Light Protocol v3

Private Solana Programs

Workshop Contents

We will:

- Learn about:
  - Shielded Transactions
  - Utxos
  - Encrypted State (Data)
- Implement a private OTC swap as Private Solana Program (PSP)
- Workshop Git Repo:

$ git clone https://github.com/Lightprotocol/breakpoint-workshop

Light Protocol: Shielded Balance

Solana

Solana Balance
- 21 Sol

- Transparent Accounts

Light Protocol

Shielded Balance
- 0 Sol

Light Protocol: Shielded Balance

Solana

Solana Balance
- 21 Sol

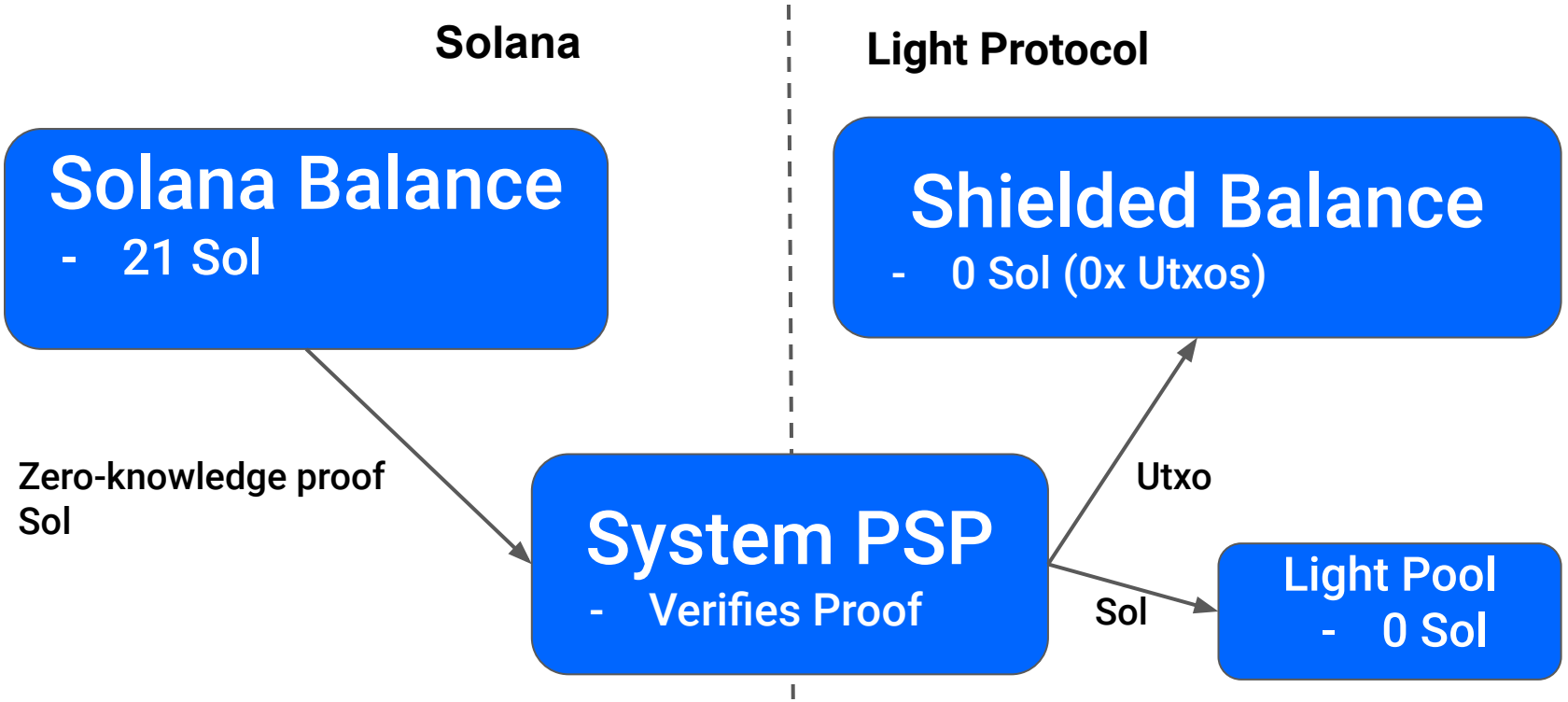
- Transparent Accounts

Light Protocol

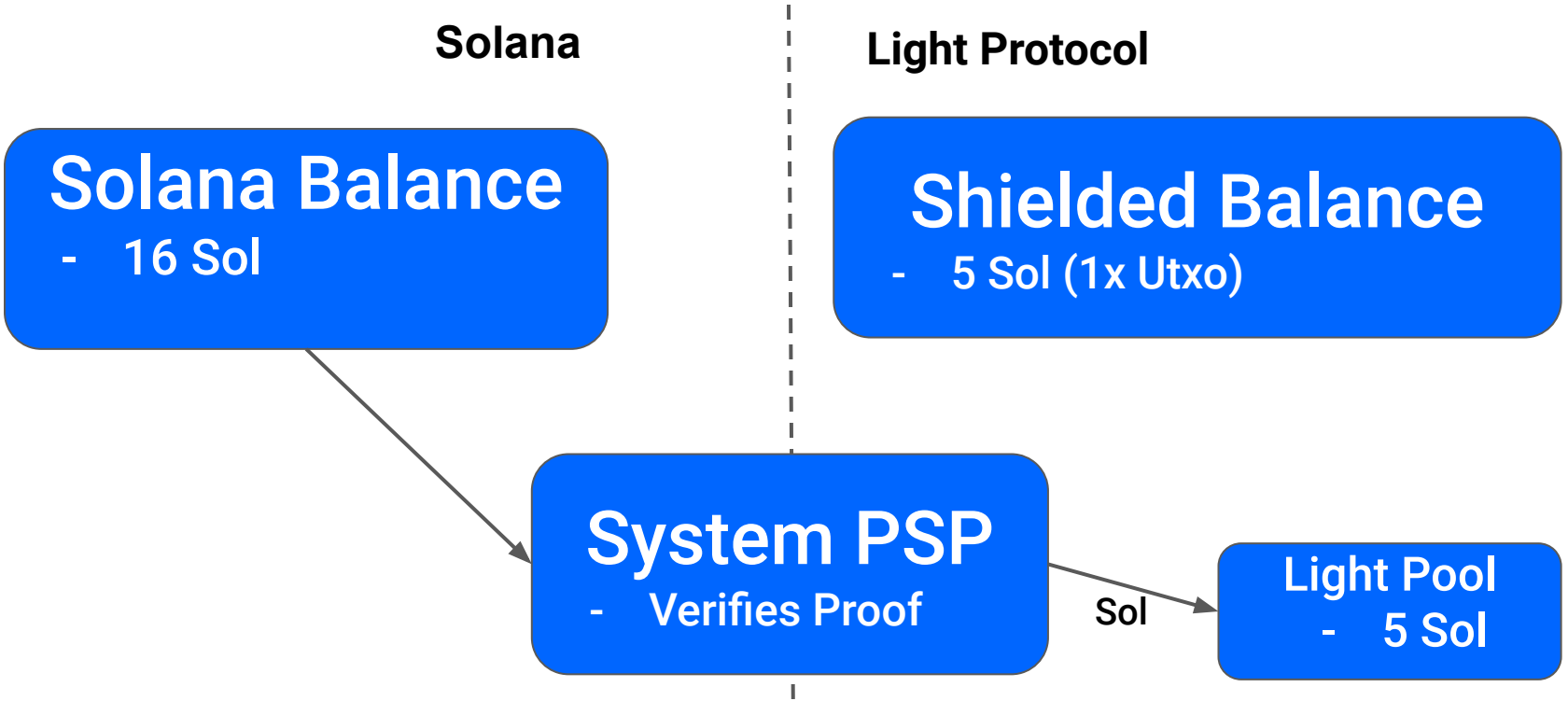
Shielded Balance
- 0 Sol

- Encrypted Balance/State (sender, recipient, amount, state transition)

