Ark - Bitcoin L2

#BitcoinL2

https://ark-protocol.org/

おそらく元になっているプロトコルのサイト・・・？

Arkの詳細と思われるPDF ark.pdf はArk LabsのURLにある



https://arklabs.xyz/

Arkプロトコルを使ったプラットフォーム Arkade を提供する

`arkd` (Arkプロトコル準拠のASPサーバ)、 `ark-client` (ユーザ向けクライアント)、Fulmine Wallet(リファレンスウォレット)

https://arkadeos.com/

https://docs.arklabs.xyz/ark/ (黒い方のページ)

https://docs.arkadeos.com/ (白い方のページ)

「Ask AI」があって、けっこう便利

https://blog.arklabs.xyz/

ロードマップ

https://github.com/arkade-os

https://github.com/ArkLabsHQ

ChatGPT

second

https://second.tech/

https://docs.second.tech/ark-protocol/intro/ ArkadeではないArk実装をしているSecondという組織

動作

Ark v0.6.x

Ark v0.7.x

Telegramのピン留めページ

👾 Welcome to the Arkade Community!

🔗 Key links

- 🌐 Website – https://arkadeos.com
- 📜 Vision Paper – https://arkadeos.com/vision
- 📚 Developer Docs – https://docs.arkadeos.com
- 🗂️ GitHub – https://github.com/arkade-os
- 🐦 Twitter / X – https://x.com/arkade_os
- ✈️ Telegram – https://t.me/arkade_os

🧩 Ark protocol

- ✨ Protocol Explainer – https://docs.arklabs.xyz/ark
- 🔬 Deep Dive – https://docs.arklabs.xyz/ark/deep-dive/intro
- 📄 Ark Litepaper (TU Wien collab) – https://docs.arklabs.xyz/ark.pdf

---

概要

Ark Labsという会社としては Arkade という商用サービスをやっていて、そのオープンソースサーバー(`arkd`)を立ち上げてはいるが、L2自体を運用していない。
プラットフォームを提供するという立場。
"Arkade OS"というのはコンセプトというかブランドというかビジョンというかの名前で、L2 の Arkade の上でアプリを動かすという見方をすると Arkade は OSだよね、という意味合い。
Second社という元Blockstreamチームが立ち上げた会社が実装した Arkもある。たぶんArkadeと互換性はない。

「exitの保証は強いが、L2転送の即時確定性は弱い」というセキュリティモデル

L1 に展開するトランザクションは VTXO の再生成 or exit での返金だけで L2での送受金は含まれない。

VTXO で送金している間、ASP運用者と送金者が結託すると二重支払いできてしまう。

CoinJoinライクな構造とVTXO匿名性で、オンチェーンから送金関係を読み取れない設計

Commitment Transaction はそのときに処理する VTXO をまとめて処理するトランザクションになるし、出金とVTXO再生成が混ざることもあり、誰が誰に送ったかわかりにくくなる。

regtestでの接続例

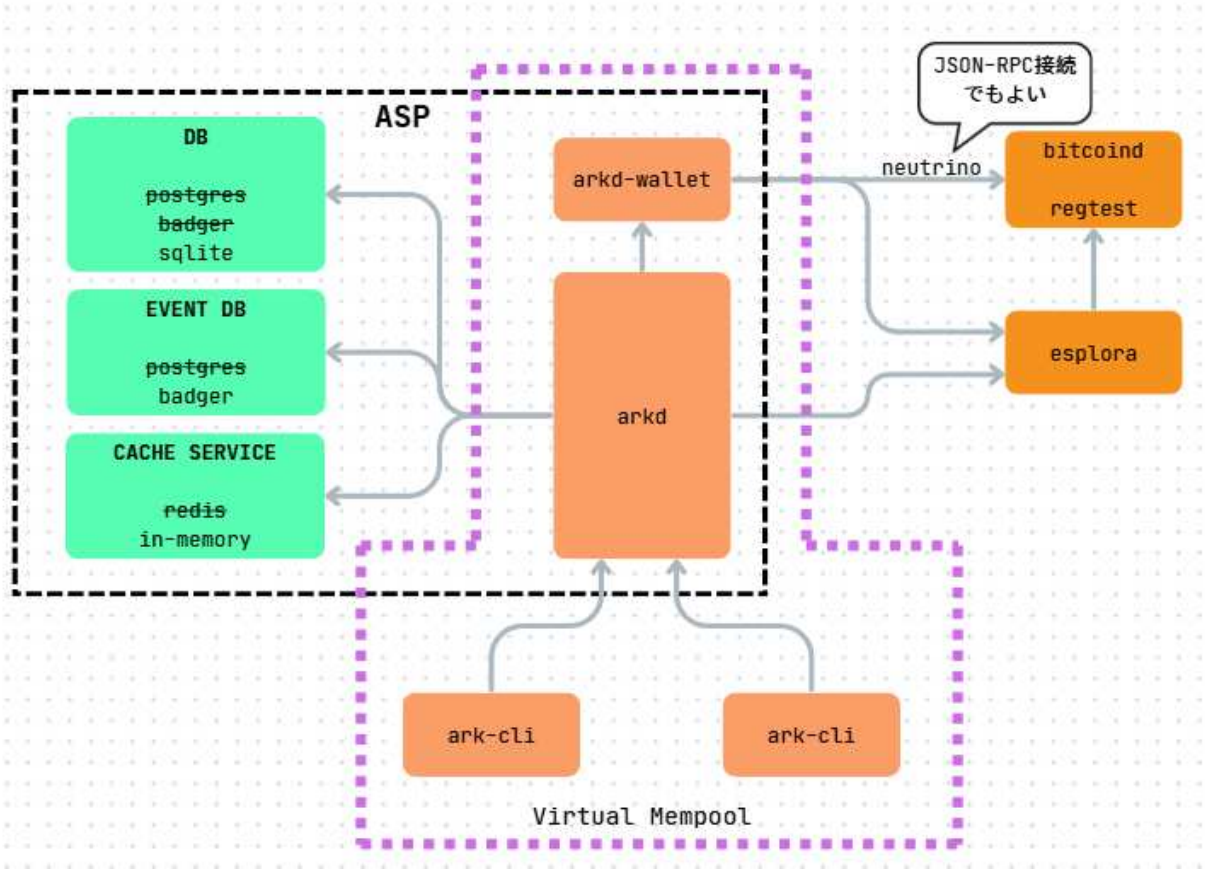
v0.7.0 でうまくいっていないのだが、こういう接続ができる。

`ark-cli` は各ユーザのクライアント相当の機能を持つ。ウォレットといえばよいか。

ArkのLayer2は Virtual Mempool という名前だと思えば良い。仮想的なmempoolの中で扱うのは仮想的なトランザクション VTXOである。つまり話の中心は**VTXO**である。

