Sundae Protocol - Permissioned DeFi

Pi Lanningham
Sundae Labs
pi@sundae.fi

Jesse Anderson
Kora Labs
papa.goose@koralabs.io

Justin Newton
Netki
justin@netki.com

This Version: October 2024

Abstract

DeFi protocols and cryptocurrency ecosystems often struggles to attract liquidity, in large part because of the lack of regulatory clarity. Large institutions who wish to participate hesitate to do so because DeFi primitives are usually wholly permissionless. This puts them at regulatory risk, as their funds may be utilized in trades that fund illegal activities.

Sundae Labs, Kora Labs, and Netki are collaborating to solve this problem. This specification outlines 3 separate standards that will enable seamless permissioned DeFi on top of the by-default permissionless Sundae Protocol on the Cardano blockchain.

1. ADA Handle DID Resolution - allow a user to associate Decentralized Identifiers (DIDs) with their Cardano address
2. Permissioned Pools - allow liquidity pools that have extra configurable conditions attached to specific orders
3. Netki Integration - specify exactly how to utilize the above, along with Netki's compliance oracle infrastructure

Together, these three proposals allow the creation of "clean" DeFi pools on top of the Sundae Protocol.

1. Introduction

The goal of this specification is to outline how Sundae Labs, Kora Labs, and Netki are collaborating to allow permissioned and regulatory compliant “Clean” pools on Cardano, leveraging ADA Handles for easy DID Discovery, the SundaeSwap protocol for DeFi primitives, and Netki as a compliance oracle.

The main thesis of this work is that DeFi is unnecessarily closed off from institutional participation because of regulatory risk. A large entity may have very deep liquidity that they would be interested in deploying to DeFi, but are unable to do so because they cannot bear the risk of those funds being used for money laundering, terrorism financing, or any other unsavory activities.

In a classical finance setting, these actors would have the assurance that the financial institutions holding the funds and executing the trades have done their due diligence such as performing Know Your Customer (KYC) on each customer executing a trade.

We strongly believe that permissionless DeFi provides options to legitimate actors in the developing world, and have spent 3 years building exactly those primitives. However, we also believe that the landscape of DeFi can provide for all users, including those that want more assurance behind who they are transacting with. Decentralized Identity standards allow entities to exchange sensitive identity information in a decentralized setting, without unduly exposing those details to the broader watching world.

> ⓘ **Note**
>
> Originally, we had planned to include a proposal to extend CIP-30 to allow a dApp to communicate with a DID wallet. However, we discovered that this work is already under way via several great standards (such as this work by the Cardano Foundation), is auxiliary to the objective of the project, and didn’t make sense to duplicate that work.

2. ADA Handle DIDs

ADA Handles are a “human readable address” product built on Cardano. By holding a Cardano Native Token with a given name at a specific address, tooling such as wallets can allow users to type in a human readable name, and unambiguously resolve that to an address.

ADA Handles follow the CIP-68 standard to allow custom data to be associated with the ADA handle. For example, this capability is used today to specify a preferred background, profile picture, and highlight color to personalize your Handle, and dApps can match their theming to that personalization information.

All CIP-68 ADA Handles begin with an asset name prefix of `000de140`. There also exists a corresponding token with a prefix of `000643b0`, and the same suffix, that corresponds to the “reference token”.

The datum holding the reference token can be updated with a signature from the wallet holding the ADA handle to prove ownership, and a signature from Kora Labs to ensure that the structure of the datum stays well formed.

The format of that datum according to CIP-68 is:

```
big_int = int / big_uint / big_nint
big_uint = #6.2(bounded_bytes)
big_nint = #6.3(bounded_bytes)
metadata =
  { * metadata => metadata }
  / [ * metadata ]
  / big_int
  / bounded_bytes
version = int
extra = plutus_data
datum = #6.121([metadata, version, extra])
```

The purpose of `metadata` is to capture metadata about the NFT itself, while `extra` is arbitrary and can be determined by use case. ADA Handle has utilized the `extra` field for their personalization metadata. We propose standardizing on a `public_did` field added to this extra map in the case of ADA Handles. The public_did field will be a Concise Binary Object Representation (CBOR) map, where the keys represent a human readable label, and the values represent a W3C DID Identifier, according to this specification.

```
did = bounded_bytes, ; UTF-8
did_map =
  { * label => did }
metadata =
  {
    ; ...
    public_did: did_map
  }
```

One of these keys can be “default”, which should indicate the DID to select in non-interactive scenarios where the user cannot be prompted.

If a dApp has some ADA Handle, and wishes to resolve a users DID, it can follow these steps:

- Strip off the `000de140` prefix from the ADA Handle
- Prepend the `000643b0` prefix to obtain the reference token name
- Look up the UTxO holding the token with the same policy ID and the reference token name
- Read the attached Datum, and deserialize it according to the CIP-68 specification
- Read the `extra.public_did` field

- Select one of the DIDs
  - If noninteractive, and one of the keys is `default`, use this one
  - If noninteractive, and there is only one DID, use this one
  - If noninteractive, and there are multiple DIDs, behavior is dependent on your use case
  - If interactive, and there is only one DID, use this one
  - If interactive, and there are multiple DIDs, prompt the user for which one to use, using the keys as labels

From there, the dApp can use existing standards, such as DIDComm, KERI, or others to interact with the user and their identity.

3. Sundae Protocol Permissioned Pools

SundaeSwap v3 is a fast, decentralized AMM based exchange built by Sundae Labs.