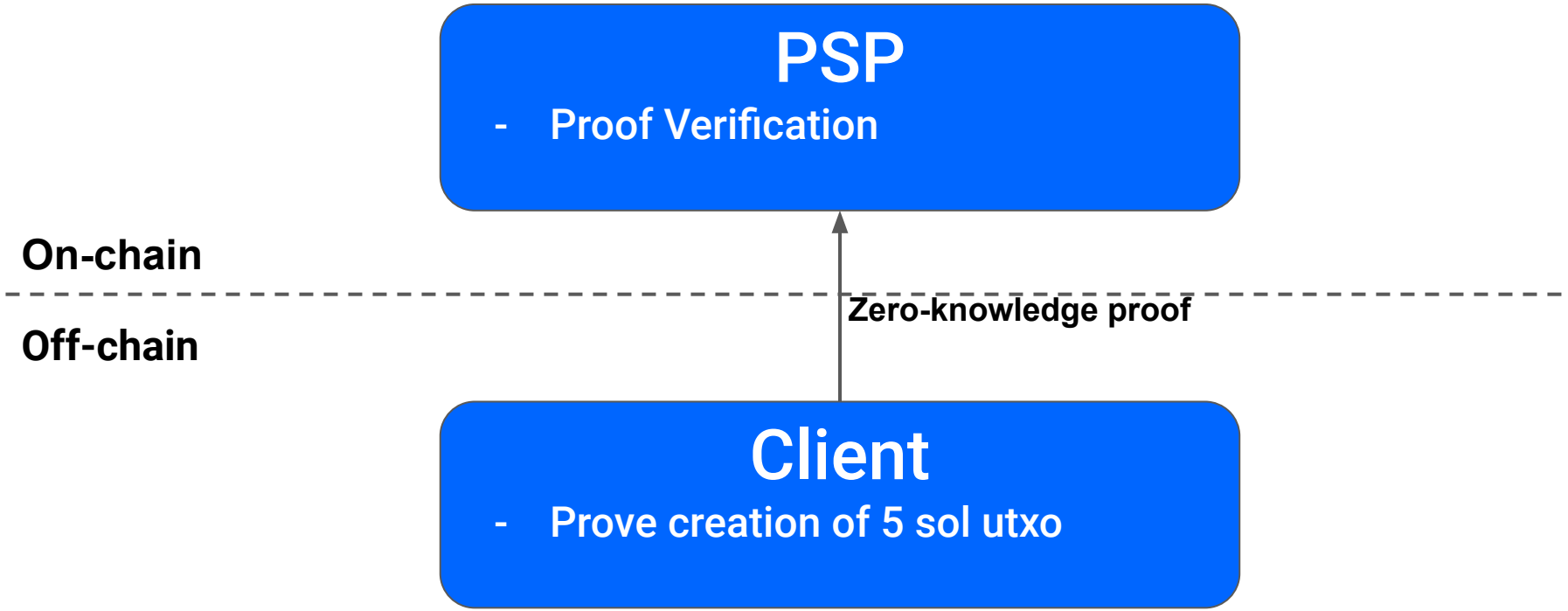
Light Protocol: Shield Sol



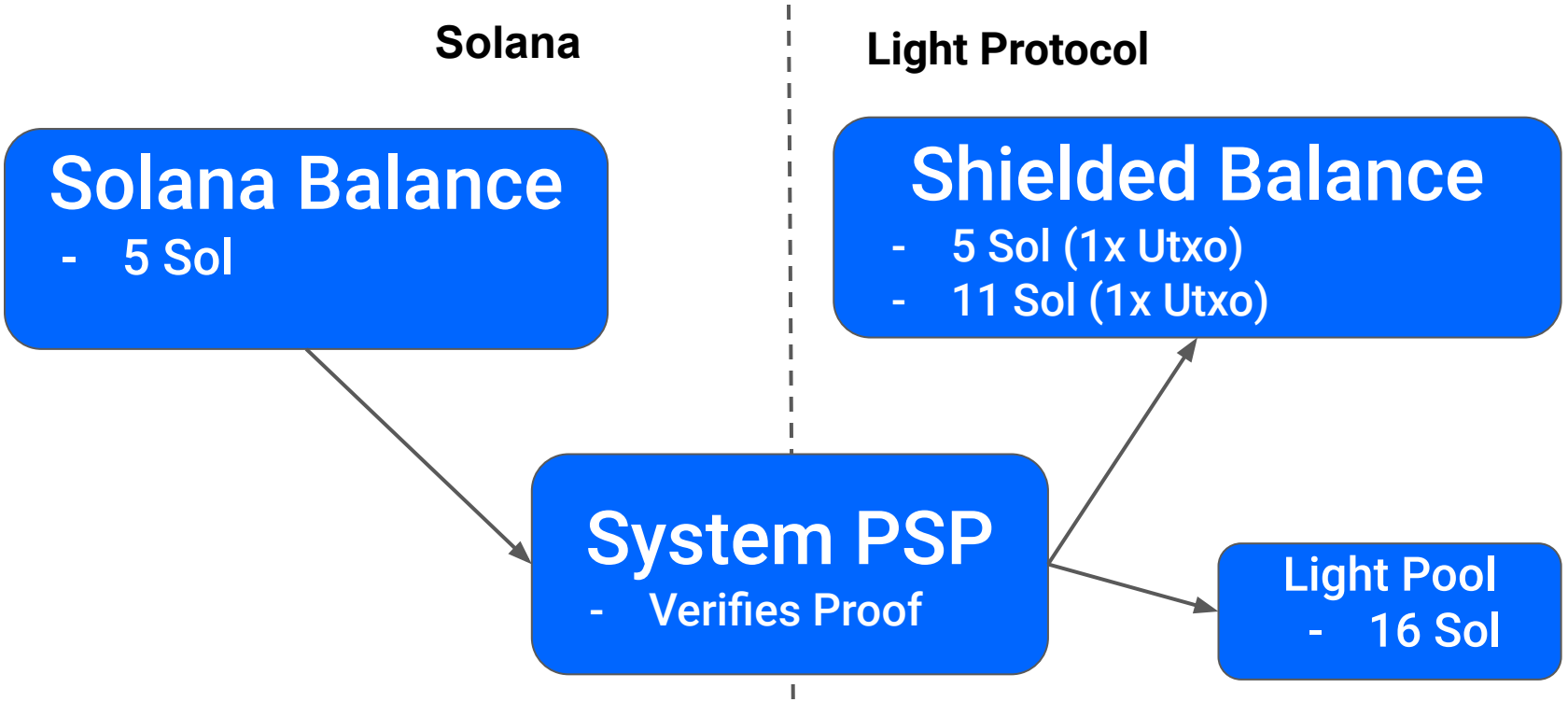
Light Protocol: Shield 5 Sol



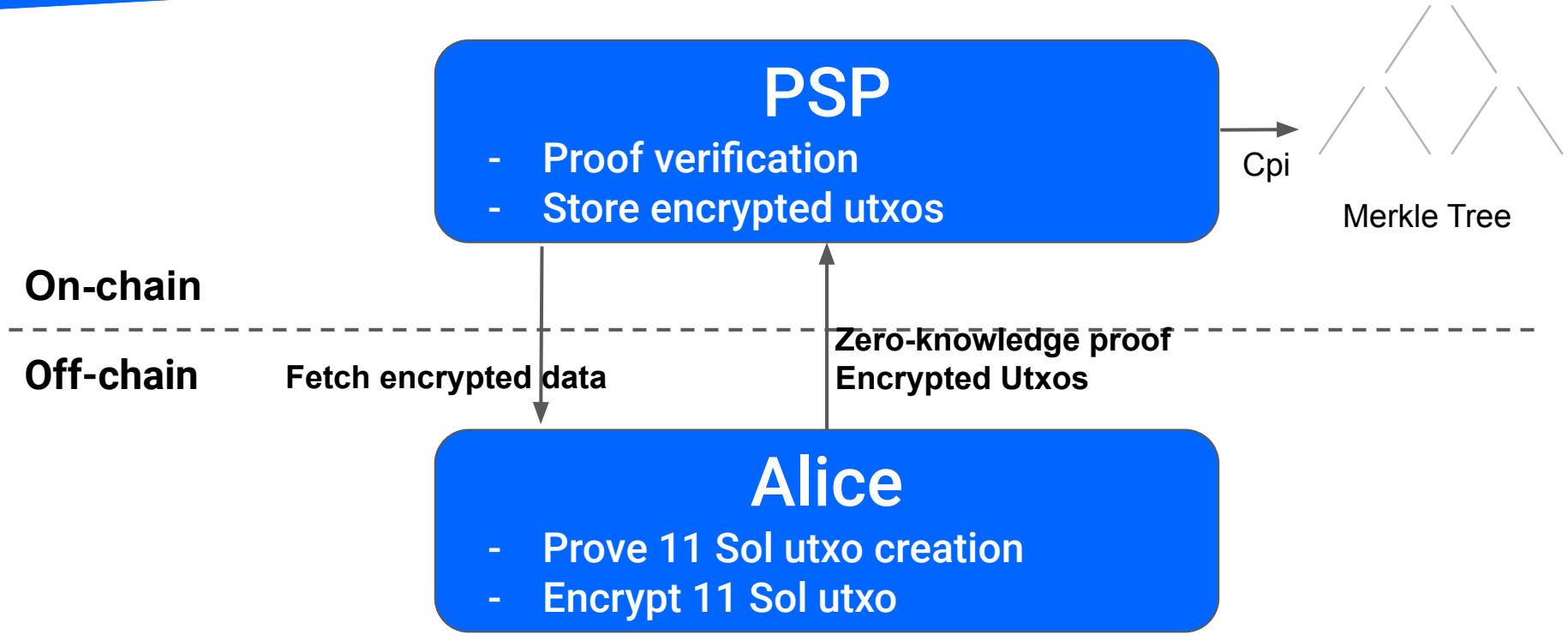
Light Protocol: Utxo Creation