図では esplora と書いているが、正確には Esplora API を提供するサーバであればよい。

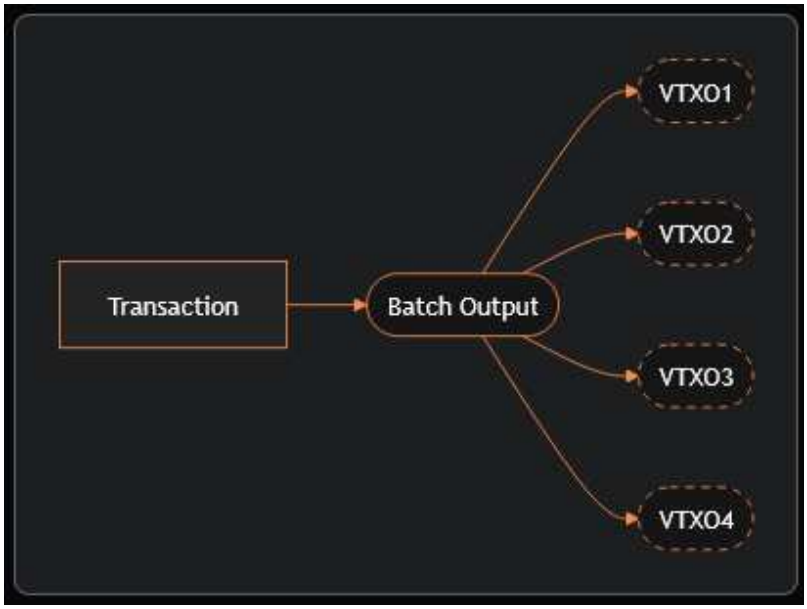
ただ、TypeScript のライブラリが mempool/electrs の API を使っていた。arkd 側がどうなのかわからないが mempool.space 版を使うのが無難だろう。



VTXOのしくみは Layer1 に類似しているが、所有者とオペレータのマルチシグになっている。

送信には仮想的なトランザクション Ark Transaction が作られる。INPUT は未使用の VTXO で OUTPUT は送金先(とお釣り)という構造はほぼ同じである。INPUT はマルチシグなので両者の署名がいる。

VTXOはツリー構造になって管理され、定期的に Commitment Transaction という形でオンチェーンに展開される。Commitment Transaction の OUTPUT で VTXO のデータと関連する方を Batch Output と呼ぶ。VTXOをまとめているからだろう。

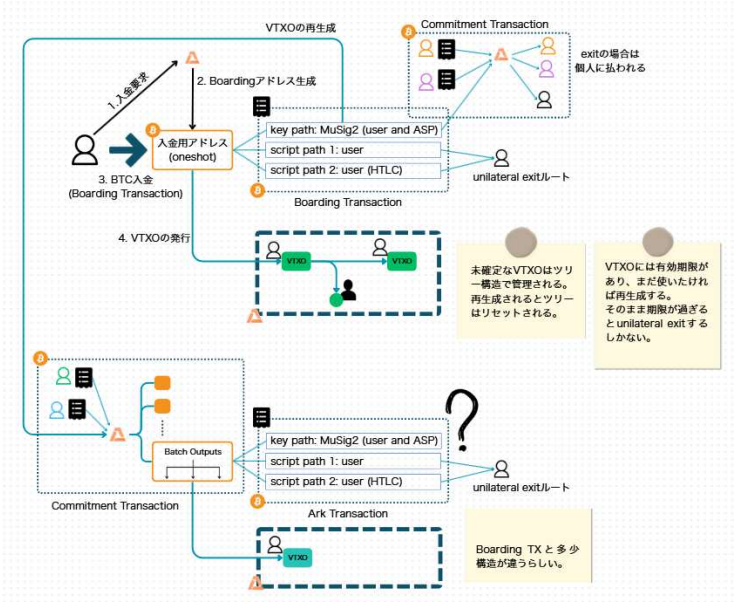


何をどうやってる？

ASP(Ark Service Provider)にユーザがぶら下がっている。
ユーザがASPにBitcoinを払うと、Ark の L2 上に同額の VTXO という仮想的なUTXOを持つことができる。

L2上では VTXO を使ってユーザ間で送受金できる。
LNのようにチャネルが無いのでユーザが流動性を気にする必要はない。
ASPはユーザのBitcoinを預かるわけではなく、Taprootのスクリプトで共同管理している形になる。
スクリプトの1つはユーザの鍵だけで支払えるようになっているので、ASPを信頼せずにオンチェーンに戻すことができる。

ユーザとASPが同意してL2を抜けたり、VTXOの期限が切れるので再生成するとオンチェーンに一度展開する仕様になっている。ASPはL2のVTXOを1日1回程度の頻度でまとめてオンチェーンに展開することで手数料を削減している。
ASPがVTXOの状態や期限、依存関係などの管理を "Virtual Mempool"と呼んでいる。オンチェーンに展開するまで貯めている様子をmempoolに見立てたのであろう。



Boarding https://docs.arkadeos.com/wallets/ramps#boarding
オンチェーンのUTXOをオフチェーンのVTXOと交換するしくみ。
Boarding Addressに送金するなど。

VTXOの期限が切れた場合、ASPが回収している。この動作を「sweep」と呼ぶ。
Batch output の tapscript の中に sweep path を持っているそうだ。

- internal key: n-of-n の MuSig2
  - VTXO treeの構造はオフチェーンで管理されてスクリプトには出てこない
- sweep path

VTXOの延命は、

- batch settlement(バッチ決済) によって新しい batch output に含めてもらう
- 送金すると自動的に新しいVTXOが作られるので延命になる

settle は pre-confirmed な VTXO を Bitcoinブロックチェーン上で confirmed にするプロセス(Settlement Process)。
現在はサーバーの `RegisterIntent` APIが呼ばれたりしているようだ。

Unilateral Exit は unroll に失敗したままで終わる(2025/08/07)。
Introduction - Build on Arkade で作ったサンプル

Arkade Contracts

https://docs.arkadeos.com/contracts/overview

Arkade言語という高級言語があり、Bitcoinスクリプトを拡張した Arkadeスクリプトに変換される

Arkade Script

Arkade Compiler

GitHubの方を見ると、 `*.ark` はArkade Scriptコントラクト？

`OP_xxx` ではなく Solidity みたいな感じか？

それをコンパイルするといろいろなデータを持つ JSONになるようだ

こんなの？ "asm" にオペコードが入っている。

`sample.apk`

```
// https://github.com/arkade-os/compiler/blob/master/examples/bare.ark
// Optional configuration options
options {
  server = serverPubkey;
  exit = 144; // Exit timelock in blocks
}

// Contract declaration with parameters
contract MyContract(
  pubkey user,
  pubkey server
) {
  // Function declarations (spending paths)
  function spend(signature userSig) {
    require(checkSig(userSig, user));
  }
}
```

