CAPE System Components

Visualizing and understanding how it works A simplified view of the CAPE system appears below. The CAPE Wallet is used to construct a transaction and submit it to the CAPE Relayer. The Relayer pays the Ethereum gas fee to submit the transaction to the CAPE Ethereum smart contract.

Other components help broker information so that the wallet can construct valid transactions.

CAPE architecture From the user's perspective, the CAPE Wallet is the center of the universe. The wallet composes transactions using information from the Ethereum Query Service and Address Book. The wallet submits transactions to the Relayer that forwards them to the Smart Contracts. The Ethereum Query Service monitors events so that the wallet can use it to track transactions and the state of the system.

CAPE Smart Contract

The CAPE smart contract on Ethereum maintains the state of the CAPE system, enabling CAPE to utilize Ethereum's distributed consensus and to provide features to existing ERC-20 assets.

The CAPE smart contract allows bidirectional transfers of assets between Ethereum and the CAPE system. An ERC-20 token transfer to the contract triggers the creation of an asset record inside CAPE. Converting CAPE assets back into ERC-20 tokens entails burning the asset record and unlocking the original ERC-20 token in the contract.

The CAPE smart contract leverages Ethereum to provide distributed consensus to the CAPE system. The contract supports the following:

- Verify transactions in each CAPE block and updates the CAPE state
- Sponsor wrapped CAPE asset types with corresponding ERC-20 tokens, policy, and asset type identifier
- Wrap ERC-20 tokens into CAPE asset records
- Process "burn" transactions enabling ERC-20 tokens to be unwrapped from CAPE asset records
- Generate CAPE "fee" asset records for the CAPE Faucet
-

Records Merkle Tree

The records Merkle tree stores every record commitment produced in the system. The CAPE smart contract stores a small digest of this set called theMerkle root . Users consuming inputs can prove membership of their records against the Merkle root in the contract, effectively proving that the input records already live in the system. For privacy, this proof is presented in zero-knowledge, hence observers and validators do not learn which record is being consumed. Moreover, verification does not reveal which record commitment is being spent.

The CAPE smart contract also stores the membership proof of the latest added record commitment, calledMerkle tree frontier. This allows the contract itself to update the Merkle root and the frontier after a new record commitment has been added. Otherwise, users would need to know the latest frontier just before they submit a transaction.

ERC-20 Smart Contract

The CAPE smart contract is compatible with any Ethereum smart contract following the ERC-20 token standard.

CAPE Wallet

The CAPE Wallet is the primary user-facing component of the system. It enables users to generate and manage keys, synchronize with the state of the CAPE system, and build and submit transactions. The wallet is intended to run client-side as a native executable or Docker container, and exposes a REST API for use by graphical user interfaces.

The wallet enables the following functionality:

- View balances and CAPE transactions
- Define new asset types, both domestic CAPE assets and ERC-20 wrapped asset types, with their associated viewing and freezing policies
- Wrap ERC-20 tokens into CAPE asset records.
- Unwrap CAPE assets that have underlying ERC-20 tokens by burning the asset record and unlocking the original ERC-20
- View any asset records for which the user has the corresponding viewing keys
- Freeze any asset records for which the user has the corresponding freezing keys
- Generate and manage the necessary keys to perform the aforementioned actions
-

The wallet follows events originating from the Ethereum ledger in order to discover asset records which it has received. When a transaction is committed, the wallet receivesowner memos associated with each output record of the transaction. The owner memos, stored in the smart contract calldata and broadcast by the Ethereum query service, contain the encrypted details of the records such as the asset type and the amount. They are created and encrypted by the sender using the recipient’s public encryption key. If a wallet is a recipient of a transaction, it will be able to decrypt the memos and add the records to its collection. This enables the user to see what they received, and to later consume the asset record in a new CAPE transfer to another user.

The transaction workflow is designed to allow wallets to transfer asset records among themselves, using the validators' consensus as a source of truth but also publishing encrypted owner memos to communicate secret information with transaction recipients.