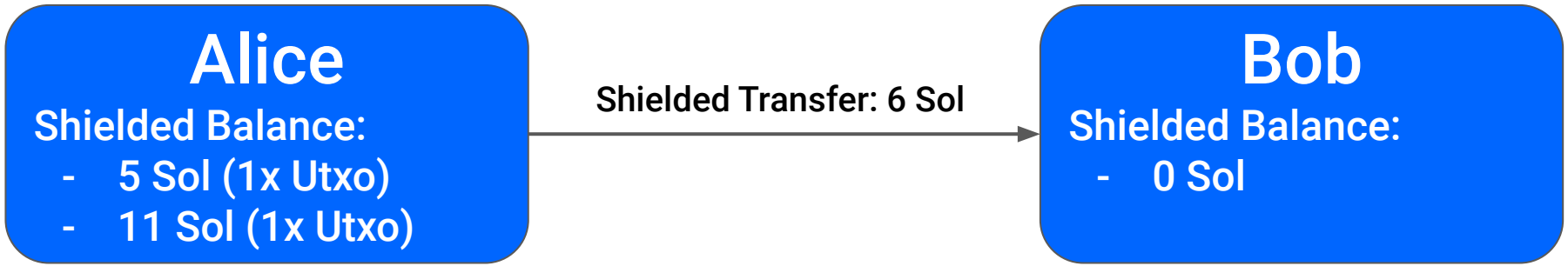
Light Protocol: Shield 11 Sol



Light Protocol: Data Availability



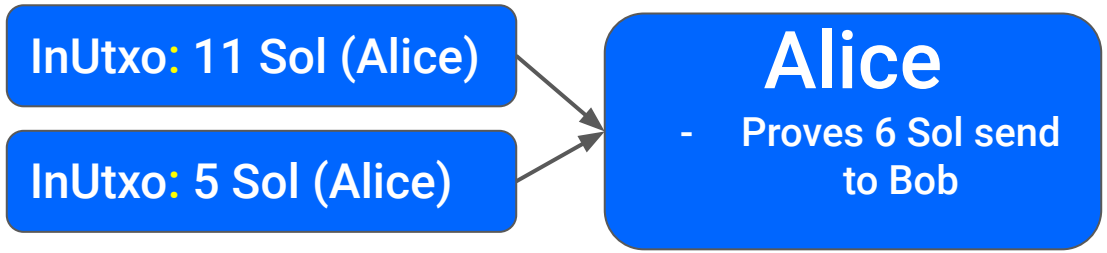
Shielded 6 Sol Transfer