UTXOモデルなのはそのままっぽい

Commit-Reveal方式に追加している

Commit-Reveal方式とは - ChatGPT: lockingスクリプトを unlockingするときに公開するようなやり方

その1: Transaction Introspection: トランザクションの情報を読むことができる

その2: Taproot and Key Tweaking: こちらは普通の P2TR と同じ？

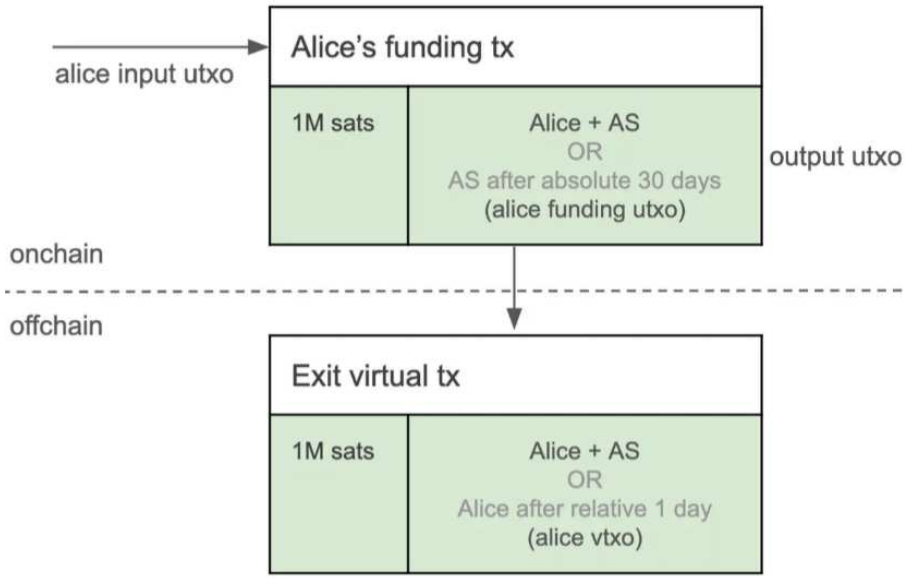
トークンサポート

再帰的コベナンツ

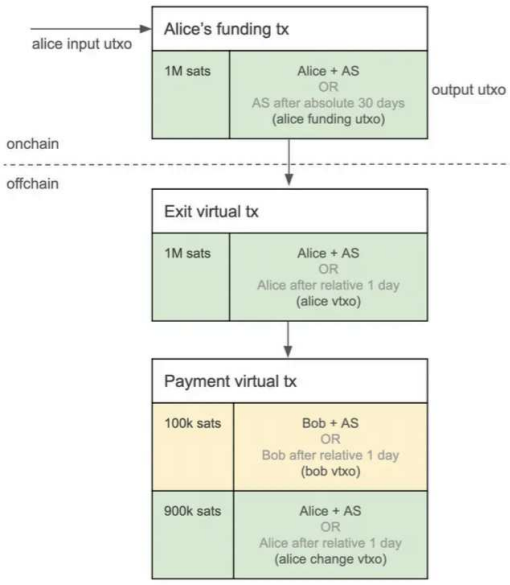
Ark 101. A Bitcoin Layer-2 Protocol - by Saravanan Mani - Jul, 2025 - Medium

これは Second社の用語が使われているので Arkadeとは挙動が違うかもしれない。

- boarding時
  - Alice が funding transactionを展開する
    - 「Alice と Ark Server(AS) のマルチシグ」or「30日後にASのみが使用できる」
      - funding transactionはAliceが支払うのだから、ASが30日間なにもしないともらえるというのは変では？
  - 展開する前にオフチェーンの終了VTXOトランザクション(VTX)にも共同で署名する
    - オンチェーンでも使用できるっぽい
    - このトランザクションを展開して、1日経てばAliceだけで使用することができるようになる
    - CPFP用の anchor output があるが図では省略
    - INPUTはfunding transactionのOUTPUT？ だとしたら「Alice + AS」「AS + 30日」の他に「Alice + 1日」という tapscript になっている？



  - ASが30日近く funding transaction を処理しないようだったら Alice が取り戻せるということか
- AliceからBobへの支払い
  - Alice の VTXO(ここではExit VTXからのoutput)からBob に 100k sats、お釣りの 900k satsを Aliceに割り振る。

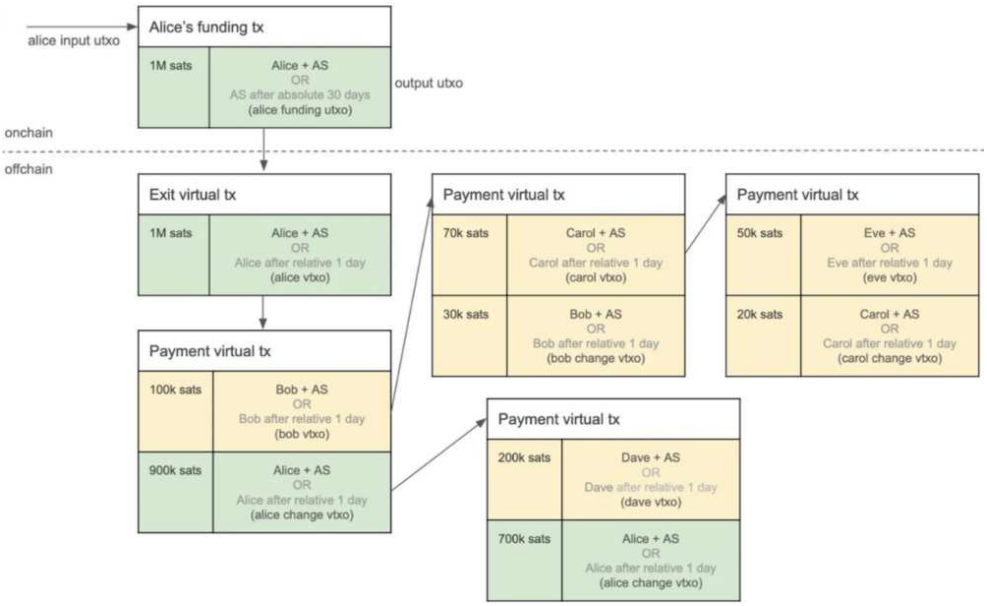


Bob の VTXOは Alice が AS と共謀して二重支払いになって無効になる可能性がある(図の黄色っぽいのはそういう意味)

Alice は自分をだます必要がないのでお釣りについては安全(図の緑色っぽい塗りつぶし)