1. User provides the needed information (asset type, amount, and addresses) to build a transaction (wrap, unwrap, transfer, mint, or freeze).
2. Wallet generates the transaction note and using the provided information and information about the ledger state, obtained from theEthereum Query Service
3. .
4. Wallet generates the owner memos, encrypted using the recipients public encryption key obtained from theAddress Book
5. .
6. Wallet submits transaction and memos toRelayer
7. , which forwards them to the Smart Contract.
8. Ethereum miners execute the smart contract functionality to validate the transaction, update ledger state, and broadcast events.
9. Ethereum Query Service
10. receives events from the contract, collects them, and generates an event containing all newly committed transactions and owner memos.
11. All wallets receive the event from theEthereum Query Service
12. and do processing universal to all transactions (updating their local state, etc.). The receiver of the transaction decrypts the owner memos and adds the received records to their collection of owned records.
13.

CAPE Relayer

The purpose of the CAPE Relayer is to prevent Ethereum gas fees, which must be paid transparently by an Ethereum account, from being linked to CAPE users. Users submit CAPE transactions to the relayer, which then publishes the transaction on Ethereum, paying the necessary ETH gas fees. Therefore, the general public cannot link transactions to users.

The current implementation of the relayer is the simplest possible implementation; the relayer performs no validation of its own and immediately forwards transactions to Ethereum as soon as they are received.

Alternative implementations of CAPE relayers may maintain a lightweight state of the CAPE system, allowing the relayer to validate CAPE transactions locally before forwarding them to the CAPE contract on Ethereum. Moreover, the relayer may choose to batch CAPE transactions into blocks and submit them in a single Ethereum contract call, rather than forwarding transactions individually.

CAPE Address Book

The CAPE Address Book maps user addresses to encryption public keys. Users publish their encryption public keys here. This is done by submitting theirUserPubKey bundle (that includes both address and encryption public key) and signing the request using the user private key associated with the address. This way, it is infeasible to publish an invalid encryption key for a user.

The CAPE Address Book maps user addresses to public keys. This public mapping allows a sender to retrieve the encryption key associated with the recipient address. This is particularly important for the freezing feature: when the freezer releases records back to the original owners, it needs to know their encryption keys in order to produce owner memos for them.

CAPE Faucet

For the testnet deployment of CAPE, (non-value) CAPE testnet “fee” tokens can be obtained from the CAPE Faucet by providing a valid CAPE address.

Please note that this is a testnet token only, not for value transfer or swapping. There is no CAPE token sale or other distribution happening or currently planned.

For the CAPE testnet, CAPE testnet “fee” tokens can be obtained from a faucet. The faucet is a source of CAPE “fee” asset records that are needed to produce valid CAPE transactions. Though fee amounts default to zero, asset records with the CAPE fee asset type are still required for a transaction to be valid.

Under the hood, upon deployment of the CAPE contract to testnet, an initial supply of CAPE “fee” tokens is created under a hard-coded owner address. These tokens are used to fund the faucet.

CAPE Ethereum Query Service

The CAPE Ethereum Query Service (EQS) supplies information to the CAPE Wallet. This information enables users to see their asset records and build transactions. The query server enables a wallet that has been offline to discover what transpired while it was offline.

The CAPE Ethereum Query Service provides information for transaction building and transaction history Specifically, the EQS provides

- CAPE transactions
- Records Merkle tree proofs
- Set of nullifiers – See the CAPE Relayer above for a description of the nullifier set
- Recent roots of the asset records Merkle tree and a representation of time
- Ethereum events related to CAPE transaction processing
-

Previous Lifecycle of a CAPE Transaction Next Components Interactions Last updated1 year ago On this page *CAPE Smart Contract * Records Merkle Tree * ERC-20 Smart Contract * CAPE Wallet * CAPE Relayer * CAPE Address Book * CAPE Faucet * CAPE Ethereum Query Service