

Tornado Cash

Using SNARKs for Privacy and Scalability

Saravanan Vijayakumaran

Department of Electrical Engineering
Indian Institute of Technology Bombay

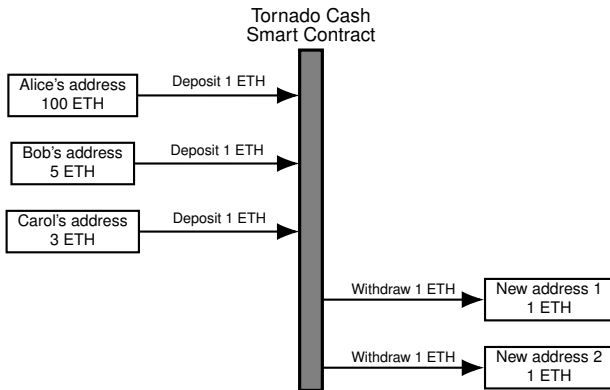
April 1, 2024

Motivation

- Consider the following scenario
 - You have 100 ETH stored in a self-custodial wallet
 - You take your family on a vacation to an exotic country
 - The hotel accepts ETH as a mode of payment
 - You pay the room rent of 1 ETH while checking in
 - The front desk clerk notices that you love your family and that your ETH address has 99 ETH
 - He has friends in the kidnapping industry
- How can you prevent leaking the total amount of ETH you hold?
 - **Option A:** You could store your ETH on an exchange and pay using their interface.
 - You risk losing funds due to exchange hacks
 - Hackers can steal customer data and sell it to their kidnapper friends
 - **Option B:** You could send 1 ETH to a fresh address from your 100 ETH address and use that to pay the room rent
 - Now suppose you decide to extend your stay
 - You make another 1 ETH transfer from your main ETH address
 - The clerk can now infer that you control a large amount of ETH
- Tornado Cash is a better **Option B**
 - It is a smart contract on Ethereum which implements a **mixer**

Tornado Cash Overview

Pre-Nova Version



- Desired functionality
 - **Soundness**
 - Only past depositors should be able to withdraw
 - No double withdrawal (only one withdrawal per deposit)
 - **Privacy:** A withdrawal should not be linkable to a particular past deposit

Deposit Workflow (1/2)

Choose amount and chain

The image displays two screenshots of the tornado.cash web application interface, illustrating the deposit workflow.

Top Screenshot: The user is on the "Deposit" tab. The "Token" dropdown is set to "ETH". The "Amount" slider is positioned at "1 ETH". A green "Deposit" button is visible. On the right, the "Statistics" section shows "Anonymity set 1" and "4884 equal user deposits". A table of "Latest deposits" lists transactions with IDs (e.g., 4884, 4883) and timestamps (e.g., 13 minutes ago, 3 hours ago). The user's IP address is shown as "193.21.127.60, Mumbai, IN".

Bottom Screenshot: The user is still on the "Deposit" tab, but a "Change network" modal is open. The modal lists various blockchain networks with radio buttons for selection:

- ☐ Ethereum Mainnet
- ☐ Binance Smart Chain
- ☐ xDAI Chain
- ☐ Polygon (Matic) Network
- ☐ Arbitrum One
- ☐ Avalanche Mainnet
- ☒ G6 Ethereum Goerli

The background interface remains the same, showing the "Deposit" form and the "Statistics" section.

Deposit Workflow (2/2)

Connect wallet, save note, and deposit

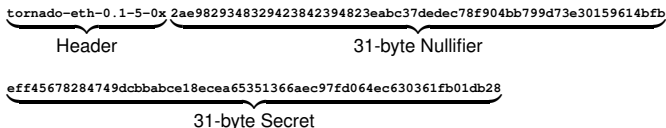
The image displays two screenshots of the Tornado.cash web application interface, illustrating the deposit workflow.