以降の支払い

他のユーザーにも支払っていくと黄色のVTXが増えていく



VTX の連結が長くなると、一方的にexitするときにユーザーがすべてのトランザクションをオンチェーンに展開する手数料を払わないといけないので負担が大きい。

自分に関係しないトランザクションも支払わないといけないの？？

Cooperative Exit

ユーザとASが協調してexitする。

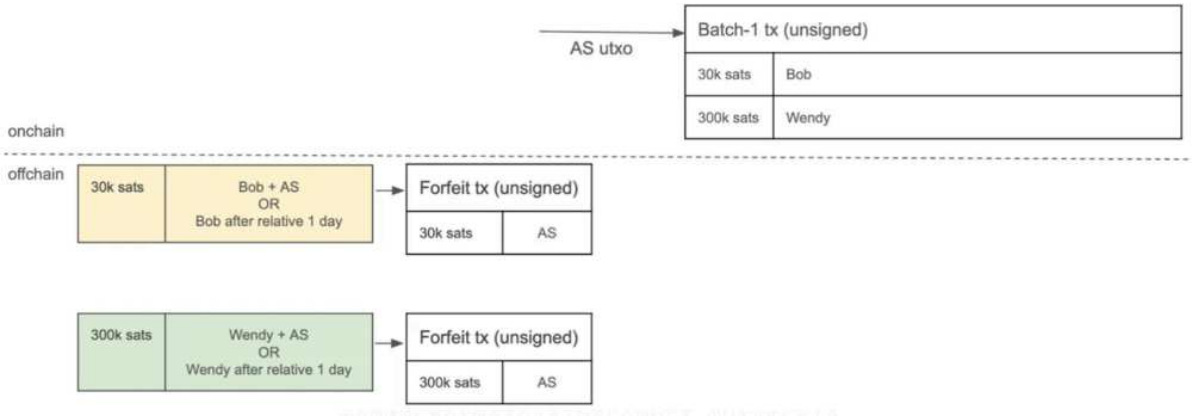
Alice は ASに exit したい(VTXOをUTXOと交換する)ことを伝える

アトミックに以下の交換を行う

Alice: ASが管理するアドレスに送金

AS: ASのオンチェーン資金から同額をユーザーが管理するアドレスに送金

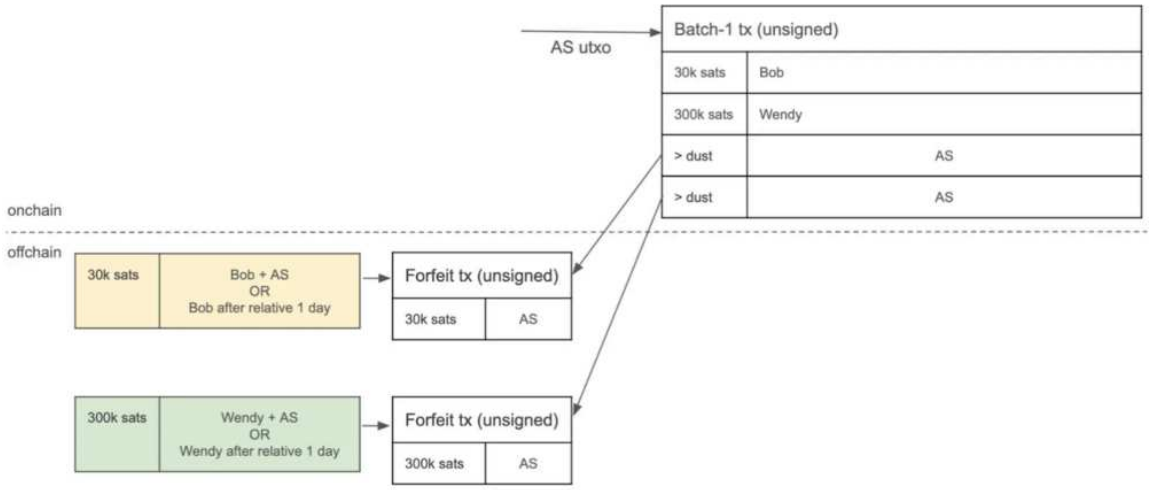
Bob と Wendy が協調するってなに？ ASとじゃないの？ 偶然同じタイミングで Bob と Wendy が協調exit しようとしただけってこと？



ASから Bob と Wendy に送金する batch transaction を展開してそれぞれに戻す準備をする

forfeit transaction はオフチェーンのVTXでそれぞれから AS に送金する準備をする

これらがアトミックスワップするようにしたいので、batch transaction に dust くらいの OUTPUT を追加し、forfeit transaction の INPUT に追加すると、オンチェーンで batch tx が confirm されないと AS が VTXO を受け取ることができない。



残念なこと(2025/07/07) の感想です。まだ理解していないことが多いので間違っているかも。

ASPをまたいだ送金はできない

構想はあるらしいが、未対応＆非推奨

VTXOの有効性はASPで閉じているので、他ASPから検証できない

ASPは独立運用する前提の設計で、他ASPを検証できない

Lightning Networkとやりとりできるするようになっていない(2025/07/07)

構想や提案はあるものの仕様や実装はない

Boltzでの Submarine Swap はあるようだ(Lightning Swaps - Build on Arkade)が、ライブラリを使うなどあるのでアプリレベルの話か？

Commitment Transactionを定期的にオンチェーンに展開する必要があるので、そのコストがかかる

ASPと送金ユーザが結託すると、VTXO に MultiSig 署名して送金したとしても、同じ VTXO に署名して別の人に送金できてしまう。次の Commitment Transaction で確認してようやく受金できていないことが分かる。

対策として、受金した VTXO を次 Commitment Transaction で確定させる Batch Swap という処理がある。その代わり次の Commitment Transactino まで待つことになる。

採用企業

2025年5月の記事で、Tether, Boltz, Breez, mempool.space などがArkadeをテスト中というのがある。

今はPoCの段階で、商業的には2025年Q3から

Ark用語

Glossary - Build on Arkade

Ark Terminology - Docs - Ark Labs

気になった用語をいくつか取り上げる。

preconfirmation, preconfirmed, Ark's Preconfirmation State

送受金などによる Ark Transaction でVTXOは作られたが L1 で Commitment Transaction として confirm していない状態。

VTXOが作られる＝Arkade Signerが署名した(cosign、共同署名？)、ということらしい。そうするとL2上で使えるようになる。

preconfirmed VTXOは脆弱で、オペレーターはトランザクションを二重支払いさせたり競合させたりできてしまう。ユーザだけでは有効な方のトランザクションを証明できない。

