either, lest blacklisting and/or KYC whitelisting get added in future.) Currently this criterion affects all major USD stablecoins. Ironically, it's precisely these coins which would benefit most from transaction privacy. Because the current US administration appears to desire the global adoption of USD stablecoins on crypto rails, and everyone understands that having privacy available would greatly facilitate that wish, it seems likely that stablecoin issuers such as Circle and Tether are unlikely to get pressured to blacklist smart contract addresses at this time – even privacy-related contracts. We therefore conclude that stablecoins will commonly be used with Enshroud, but that users should do so at their own risk, given the impossibility of predicting future political wind directions.

- TC offers an optional Compliance Tool which uses the deposit note to generate a document showing the real depositing and withdrawing addresses and amounts.

What Enshroud Offers

The usage process for Enshroud is actually similar to TC, but provides much more flexibility and achieves a different goal (encrypting spends rather than sterilizing coins). The process works like this (with some ==> *comments* interspersed):

1. **(Deposit)** A user deposits any amount of any ERC-20 compatible asset to the Enshroud smart contract's address (via ERC-2612 Permit/Transfer or ERC-20 Approve/Transfer). The contract responds by awarding the user account an equal amount of balance credit. This is similar to the way collateral is deposited to DeFi loan services such as Aave or Compound, except that no separate receipt token is issued (such as cDAI or aETH). A small fee (such as 0.3%) is charged for this service. (Implying the user can mint 99.7% of their deposited balance into new eNFTs.)

   ==> *So far, nothing is private.*

2. **(Mint)** The user now mints one or more eNFTs (encrypted Non-Fungible Tokens), using their recorded balance credit as source funding. The Enshroud smart contract mints the eNFT(s) (each with a unique ID) to the user's designated address(es), and makes a matching debit against their balance credit.

   ==> *Because these eNFTs are encrypted, only the owner(s) can see how much each one is for.*
   ==> *Although, obviously, the sum of the newly minted eNFTs must equal the balance debited. For this reason, eNFTs minted to others are less private than when making a Spend.*

3. **(Spend)** The user can now make spends to other wallet addresses, using owned eNFTs as the funding inputs, and designating explicit amounts and assets to be paid to each payee address. The Enshroud smart contract will de-mint (burn) all input eNFTs, minting output eNFTs of equal value. If there is any unspent input value leftover, the user will receive a fresh eNFT back as change. (Account balance credit is neither increased nor decreased in a spend.)

   ==> *Only the recipients/owners can view the amounts (or asset types) of their eNFTs.*
   ==> *Transaction receipt data for the payer's transaction will log the list of TransferSingle events which minted/burned eNFTs associated with this spend, revealing which*

addresses were paid. However the values *are opaque. (Note this is the* opposite *of TC.)*
==> All mint events are transfers from 0x0 (the smart contract). All burn events appear as transfers to 0x0. Therefore any given eNFT is never transferred between addresses.
==> Asset types can be mixed and matched in the same spend, as long as valid inputs exist.
==> It costs recipients no gas to receive eNFTs. Therefore they can be sent to empty addresses, just like spends of ERC-20s or ordinary NFTs.

The deposited value may then circulate *indefinitely* among wallet addresses in eNFT form. The *generation* (distance from an on-chain deposit) of an eNFT is equal to the lowest generation of the inputs burned to mint it, plus one. As value circulates opaquely among addresses, the generation increases. While no fees are charged by Enshroud for spends, spenders pay all gas.

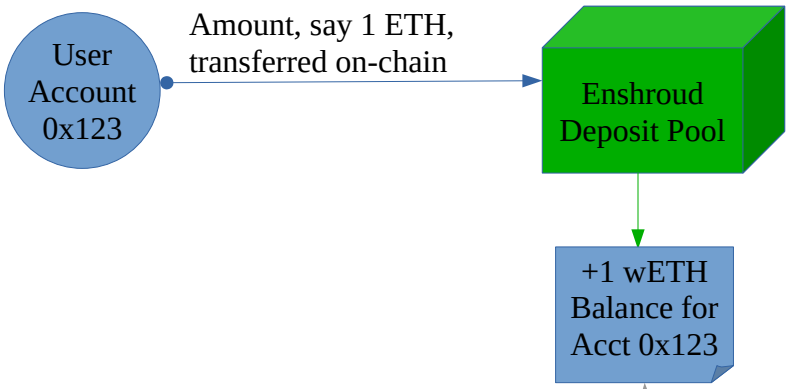
==> It is permitted to make spends to oneself, by including one's own address as a payee.
==> eNFTs can be issued for 0, either to send a message (via the memo line) or to generate additional ambiguity. However each additional eNFT slightly raises the gas cost.
==> Up to 20 eNFTs may be minted, or burned, in a single spend, constrained only by gas.

