# Zswap

> INFO
>
> The details of Midnight's native currency implementation are not yet stable and will undergo further revisions. The performance of basic operations has not been optimized at this time.

Zswap[^1] is a shielded token mechanism, based on Zerocash[^2], extended with *native token support* and *atomic swaps*. Zswap's basic component is an offer, which conceptually is a set of inputs and outputs. In this matter, it matches the UTXO model, although the set of unspent transactions itself is not computable due to the inability to link matching inputs and outputs, a property inherited from Zerocash.

This section describes a slight variation of Zswap used in Midnight that permits contracts to hold funds.

## Offers

A Zswap offer consists of four elements:

- a set of input coins (also called 'spends')
- a set of output coins
- a set of transient coins
- a balance vector.

Transient coins are coins that are both created and spent in the same transaction. This may seem superfluous, but it extends the ability for contracts to manage coins. Conceptually, this is an output immediately followed by an input, with the sole distinction that the input spends from a locally created coin commitment set, as opposed to the global one, to prevent index collisions.

The balance vector is a vector of the total value of this offer. Its dimensions are all possible token types, with each dimension carrying its own value. An input of a given type counts positively towards this vector and negatively towards an output. A balance vector is considered *balanced* if, for all dimensions, it is non-negative. Typically, it is *adjusted* before checking for balance, to account for token mints and fee deductions.

## Outputs

A Zswap output creates a new coin and places a corresponding *commitment* in a global Merkle tree. It consists of:

- the commitment itself
- a multi-base Pedersen commitment to the type/value vector
- an optional contract address, iff (if and only if) this output is targeted at a contract
- an optional ciphertext, if the output is toward a user that must receive it
- a zero-knowledge proof that the former are correct with respect to each other.

Outputs are valid if their zero-knowledge proof is verified.

## Inputs

A Zswap input spends an existing coin, by referencing (without revealing) its original commitment in the global Merkle tree and producing a corresponding (but unlinkable) *nullifier*. It consists of:

- the nullifier itself
- a multi-base Pedersen commitment to the type/value vector

- an optional contract address, iff the output is targeted at a contract
- a Merkle tree of a tree containing the commitment corresponding to the nullifier
- a zero-knowledge proof that the former are correct with respect to each other.

Inputs are valid iff the zero-knowledge proof verifies *and* the Merkle tree root is in the set of past roots.

## Token types

A token type in Midnight is a 256-bit collision-resistant hash output or the pre-defined zero value, which represents the native token. Users can issue their own tokens from contracts, with these token types being derived as a hash of the contract's address and a domain-separator given by the user.

[^1] Engelmann, F., Kerber, T., Kohlweiss, M., & Volkhov, M. 2022. Zswap: zk-SNARK based non-interactive multi-asset swaps. *Proceedings on Privacy Enhancing Technologies (PoPETs)* 4 (2022), 507-527. https://eprint.iacr.org/2022/1002.pdf

[^2] Ben-Sasson, E., Chiesa, A. Garman, C., Green, M., Miers, I., Tromer, E., & Virza, M. 2014. Zerocash: Decentralized Anonymous Payments from Bitcoin. *2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, May 18-21, 2014*, 459-474. https://eprint.iacr.org/2014/349.pdf