要素

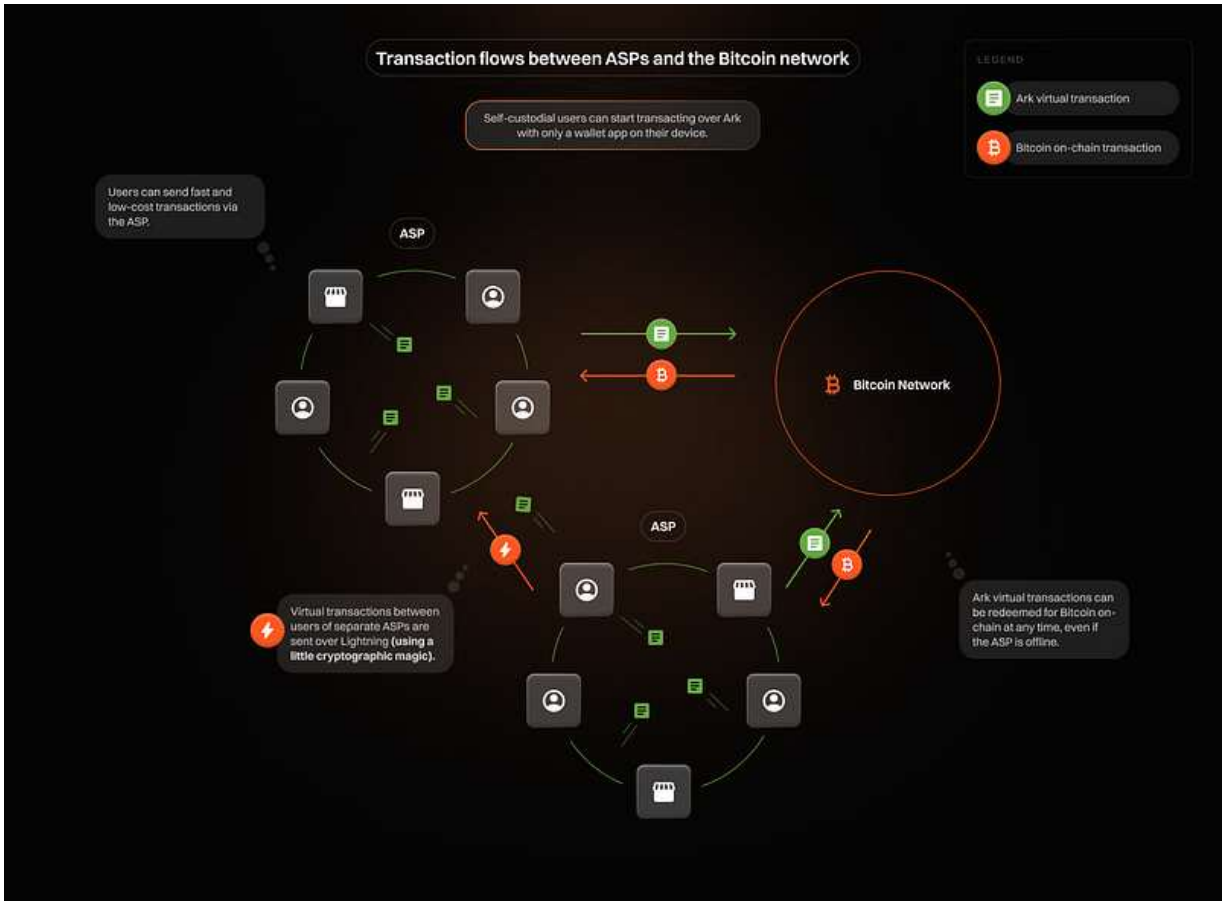
VTXO

ASPがユーザに対して発行する

有効期限付きで定期的な再発行が必要

- 単位は Bitcoin と同じ。
    - VTXOの最小値はASPで設定できるようでデフォルトは1だった。
- ASP(Ark Service Provider)
    - 機能
        - VTXOの発行
        - 送金処理の仲介
        - オンチェーンのArkトランザクションの作成・展開
- Commitment Transaction
    - Tapscript で複数の送金を1つのトランザクションにまとめる
    - `SIGHASH_ANYPREVOUT` や `SIGHASH_ANYONECANPAY` を使って多人数の署名をうまくまとめる
    - UTXOの再構成やVTXOのロールオーバー(再発行)を行う
        - ロールオーバーはオフチェーンで使われるArk Transactionの組み替えだけでオンチェーンとは関係が無い
        - ASPごとに決められる。ブロック時間で1日程度が目安だとか。
        - ユーザがexitを要求すると次の commitment transaction にそのユーザが含むことになる
    - `OP_CAT` や `OP_CTV` などのsoft forkは不要
        - `SIGHASH_ANYPREVOUT` が使えるようになったら使いたいようだが、今はそう言うの無しで良い
        - その代わり、信頼できる人が一人でもいるようにしてマルチシグにして送金先をチェックできるので、それが実質的なコベナンツだ、という
- Exit機構
    - ASPが不正をしてもユーザは L1で署名して単独で資産をExitできる(unilateral exit)
    - ただ、サーバ・クライアント

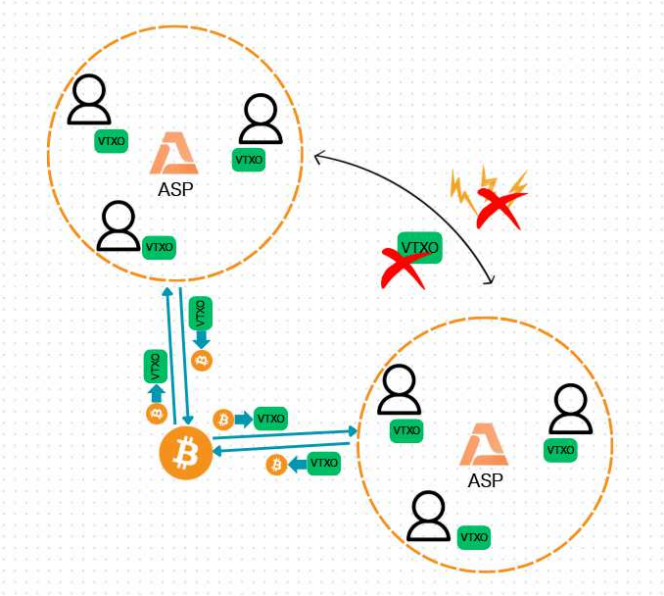
こういう図があったので小さなフェデレーションを作るのかと思ったが、1つのまとまりは1つの ASP が管理する。
しかし ASP を信用する必要はなく、Exit Path といってユーザはオンチェーンで資金を取り戻せるようになっている。
[Introducing Ark Labs: A new venture to bring seamless and scalable payments to bitcoin]



ブログではこういう図だったが、2025年7月ではまだこういう状況である。

ASPをまたいだ送受金はない

Lightning Networkもない(SWAPはできるみたい)



入金をASPに要求すると入金用のアドレスがもらえる。再利用はしないアドレスである。

ユーザが入金した BTC は Key path + Script path になっている。ASP が回収できるのは Key path だけ。しかもユーザとの MuSig2 になっているため、ユーザが動かないことには ASP は資金が回収できない。

ユーザは同意してexitするか、強制的にexitするかのどちらかである。

LSP のように資金をロックされたまま相手がいなくなると運用資金がつらくなる。
LN の unilateral close のようなものがないプロトコルである以上どうしようもないので、運用でがんばる。
boardingのトランザクションについては、そこに送金して30日(`ARKD_BOARDING_EXIT_DELAY` ?)経つとASPが使えるようになるそうだ。

