Transaction semantics

Ledger states

Midnight's ledger consists of two items:

- Zswap's state
  - a Merkle tree of coin commitments
  - an index to the first free slot of the coin commitment tree
  - a set of nullifiers
  - a set of valid past Merkle tree roots
- a map from contract addresses to contract states.

Contract state

A contract state consists of:

- an Impact state value
- a map of entry point names to operations, where entry points correspond to exported circuits in a contract.

A contract operation consists of a Snark verifier key that is used to validate contract calls made against this contract and entry poi

Transaction fallibility

Transactions execute in three stages:

- well-formedness check
- guaranteed phase
- fallible phase.

The well-formedness check is run without any state and checks the general integrity and consistency of the transaction. In contrast, both the guaranteed and fallible phases are run against the ledger state and either produce a new state or fail. If a transaction fails during the guaranteed phase, it is *not included in the ledger*. If it fails during the fallible phase, any effects of the guaranteed phase *still apply*, and the ledger will record the transaction as a *partial success*.

The fees for all phases of execution are collected in the *guaranteed* phase and are forfeited if a transaction fails in the fallible phase.

Well-formedness

The well-formedness check verifies that a transaction is in a canonical format, and that:

- all zero-knowledge proofs in Zswap offers can be verified
- the Schnorr proof in the contract section can verified
- the guaranteed offer is balanced with respect to the following adjustments:
  - subtraction of the fees of the entire transaction
  - addition of any mints performed in guaranteed transcripts
- the fallible offer is balanced with respect to the following adjustment:
  - Addition of any mints performed in fallible transcripts

- each contract-owned input or output is claimed exactly once by the same contract in the effects section of the transcript matching the fallibility of the offer it appears in
- any outputs claimed as being created by a contract in the effects section of a transcript are claimed at most once, and they appear in the offer matching the fallibility of the transcript
- any contract calls that are claimed in a transcript are present and claimed at most once
- if a contract call has both a guaranteed and fallible section, the fallible section starts with a `ckpt` operation.

Phase execution

Other than the notes in the transaction fallibility section, the guaranteed and fallible phases operate similarly, except that the following additional work is performed in the guaranteed phase:

- contract operations for all calls are looked up, and the zero-knowledge proofs are verified against them
- the fallible Zswap section is also applied during the guaranteed section, to ensure that it cannot invalidate the fallible section by itself. [^1]

Then:

1. The phase's Zswap offer is applied, by inserting new commitments into the Merkle tree and nullifiers into the nullifier set (aborting if they are already present), checking that the Merkle roots used are valid past roots (aborting otherwise), and updating the past roots set
2. The above additional checks for the guaranteed phase are performed, if applicable
3. For each contract call in sequence, the transcript relevant to this execution phase is applied
    i. The contract's current state is loaded
    ii. The context is set up from the transaction

iii. The Impact program is executed against the context, an empty effects set, the transcript program, and the declared gas limit, in verification mode

iv. The resulting effects are tested to be equal to the declared effects

v. The resulting state is stored as the contract's state, iff it is "strong".

[^1] This would permit invalidating any fallible section by merging with an invalid spend otherwise.