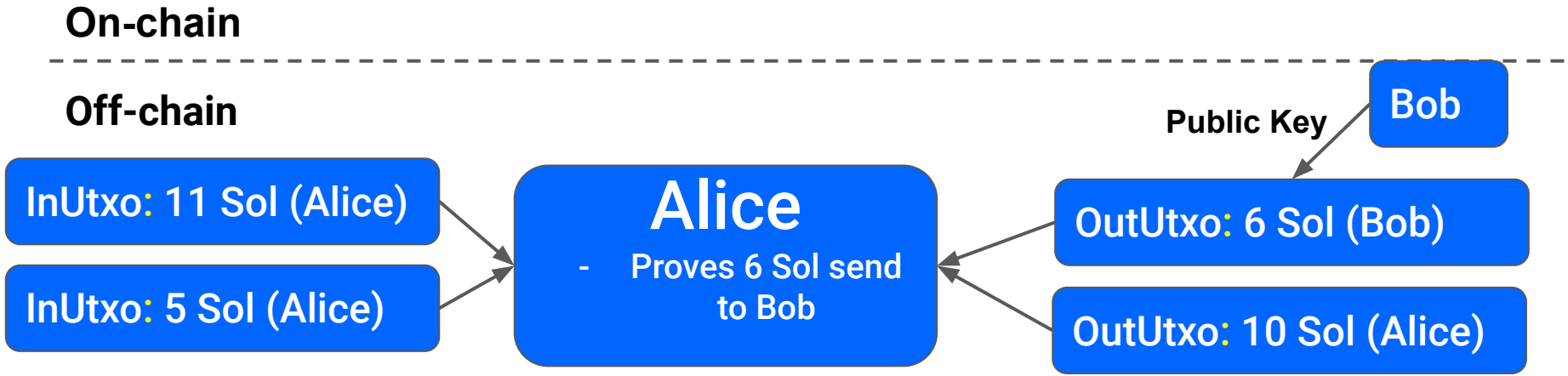
Shielded 6 Sol Transfer

On-chain

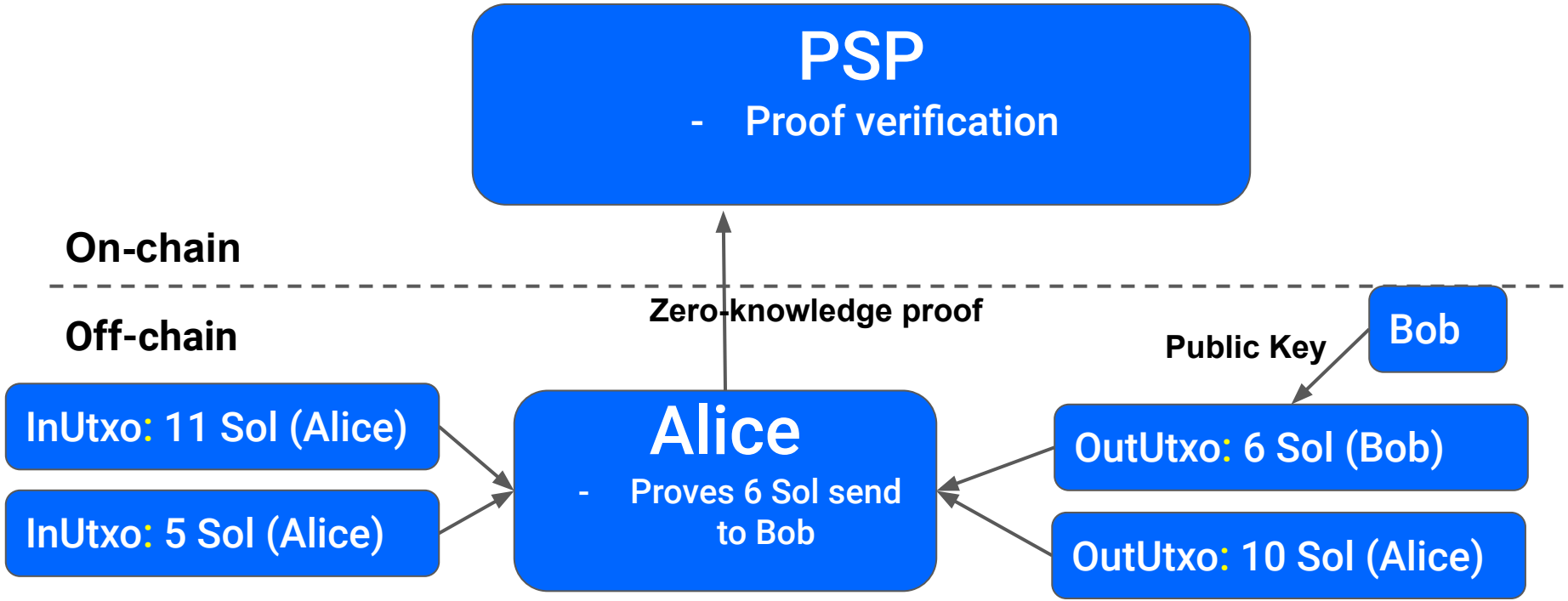
Off-chain



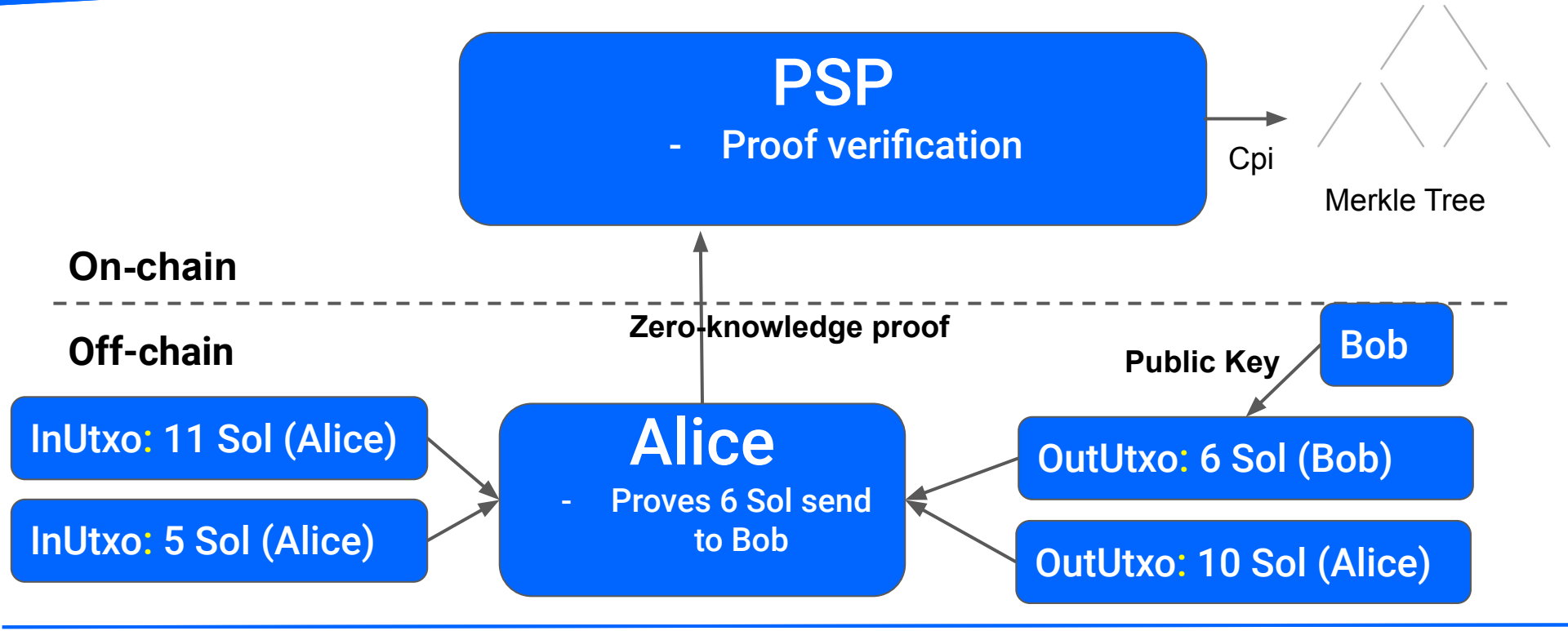
Shielded 6 Sol Transfer