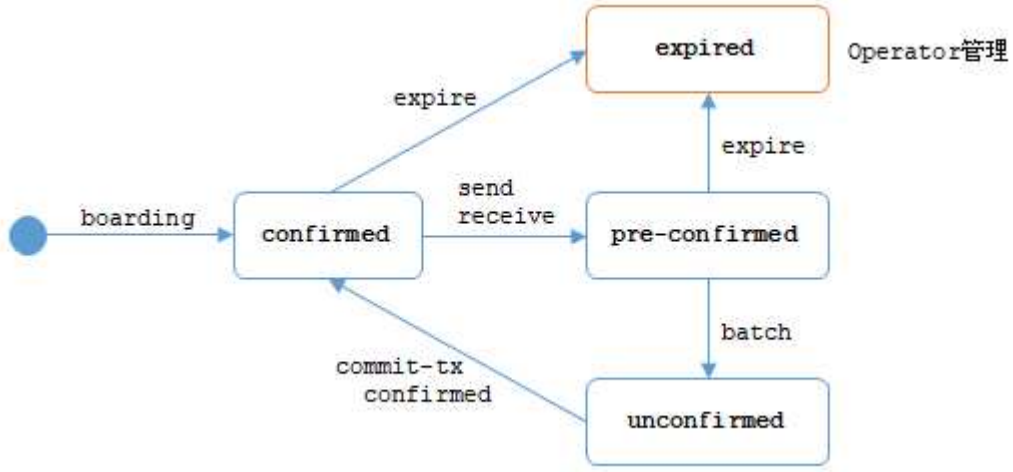
トランザクションの種類

- Commitment Transaction
  - 送金先
    - batch outputs
      - collaborative exit した VTXO(出金)
      - ロールオーバーする VTXO(VTXOの再生成)
    - connector output
      - anchor output とも呼ばれる
      - dust limit 付近の output で、forfeit transaction の input になる(後述)
- Virtual Transaction：基本的にオフチェーンで使用するトランザクション。オンチェーンにも展開はできるが普通はしない。
  - Ark Transaction
    - VTXOでの送受金に使う
  - Forfait Transaction
    - VTXOを次のbatch でVTXOとして使えるようにしたり、協調的exit してユーザにUTXOとして戻したときに同量の VTXOを ASPが取るのに使う。
    - batch output と連動するので、Commitment Transaction の anchor output を INPUT にすることで「forfeit transaction を有効にしたいなら commitment transaction を展開する必要がある」というインセンティブを生む

VTXO

Virtual UTXO のことで、Ark の Layer2 上で送受金する単位になる。
金額は Bitcoin と同じ。

Lifecycle Statesを見ながら図にしたが、どうもすっきりしない。



種類？

Arkのドキュメントにはないのだがこちらの資料にあった。実際のところ、最初に入金した Boarding Transaction ではタイムロックされていたりするので、そういった違いを種類としているのかもしれない。

有効期限がある理由

1. 不在ユーザーによる資金ロックの防止
2. ロールオーバーにユーザー署名が必要なため
3. exitの保証と無限保守義務の切り離し
4. CoinJoin的構造の維持

有効期限を過ぎたユーザのVTXOはどうなる？

そのVTXOはASPやArkのものになるわけではない。
ASPはそのVTXOに対するサービス義務から解放されることになる。
ユーザだけの鍵でオンチェーンに戻す

これに関してはnotesというしくみがあるそうだ(telegram)。

> Working on adding to docs as well :) but TL;DR is a mechanism where in the remote edge case the client software fails repeatedly to “renew” your balance before expiration (every VTXO has an expiry date attached to it), the operator basically credits you those balances, as soon the user is back online; these don’t have an unilateral claim path, but the goal is to redeem those notes for VTXO at the first opportunity (automated by client SDKs)
> ドキュメントへの追加作業も進めています :) TL;DR は、リモートエッジケースでクライアン

> トソフトウェアが有効期限前に残高を「更新」するのに繰り返し失敗するメカニズムです (すべての VTXO には有効期限が付いています)。オペレーターは基本的に、ユーザーがオンラインに戻るとすぐにそれらの残高をクレジットします。これらには一方的な請求パスはありませんが、最初の機会にそれらのノートを VTXO に引き換えることが目標です (クライアント SDK によって自動化されます)。

ASPの不正

Commitment Transaction は ASP が作る。batch outputs の選択も ASPが行うため、一部のL2支払いを含まない Commitment Transaction の作成は可能である。
ユーザは Commitment Transaction とオフチェーンの Ark Transaction を見比べることでオンチェーンに展開されたかどうかの検出はできる(タイミングが悪くて次回の Commitment Transaction に回されただけかもしれないが、そういう違いはわかるのだろうか？)。
Transaction Finality - Build on Arkade
ここにはそういうのは出てこなかった。

ASPの不正にせよ、DBが吹っ飛んでデータが消えたにせよ、それによって Commitment Transaction を取り消したりすることはできない。できるのは exit することだけである。

DEX

ArkはDEX AMM型のアプリを作ることができる。といっている。

Ark AIの主張

1. AMMの実装
   Liquidity Poolの自動管理
   数学的な価格決定メカニズム
   信頼する仲介者無しでの継続的なマーケット作り
   効率的な資本利用
2. 非対話型swap
   Arkadeでは両当事者が同時にオンラインである必要がない非対話型スワップを実装できる
   一方の当事者が swapオファーを作成
   相手方が都合の良いときに swapを完了
   Liquidity Pool もクライアントの1つだから常にオンラインとは限らないということか
   信頼できる第三者無しでの自動検証

3. 主要な技術的特徴
    Arkade Scriptの活用
        トランザクション内省機能 - スクリプトがトランザクションの詳細を検査可能
        高度な算術演算 - 金融ロジック用の包括的な数学演算
        複雑なロジック - 多経路実行と洗練された条件分岐
    並列実行
        独立したVTXO（仮想トランザクション出力）ベースのトランザクションが並列実行
            といってもASPがそれぞれさばく感じになるよな？
        グローバルな競合状態の排除
            VTXOをツリー状に管理しているので二重支払いのようなことは起きないと言いたい？
        水平スケーラビリティの実現
            ？なんだろう？
        セルフカストディアルセキュリティ
            ユーザーは一方的な退出権を保持
                これは VTXO の機能か
            事前署名されたBitcoinトランザクションによる強制力
            完全な資産コントロールの維持
                ASP や Liquidity Poolに 預ける形にならないということ？
4. 実装可能なDEX機能
    スポット取引 - 即座の資産交換
    流動性プロビジョニング - 流動性提供者への報酬メカニズム
    マルチアセット対応 - Bitcoin以外の資産もサポート
    高頻度取引 - 低レイテンシーでの取引実行
    クロスチェーンスワップ - 異なるブロックチェーン間での資産交換
5. 開発状況(2025/07/16)
    クローズドベータ - Bitcoinメインネットで選定パートナーと実施中
    パブリックローンチ - 2025年Q3予定
    開発者SDK - GitHubで早期ビルダー向けに利用可能

Automated Market Makers - Build on Arkade

AMMの基本実装コードがある。

contract ConstantProductAMM
    function verifyConstantProduct
    function addLiquidity
    function removeLiquidity

function swapXforY

function swapYforX

`X` や `Y` がトークンというかアセットだと思うが、 `bytes32 assetX` で `tx.inputs[1].asset` のような変数と比較するようになっている。ETHなどのトークンを扱う場合はどう使うのか？