We propose making the following updates to the Sundae v3 Protocol:

- Define a new pool type, `v3-permissioned`
- This new pool has a new `condition` property, which is of type `ScriptHash`
  - If the pool UTxO is spent with the `PoolScoop` redeemer, the `condition` must be present in the `withdrawals` of the transaction, effectively "blessing" the scoop.
  - To avoid locking user funds in the pool, a scoop that consists of only withdrawals is exempt from this condition.
- A list of `allowed_conditions` is added to the `settings` datum, utilizing the `extensions` field
  - Minting a new `v3-permissioned` pool requires that the `condition` be one of the conditions in the `allowed_conditions` in the `settings` datum.
- A new redeemer is added to `ManageRedeemer` to allow management of the condition.
  - If the pool UTxO is spent with the `Manage` redeemer, the `PoolManage` script must be present in the withdrawals of the transaction. (This is already the case.)
  - If the `PoolManage` script is invoked with the `ManageCondition` redeemer, the current condition must be present in the withdrawals, the new `condition` must be present in the `allowed_conditions` field in the settings datum.
- Validation of the following conditions are moved to a default condition script:
  - Checking that the transaction is signed by a scooper
  - Checking that the correct protocol fee is paid
  - Checking that the correct pool fee is paid

The above allows arbitrary additional logic to be layered on to the pool. Some examples of how this might be used:

- A pool that only allows trading from 9 to 5 to synchronize with an existing market

- A pool that only allows deposits or swaps if they bear a signed authorization token
- A pool that allows swaps to pay a lower protocol or pool fee if they hold a threshold of token or a membership NFT

4. Netki Compliant Pools

Netki is a company that provides compliance and identity verification services for Web3 contexts. We have been in discussions with them on a way to bring "clean" pools to SundaeSwap v3.

Based on those discussions, we propose the following scheme that builds on the standards in the previous sections:

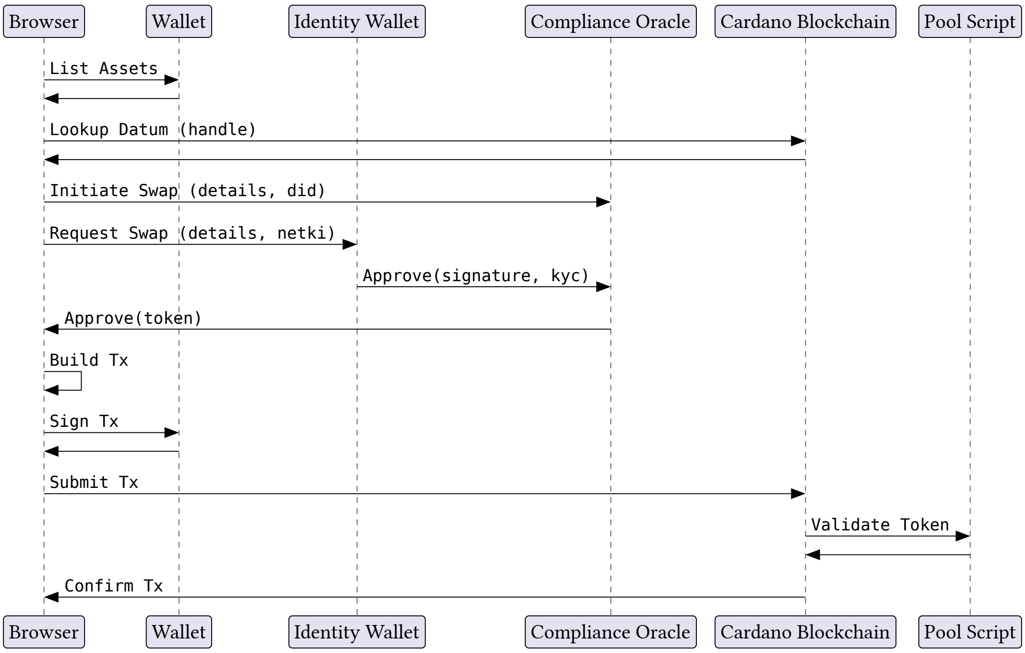
- Each pool will include, in the `extensions` field, a list of approved compliance oracles
  - This can be set when the pool is created
- Each order will include, in the `extra` field, a signed compliance token
  - The compliance token will consist of:
    - The users DID Identifier
    - The users public key
    - The destination address
    - Blake2b-256 hash of the cbor serialized details from the order
    - A valid range
    - The public key of the oracle
    - A signature from the compliance oracle for fields 1-6
  - The condition will validate that each order:
    - Has an attached compliance token
    - The `owner` of an order is the same as the public key in the token
    - The `destination` of an order is the same as the one in the token
    - The order details hash to the same hash from the token
    - The valid range of the transaction entirely contains the valid range from the token
    - The oracle public key is one of those listed in the approved oracles
    - The signature is valid

When a user wishes to perform a swap, the dApp then:

- Notifies Netki that a swap has been started for a specific DID
- Asks the identity wallet to show their KYC credential to Netki
  - This may also redirect to a flow asking the user to perform KYC with Netki to issue said credential
- Netki checks the configured compliance rules against that DID
- Netki notifies the dApp with an authorization token that can be included in the transaction
- The dApp builds the transaction, asks for a signature, and submits it to the blockchain

5. Conclusion

End to end, here is a sequence diagram that illustrates how the above three protocols enable a decentralized, compliant liquidity pool.



6. Glossary

***CBOR* – Concise Binary Object Representation**: A binary encoding format used heavily by the Cardano blockchain 3

***DID* – Decentralized Identifier**: A standard for creating unique identifiers for entities in a decentralized setting 1, 2, 5

***KYC* – Know Your Customer**: A series of steps and data collection policies that one entity might employ so it can transact with another and verify the identify of a customer 2, 5

Netki: A compliance oracle solution for web3, that can check the compliance of a transaction against a flexible and configurable ruleset. 5