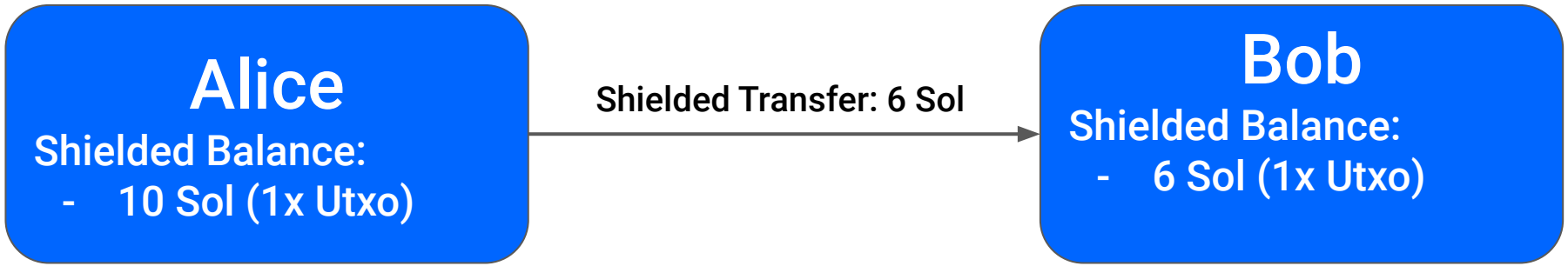
Shielded 6 Sol Transfer



Shielded 6 Sol Transfer



Shielded 6 Sol Transfer



Shielded OTC Swap

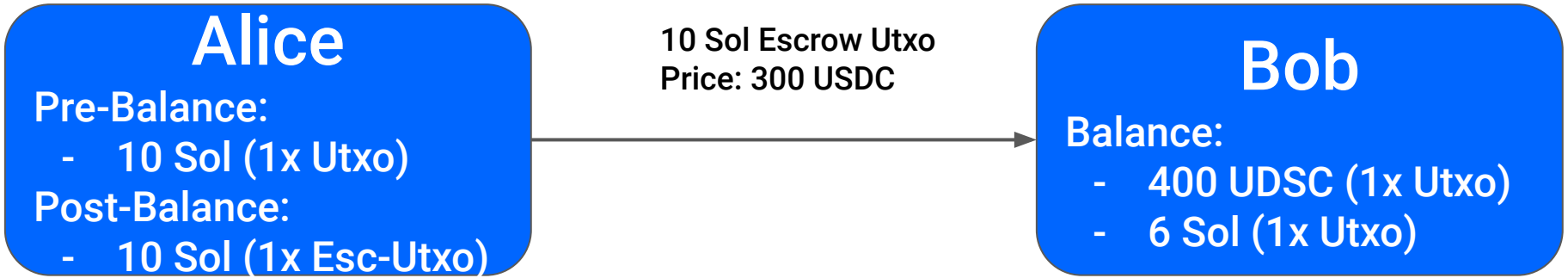
Alice trades with Bob.