Synthetic Assets

> Synthetic assets は、基礎となる資産の直接的な所有権を必要とせずに、他の資産の価値を模倣するトークン化されたデリバティブです。 Arkade Scriptを使用して、開発者はArkadeエコシステム内のビットコインまたは他の資産に裏付けられた洗練された Synthetic assets を作成できます。

Arkade上でできること

従来の金融資産（株式、商品、外国為替）への価格エクスポージャー

新しい金融商品の作成（オプション、未来、永久）

プログラム可能な清算(liquidation)メカニズムを備えた担保付き債務ポジション

価格供給のための信頼のないOracle統合

オリジナルのトークンを直接扱うわけではないが、それに連動した価値のトークンを用意して(今回は Arkade上)そちらを扱うということか。
Synthetic Assets で ETH価格に連動する sETH を作り、オラクルで価格を合わせ、最後はクロスチェーンブリッジ・・・ってどうするんじゃい。

---

ark.pdf

ark.pdf (日本語訳)(ChatGPT)

Virtual Transaction は基本的にオンチェーンに展開しない

送金に使うのが Ark Transaction

Forfeit Transaction は Commitment Transaction の anchor output(dust付近) を INPUT に持つことで、Commitment Transaction が展開されないと展開できないようにしている(Atomic Swap的な)

VTXO を Operator に送金する

それによって新しい VTXO にしたり、協力的に exit して UTXO にしたりする

Checkpoint Transaction

Commitment Transactionを展開することを「Run the Ark」などと呼んでいる。資金のほと

んどは Operator が出している。出力先は batch で VTXOを生み出したりする。

ユーザは batch の VTXO や Ark Transaction の VTXO から Commitment Transaction によって新しい VTXO にスワップできる。batch swap というらしいが文章がよくわからない❓

Figure 1のシナリオ

1. Alice が Arkに参入

   boarding transaction と書いてあるが正式名称かは不明
2. Bob に簡単な L2取引を行う
3. Alice は Operator と協力して離脱
4. Bob は離脱しようとしたが Operator が不在だったので一方的に離脱

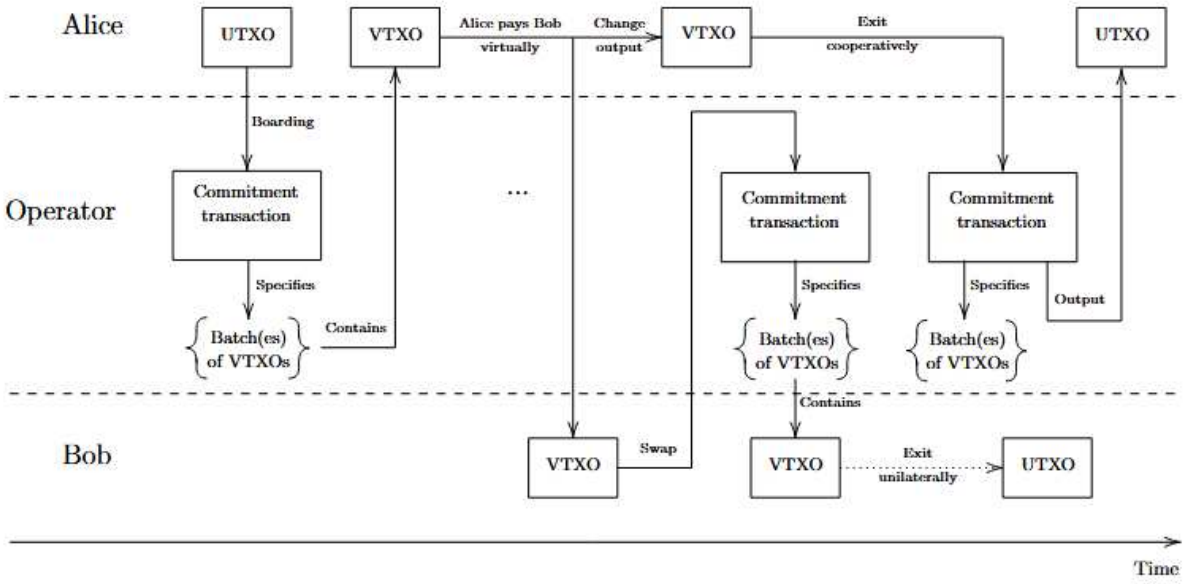


Figure 1: A general flow within the Ark protocol.

4.2章 Ark transactions

VTXOでの送受金直後、受金したVTXOがまだその人のものではないということについての説明。

例として、Alice から Bob への送金。Alice が持つ VTXO(`vtxoA`) 1つで十分なくらいの送金額 `p`

Ark transactionの構成

INPUT: Alice の VTXO(`vtxoA`)

OUTPUT

Bobへの送金(`vtxoB = p`)

Operatorへの手数料(`vtxoO = f`)

お釣り(`vtxoA' = vtxoA - vtxoB - f`)

Witness

$W_{collab}(O \oplus A)$

つまり持ち主単独では署名できないということだ。

MuSig2 ではなくスクリプトでのマルチシグかもしれない

Bob は Alice と Operator によって署名された `vtxoB` を受け取った。しかし、それがBobのものになったとするには弱い点が2つある

- Bob は `vtxoB` を使って unilateral exit することができない。

  `vtxoA` がオンチェーンに載らないと unilateral exitするトランザクションの INPUT がないからである。

  `vtxoA` がオンチェーン展開されるのを監視するしかない。

  それか普通に cooperative exit するか。

  `vtxoA` が unilateral exit でオンチェーン展開できるということは、 `vtxoA` を含むバッチのVTXTに複数のVTXOを含んでいることになる(お釣りとか)。

  - ? アリスがこれらのトランザクションをすべてボブに送信した場合、ボブもこれらのトランザクションをすべてvtxoAをオンチェーンにポストし、vtxoBを取得するための後続のArkトランザクションもポストすることで、一方的にExitできるようになります。

    関連するVTXOが全部Bob 宛になるので展開できるという意味？

- Alice と Operator が共謀して、Bob への `vtxoB` とは別に `vtxoA` を Inputにした VTXO を作ることができてしまう。つまり二重支払いである。

  その対策は4.3章で説明。

4.3章 Batch Swaps

4.2章で述べたように、preconfirm な VTXO を交換している間は ASP が送金者と共謀していないことを信頼する必要がある。受け取ったVTXO が次の commitment transaction 以降でもちゃんと使えることがわかれば心配は無くなる。なぜなら commitment transaction の OUTPUT から直接生えている VTXO は unilateral exit してオンチェーンで UTXO に変換できるからだ。

Batch Swaps(batch swapping) は受信者が受け取った preconfirmed VTXO を、次の commitment transaction に含まれる confirmed VTXOと交換するしくみである。