4. **(Burn aka Redeem/Withdraw)** A recipient user account which owns one or more eNFTs can redeem them at any time by submitting them to the Enshroud smart contract. The redeemed eNFTs are burnt and a matching on-chain spend of the underlying token asset is made to their owning account. As with minting, a modest fee is charged by the smart contract, which is deducted from the withdrawal spend performed.

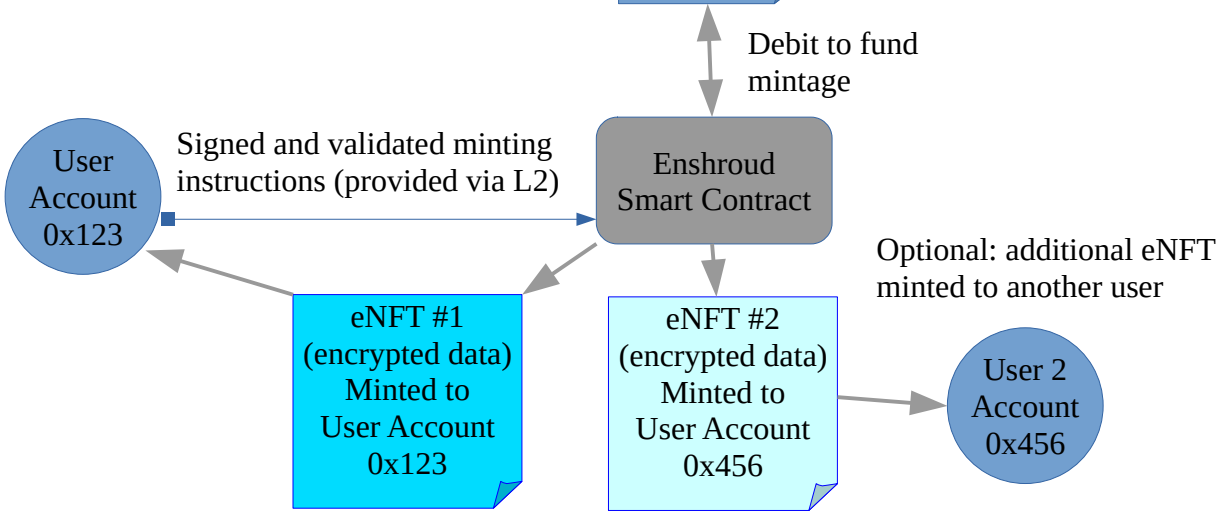
   ==> Necessarily, the redeemed eNFTs must have equaled the amount withdrawn, less change. Therefore the act of redemption "un-enshrouds" the value of the input eNFTs.

These four basic steps (Deposit, Mint, Spend, Burn) can be represented pictorially as follows:

Deposit

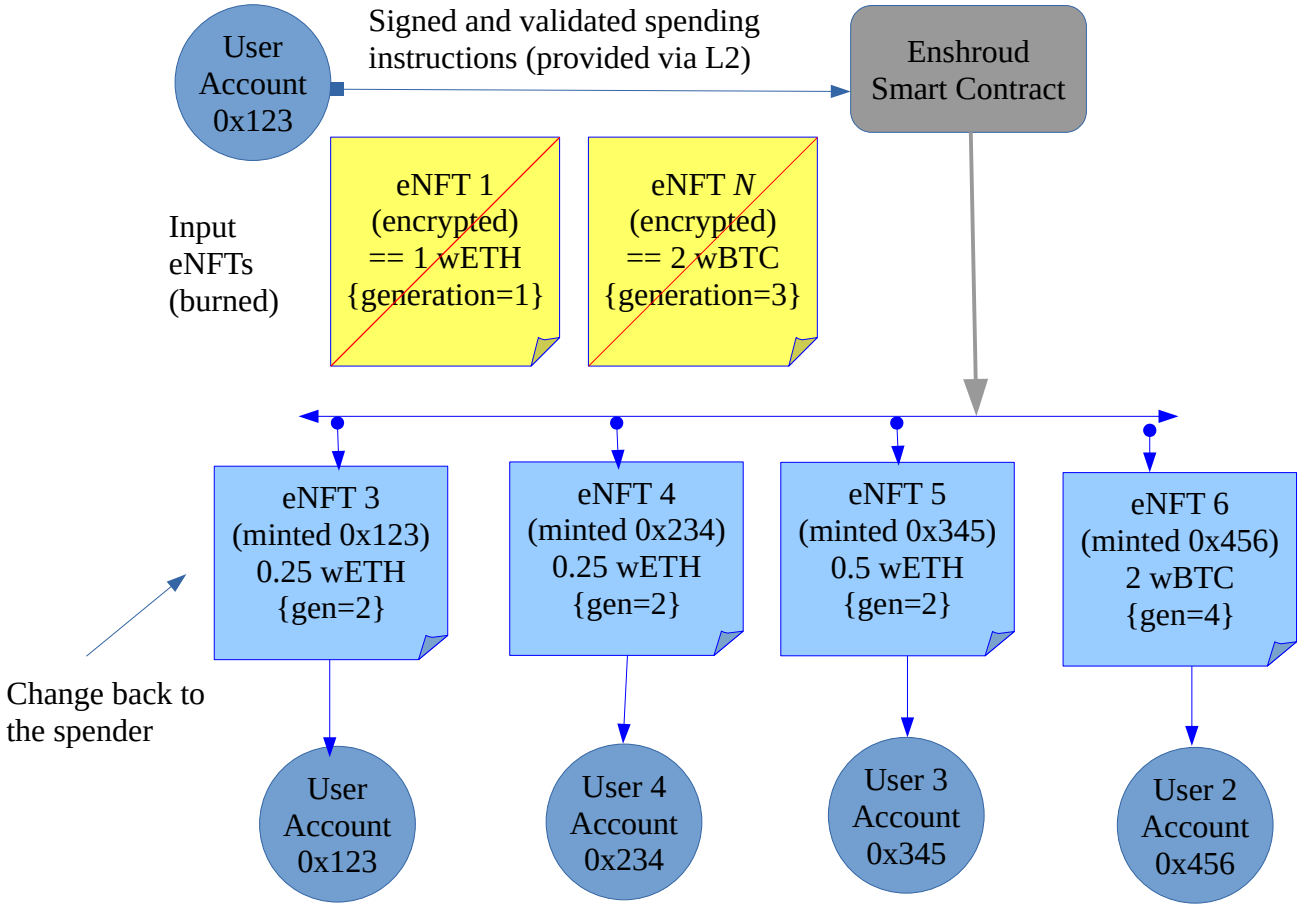


Mint



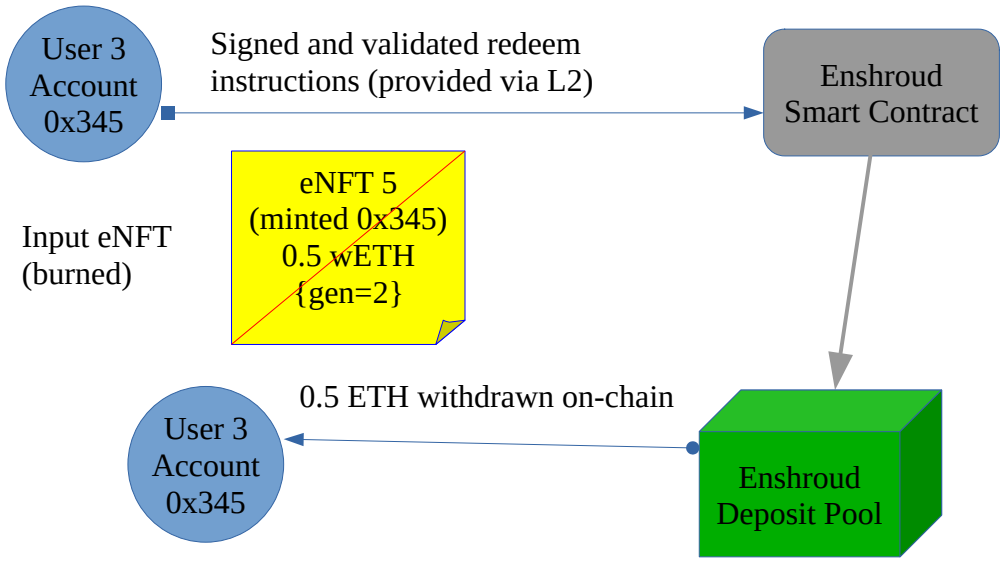
* Sum of eNFTs issued equals balance credit used.
* Asset types are the same as the balance.