10 Sol for 300 USDC.

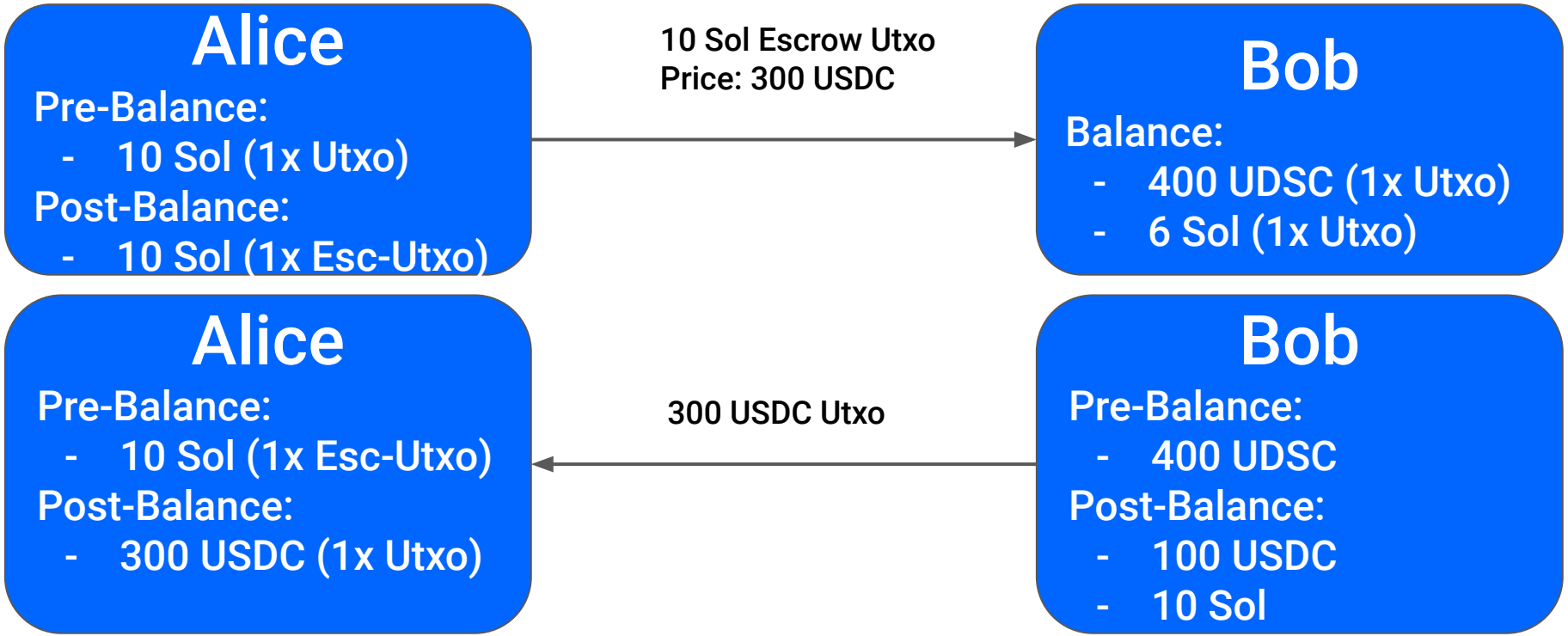
Properties:

1. Trustless swap (with shielded escrow)
2. Private amounts
3. No observable information other than that a swap transaction happened.