Top Screenshot: The "Deposit" tab is active. The "Token" dropdown is set to "ETH". The "Amount" slider is positioned at 1 ETH. A "Deposit" button is visible. The "Statistics" section shows "Anonymity set" as 4884 equal user deposits and a list of "Latest deposits" with timestamps.

Bottom Screenshot: The "Deposit" tab is active. A modal window titled "Your private note" is displayed, providing instructions on how to backup the note and save it as a file. The modal includes a checkbox labeled "I backed up the note" and a "Send Deposit" button. The background shows the "Deposit" form with the "Token" set to "ETH" and the "Amount" slider at 1 ETH.

Deposit Steps (1/2)

- Anatomy of a Tornado Cash private note



- The 62 bytes in the nullifier and secret are randomly generated on the user's computer
- A **commitment** (Pedersen hash of the 62 bytes) is calculated and submitted to the contract

$$\text{Pedersen hash of bitstring } b_1 b_2 \dots b_n = g_1^{b_1} g_2^{b_2} \dots g_n^{b_n}.$$

- Contract checks that `_commitment` has not been seen before

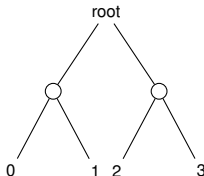
```
mapping(bytes32 => bool) public commitments;  
// <snip>  
require(!commitments[_commitment], "The commitment has been submitted");
```

Deposit Steps (2/2)

- Contract inserts `_commitment` into a Merkle tree

```
uint32 insertedIndex = _insert(_commitment);
```

- Tree has 20 levels
- `insertedIndex` is the index of new leaf



- No leaf deletions allowed \Rightarrow Maximum of 2^{20} deposits
- Stores the fact that `_commitment` has been seen

```
commitments[_commitment] = true;
```
- Checks that ETH being sent equals contract denomination


```
require(msg.value == denomination, "Please send 1 ETH with transaction");
```
- Emits an event

```
emit Deposit(_commitment, insertedIndex, block.timestamp);
```



Withdrawal Workflow (1/3)

Enter note string and recipient address

https://app.tornado.cash


 **tornado**

[Airdrop](#) [Mining](#) [Voting](#) [Compliance](#) [Docs](#)

[G0](#) [Goerli](#)   [Settings](#)

Deposit

Withdraw

Note 

1f67d7af3e1ba93af14b77de3b6d42ab375e0ea19f4

Amount1 ETH

Time passed2 hours

Subsequent deposits3 deposits

Recipient Address

Donate

0x3081b6978f7374e6427Db8b78dA2985b6C5D6483

Total

Gas Price4 Gwei

Network fee0.0022 gETH


Relayer fee0.000099 ETH

Total fee0.002299 ETH

Tokens to receive0.997701 ETH

Withdraw

Statistics1 ETH

Anonymity set 

4087 equal user deposits

Latest deposits

4087. an hour ago

4082. 11 hours ago

4086. an hour ago

4081. 21 hours ago

4085. an hour ago

4080. a day ago

4084. 2 hours ago

4079. a day ago

4083. 5 hours ago

4078. a day ago

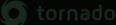
Your IP 103.21.127.60, Mumbai, IN

eth-1.tornadocash.eth

Withdrawal Workflow (2/3)

Choose relay

https://app.tornado.cash



Airdrop

Mining

Voting

Compliance

Docs

Gö

Goerli

Settings

Deposit

Note

1f67d7af3e1ba93af14b77de3b

Amount

Time passed

Subsequent deposits

Recipient Address

0x3081b6978f7374e64270b8b7

Total

Tokens to receive

Withdraw

eth-1.tornadocash.eth

Withdrawal settings

Relayer

Wallet

Relayer

goerli-v2.tornadosolutions.eth

Relayer fee

0.1%

Relayer status: OK

Total

Gas Price

4 Gwei

Network fee

0.0022 gETH

Relayer fee

0.001 ETH

Total fee

0.0032 ETH

Tokens to receive

0.9968 ETH

Reset

Save

osits

4082, 11 hours ago

4081, 21 hours ago

4080, a day ago

4079, a day ago

4078, a day ago

Your IP 103.21.127.68, Mumbai, IN

Withdrawal Workflow (2/3)