Spend



==> In this single example transaction, User 0x123 spends an eNFT worth 1 wETH along with a second eNFT worth 2 wBTC. The ether is divided up between three recipients: User 0x234 (who gets 0.25 wETH), User 0x345 (who gets 0.5 wETH), and some change back for the unspent difference (0.25 wETH). User 0x456 receives 2.0 wrapped Bitcoin, which equaled the input amount, therefore there was no change back. The {generation} values of the new output eNFTs are incremented.

Redeem/Burn



A Few Notes

- All the instructions (calldata) said to be "provided via L2" are the result of a separate signed transaction previously made by the user (via the Enshroud dApp) with the Enshroud quasi-Layer 2 validators, which are called MVOs, for Metadata Validator Oracles. While the design of Enshroud's "Layer 2" is beyond the scope of this article, the function of the MVOs is to validate what the user is doing, such as guaranteeing that input eNFTs are valid and that outputs do not exceed inputs. While there is no actual Layer 2 blockchain, an *N of M* multi-signature proof-of-stake consensus mechanism is used to generate signed calldata for the smart contract methods, such that only (signed) hashed details are exposed in mempool calldata. Note that MVOs only perform text processing services, never have access to any coins, run only published open source code, and can be sliced for misbehavior. Their purpose is to preapprove user actions by signature so that the on-chain smart contract can trust hash-blinded user calldata.

- Encrypted signed receipts are also generated (by the MVOs) for both senders and recipients. Sender receipts show all payees, while payee receipts show only the sender (but not any other payees). Only the owners can download or decrypt their receipts, and – importantly – owners can permanently delete them at will. No receipt transaction history is ever stored on-chain.

- Also, note that the AES-256 keys used to encrypt eNFTs are purged from the keyservers after the eNFTs have been burned (i.e. used as inputs for a spend or redemption). Ditto the keys for

receipts after their deletion. This leaves the actual owners of the data as the *only* parties who might retain a copy of the keys required to successfully decrypt those eNFTs. All eNFTs are permanently appended to the blockchain's event log, but that's of no use to anyone without the individual unique keys necessary to decrypt them.

Similarities and Differences

We are now able to discuss in detail the comparison between Tornado Cash and Enshroud. Let's start with the similarities.

How Enshroud and TC are Similar

- Both systems are 100% non-custodial. That is, no actor within TC or Enshroud ever has control over spending deposited coins (or eNFTs). Every funds transaction is submitted on-chain by a user's wallet.

- Writers of open source software have no ability to control the future use of their software, or to limit, screen or otherwise restrict access by any particular person or account address.

- Blockchains supported are determined by the project (DAO) via smart contract deployment.

- The dApp client is hosted using IPFS, effectively restricting access to power Web3 users in exchange for censorship resistance.

How Enshroud and TC are Different

- TC breaks the link between the depositor and the withdrawer, with a probability of success related to the amount of parallel activity by other users during the deposit period. Enshroud obscures but does not break the metadata linkage between payer and payee; however it conceals the details of the transaction from observers 100% of the time. This is the most important difference, reflecting a fundamentally different aim. *TC acts as a coin sterilizing service; while Enshroud provides encrypted spends of coins between addresses.*

- TC amounts are quantized to match predetermined pool amount thresholds; Enshroud amounts are variable and set by the users. (But note gas costs will dictate practical minimums.)

- TC deposits are for specific durations; Enshroud eNFTs can be held indefinitely after minting, or spent immediately. Given intelligent use, Enshroud's privacy does not depend on wait time.