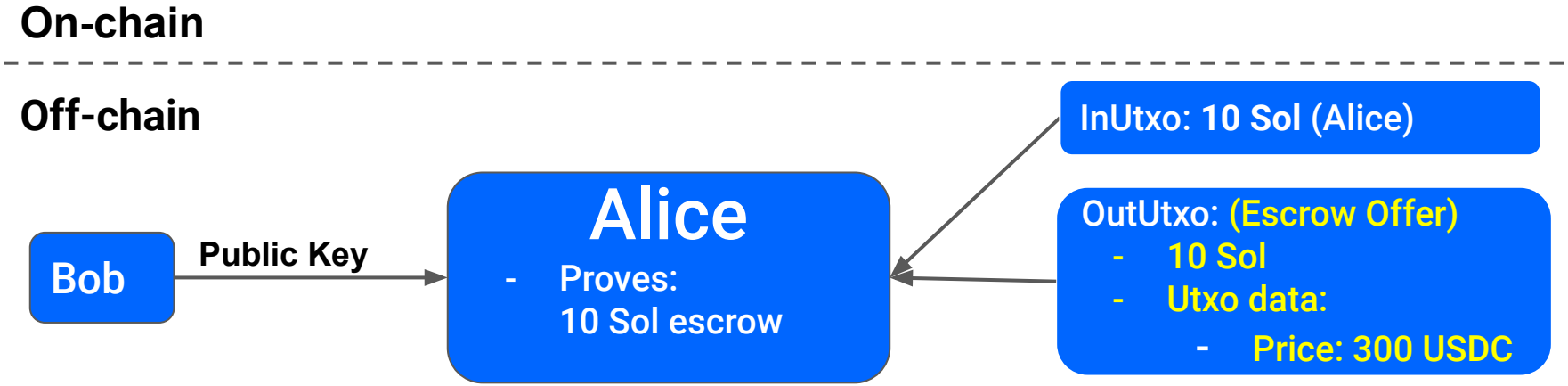
Shielded OTC Swap



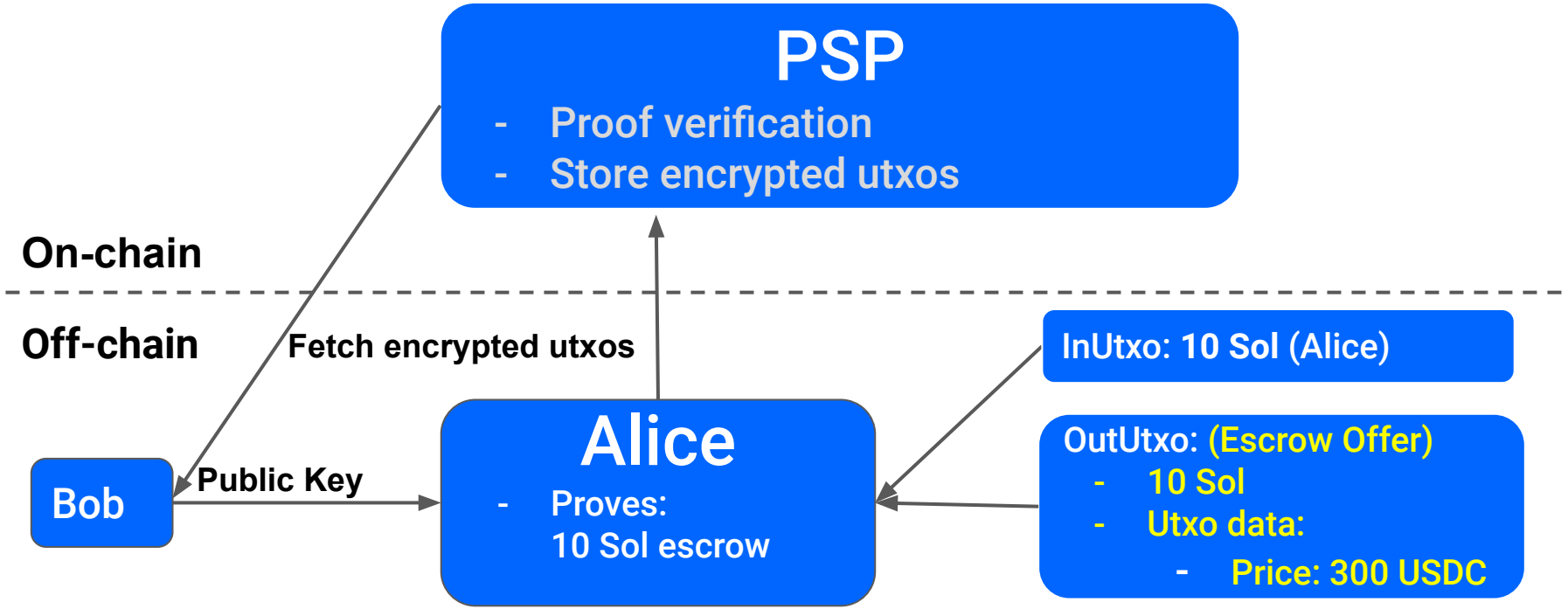
Shielded OTC Swap



OTC Swap: Make Offer



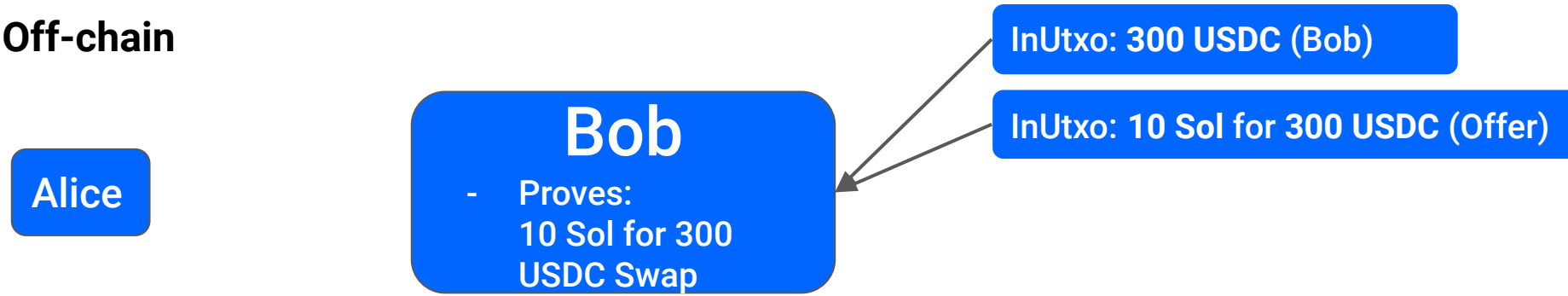