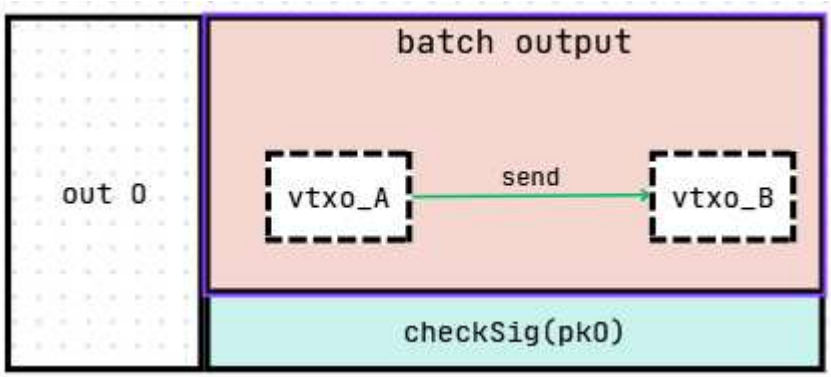
英文では "new VTXO" と交換すると書いてある。

英文も、英文をツールで直訳しても、何を言いたいかわからない。。。

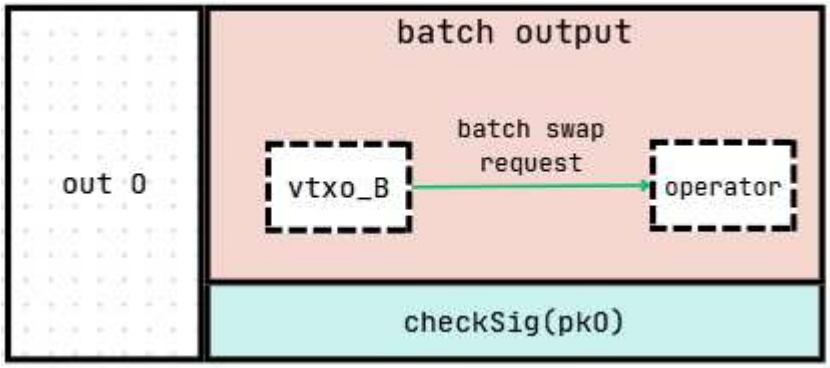
プロトコルの流れ

1. Bob が Alice から受金して `vtxoB` をもらう

   この時点で `vtxoB` は preconfirmed



2. Bob は Operator に対して Ark Transaction の swap要求を送信する



3. Operator はBob のリクエストに含まれる Ark Transaction の整合性と正当性を検証する

入力のVTXO(ここでは `vtxoB`)が未使用である

Ark Transaction の署名が正しい

Transaction 構造に間違いが無い

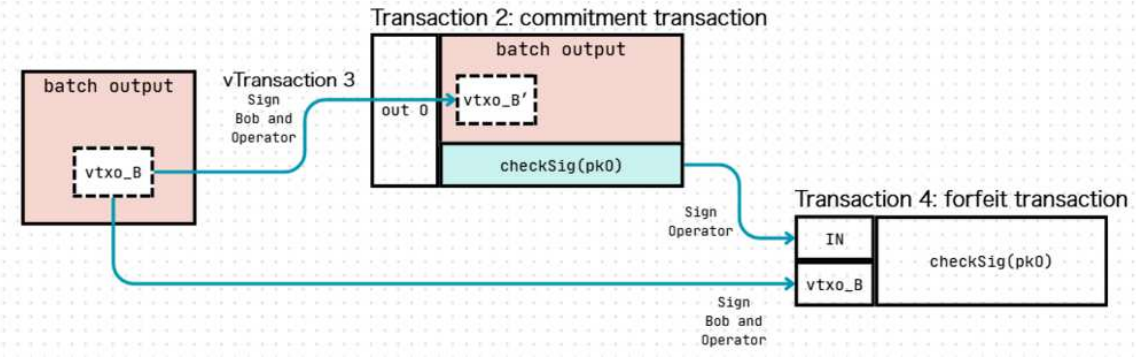
4. Operator は次の commitment transaction で `vtxoB` に相当する `vtxoB'` を生成して含める。

`vtxoB` から手数料分が引かれてamountが `b` から `b'` になっている(おそらく)

`vtxoB'` を output とする Virtual Transaction に Bob と Operator が署名する

Bob は Operator が `vtxoB'` に署名したデータを受け取って確認したら Forfeit Transaction を作成できる

と書いてあるが、connector output の署名は



`vtxoB` は unconfirmed になってそのラウンドでは使用できない

5. Bob は次の commitment transaction(Transaction 2) がオンチェーンで confirm され `vtxoB'` があることを確認する。

4.4章 Commitment Transaction

Commitment Transaction は最低でも 1つの batch output と 1つの connector output を持つ。

Batch output

Batch Output - Terminology

仮想トランザクションのツリーをカプセル化するコミットメントトランザクションのタプルート出力。

各葉はVTXOを表します。バッチ出力はMUSIG2キーによって保護されており、有効期限が切れた後にオペレーターによって制御されるフォールバックスクリプトパスが含まれています。

Batch Output - Glossary

複数のユーザーの所有権主張を、署名済みの仮想トランザクションツリーを通じて統合する単一のビットコイン出力。

コミットメントトランザクションを通じて作成され、特定の支出条件を持つTaprootアドレスにロックされます。

presigned vtx のツリーを通して複数ユーザの所有を統合した単独の Bitcoin output。

VTXT を merkle tree にしている？ それは Connector output？

ということは各 VTXO のルートは Batch output のルートから生えたリーフ？

複数ユーザの所有なので、マルチシグは関連する VTXO の所有者全員？

1つの batch output で現在のところ 128 VTXO まで扱えるようにしている(FAQ)

Connector output

Connector - Terminology

アトミックスワップに使用されるコミットメントトランザクションの特別な出力。

仮想コネクタを介して没収されたトランザクションを固定し、コミットメントトランザクションが確認された場合にのみスワップが成功するようにします。

Connector Bonds - Glossary

貸し手がバッチで使用済み VTXO にリンクされたコネクタ出力と BTC を交換することで、利子獲得の可能性のある出口パスを一時的に制御できる、外部流動性を調達するためのメカニズム。

コネクタとは、タップルートスクリプト（connectorScript）によってロックされるトランザクション出力です。このスクリプトは、使用不可のキーパスと、VTXTに従って支出を指定するスクリプトパスを持ちます。VTXTでは、ルートはコネクタ全体を支出し、VTXTの各リーフはアンカー出力を唯一の出力として持ちます。VTXTの残りのノードは、コネクタを唯一の出力とする仮想トランザクションです。

Arkプロトコルでは、コネクタは入力として使用されるすべてのアンカー出力をカプセル

化し、そのコネクタを含むコミットメントトランザクションがオンチェーンに取り込まれた場合にのみオンチェーンに取り込まれるトランザクションを放棄します。バッチとは異なり、コネクタ内の仮想トランザクションはオペレーターによってのみ署名されます。これで、コミットメントトランザクションを正式に導入できます。

❓前のCommitment Transaction の Connector output を新しい Commitment Transaction の INPUT にする？

❓forfeit transactionで使う？