Choose relay

The screenshot shows the Tornado Cash web interface. A modal window titled "Withdrawal settings" is open, displaying a list of relays. The modal has two tabs: "Relayer" and "Wallet". The "Relayer" tab is active, showing a list of relays with their respective fees. The relay "goerli-v2.tornadosolutions.eth" is highlighted in green, and its fee of 0.1% is also highlighted. The background shows the "Deposit" section of the interface, including fields for Note, Amount, Time passed, Subsequent deposits, Recipient Address, Total, and Tokens to receive. There is also a "Withdraw" button and a "Filter by" section with options for ETH, BAL, USDT, and USDC.

Withdrawal settings

Relayer

Relayer

- goerli-v2.tornadosolutions.eth
- goerli-v2.poanet.eth - 0.05%
- goerli.v2.odanrot.eth - 0.01%
- goerli-v2.releth.eth - 0.05%
- goerli-v2.relaymy.eth - 0.05%
- goerli-v2.gaasservices.eth - 0.05%
- v2.goerli.thewizardseye.eth - 0.01%
- goerli-v2.reasoned.eth - 0.1%
- goerli.t-relay.eth - 0.045%
- goerli-v2.therelayer.eth - 0.03%
- goerli.relayer-service.eth - 0.05%
- goerli-v2.tornadosolutions.eth - 0.1%
- goerli-v2.torn.eth - 0.0099%
- Custom


Withdrawal Workflow (2/3)

Choose wallet if you have an unlinkable address with ETH

https://app.tornado.cash

tornado Airdrop Mining Voting Compliance Docs G&S Goerli Settings

Deposit

Note 

1f67d7af1e1ba93af14b77de3b...

Amount

Time passed

Subsequent deposits

Recipient Address

0x3081b6978f7374ee427Db8b7...

Total

Tokens to receive 1 ETH

Withdraw

eth.tornado.cash

Withdrawal settings

Relayer **Wallet**

Make sure that gETH used to pay for the gas fee is not linkable to ANY of your addresses. Otherwise, the anonymity of the withdrawal will be compromised. We recommend using a Relayer instead.

Total

Tokens to receive 1 ETH

Reset Save

4082, 11 hours ago

4081, 21 hours ago

4080, a day ago

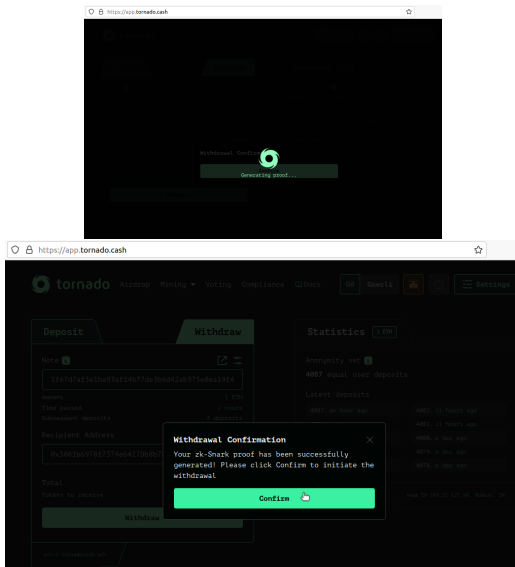
4079, a day ago

4078, a day ago

Your IP: 193.21.127.66, Mumbai, IN

Withdrawal Workflow (3/3)

Generate proof and confirm withdrawal



Withdrawal Steps (1/2)

- Recall our requirements
 - **Soundness**
 - Only past depositors should be able to withdraw
 - No double withdrawal (only one withdrawal per deposit)
 - **Privacy:** A withdrawal should not be linkable to a particular past deposit
- The `withdraw` method is executed

```
function withdraw(  
    bytes calldata _proof,  
    bytes32 _root,  
    bytes32 _nullifierHash,  
    address payable _recipient,  
    address payable _relayer,  
    uint256 _fee  
    // <snip>  
)
```

- `_proof` is a SNARK proof for the following statement:
I know the secret and nullifier for a commitment which is included in the Merkle tree with root `_root`.
Furthermore, `_nullifierHash` is the Pedersen hash of the commitment's nullifier.