- Assets supported on TC are determined by the DAO (operators), which needs to create explicit Deposit Pools for each such supported asset / quanta permutation. Enshroud allows users to deposit to its smart contract any ERC-20, ERC-777, or ERC-4626 compatible asset, in variable amounts. (Effectively this means that the EnshroudProtocol smart contract is itself the Deposit Pool for all suitable assets.)

- Reclaims of TC deposits require gas; Enshroud eNFTs can be spent to empty addresses. (Although spend/burn operations will require gas.) Therefore, there are no Relayers in Enshroud. (This does not prevent *ad hoc* relayers from offering their services.)

- Enshroud eNFTs once minted can be circulated among users any number of times; TC deposits can be "spent" only once, directly to the withdrawing address. Enshroud thus creates a long-term pool of blinded circulating value, albeit between known addresses.

- Enshroud has a quasi-Layer 2 system which operates off-chain on a VPN, with all access to it made through encrypted protocols (ECIES-based, rather than https); TC is on-chain only.

- A TC user is probably transferring coins to another address they control. An Enshroud user can certainly do that, but is more likely to transfer eNFTs to other independent parties.

- Enshroud eNFTs can contain "memo" lines (up to 1024 characters), and can even be issued for 0 amounts. This makes Enshroud a literal secure wallet messaging system in addition to a private value transfer mechanism.

- TC offers generation of a PDF summary of a transaction, using its Compliance Tool. Enshroud creates encrypted receipts for participants, for Mint, Spend, and Burn operations. These can only be accessed by the owning addresses via a download request made to the MVO layer. Receipts embody transaction history to assist with bookkeeping, and can be downloaded in clear text. They also indicate to eNFT recipients which address sent them that eNFT, and serve as irrefutable proof-of-payment (because they are signed by MVOs). Receipts can be deleted permanently at user discretion and their associated AES-256 encryption keys destroyed.

- Enshroud purges its ephemeral AES keys when they are no longer required; TC does not encrypt anything.

Summary:

Design Feature	*Tornado Cash*	*Enshroud*
Address linkage in metadata	None (link broken)	From payer only
Spend amounts / assets encrypted	No	Yes
Deposit durations	Fixed term	Variable at user discretion
Relayers needed	Yes	No
Circulation limit	Single iteration	No iteration limit
Secure protocol for sensitive data	No	Yes
Attached memo message	No	Yes
Encrypted transaction receipts	Yes, but not encrypted	Yes
Benefits require active community	Yes	Not required, but enhanced
Amounts	Fixed according to pool size	Variable
Ephemeral encryption keys	N/A	Yes

Conclusion

It would be mistaken to assert that TC and Enshroud are truly competitors. They were designed to solve very different problems, and in fact it's easy to imagine how they could work synergistically. Enshroud addresses a distinct need which is not currently offered in the PrivFi sector, probably because no one has believed it was possible: making private encrypted payments on a public blockchain.

The default position of financial regulators and law enforcement has become that their right to surveillance over all payments is absolute, and that no one has any right whatsoever to financial privacy in the realm of electronic transactions. This of course is false; indeed on the contrary financial privacy should be absolute in the absence of prior suspicion based on probable cause. And in such cases, the target of any investigation should be the individuals themselves, by means of following due process, not blanket monitoring conducted through third parties (typically with a prohibition on disclosure, aka a gag order). Enshroud can help to right this imbalance, by making the participants in a transaction literally the only ones who can supply any information concerning it.

Blockchains are just a specialized form of messaging system. Therefore all blockchain transactions are speech, which sometimes happen to behave something like money. Speech that is math is still speech.

Enshroud provides a software service which encrypts payments much like PGP encrypts email. It is a tool that empowers its users. As with email encryption, its use is entirely in the hands of its users.

It will of course be said that Enshroud enables criminality. But actually it works to prevent criminal behavior – by governments. Blanket suspicionless surveillance is illegal, since by its very nature it violates the highest law of the land, the Bill of Rights. (Most countries have one.) Widespread oppression of the innocent is not an acceptable cost of doing business to apprehend the guilty. Tyranny is a serious crime, so like any encryption tool, in a sense Enshroud is really about the enforcement of laws against tyranny and violations of privacy generally.

We hope this essay has assisted you in understanding what Enshroud was designed to accomplish. Thank you for reading!

– The Enshroud Team