Withdrawal Steps (2/2)

- Contract checks that `_nullifierHash` has not been seen before.

```
mapping(bytes32 => bool) public nullifierHashes;  
// <snip>  
require(!nullifierHashes[_nullifierHash], "Note already spent");
```

This prevents double withdrawal

- Checks that `_root` is any of the last 100 Merkle roots
- It then verifies the SNARK proof on-chain

```
require(  
    verifier.verifyProof(_proof,  
        [uint256(_root), uint256(_nullifierHash), ...]  
    ),  
    "Invalid withdraw proof"  
);
```

- Stores the fact that `_nullifierHash` has been seen
`nullifierHashes[_nullifierHash] = true;`
- Sends relevant amounts to `_recipient` and `_relayer`
`_recipient.call.value(denomination - _fee)("");`
`_relayer.call.value(_fee)("");`
- The SNARK proof also “signs” the `_recipient`, `_relayer`, `_fee` fields to prevent tampering
- The verifier contract is generated using the `circom` compiler.

withdraw.circom

```
template Withdraw(levels) {
    signal input root;
    signal input nullifierHash;
    signal input recipient; // not taking part in any computations
    signal input relayer; // not taking part in any computations
    signal input fee; // not taking part in any computations
    signal private input nullifier;
    signal private input secret;
    signal private input pathElements[levels];
    signal private input pathIndices[levels];

    component hasher = CommitmentHasher();
    hasher.nullifier <== nullifier;
    hasher.secret <== secret;
    hasher.nullifierHash === nullifierHash;

    component tree = MerkleTreeChecker(levels);
    tree.leaf <== hasher.commitment;
    tree.root <== root;
    for (var i = 0; i < levels; i++) {
        tree.pathElements[i] <== pathElements[i];
        tree.pathIndices[i] <== pathIndices[i];
    }

    // Add hidden signals to make sure that tampering with recipient or fee will invalidate the snark proof
    // Most likely it is not required, but it's better to stay on the safe side and it only takes 2 constraints
    // Squares are used to prevent optimizer from removing those constraints
    signal recipientSquare;
    signal feeSquare;
    signal relayerSquare;
    recipientSquare <== recipient * recipient;
    feeSquare <== fee * fee;
    relayerSquare <== relayer * relayer;
}

component main = Withdraw(20);
```

OFAC Sanctions

- On Aug 8, 2022, the US Office of Foreign Assets Control placed Tornado Cash addresses on a sanction list
- US residents/businesses cannot interact with entities on the list
- Allegations include facilitating money laundering by ransomware operators and smart contract attackers
- Github removed source repos and three contributors had Github accounts suspended
- Due to the efforts of Prof. Matthew Green and EFF, OFAC allowed use of code for educational purposes
- Github repositories and accounts restored in 2023
- Developer Alexey Pertsev arrested in Netherlands in Aug 2022; released on bail in April 2023
- Developer Roman Storm arrested in US on Aug 23, 2023
- Pertsev's trial began on March 26, 2024. Verdict expected on May 14

References

- Tornado Cash App <https://tornado.ws/>
- Tornado Cash Docs <https://docs.tornado.ws/>
- Circom <https://docs.circom.io/>
- <https://github.com/tornadocash/tornado-core>
- EFF article on OFAC sanctions
- EFF update in April 2023

Thanks for your attention