OTC Swap: Make Offer



Shielded OTC Swap: Take Offer

On-chain

Off-chain

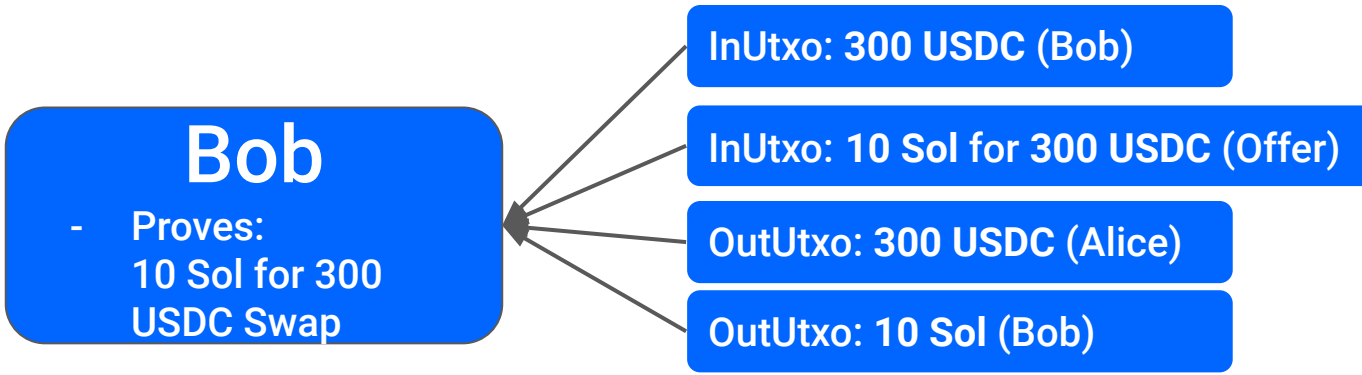


Shielded OTC Swap: Take Offer

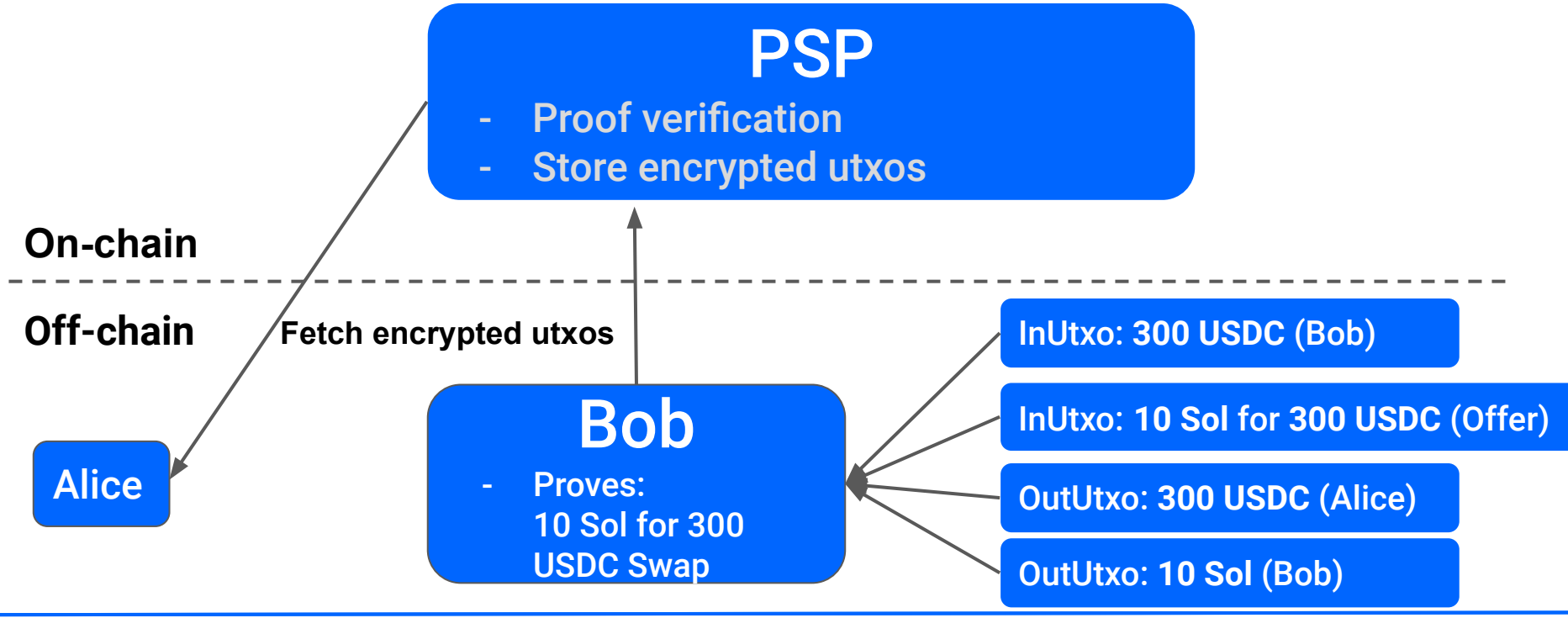
On-chain

Off-chain

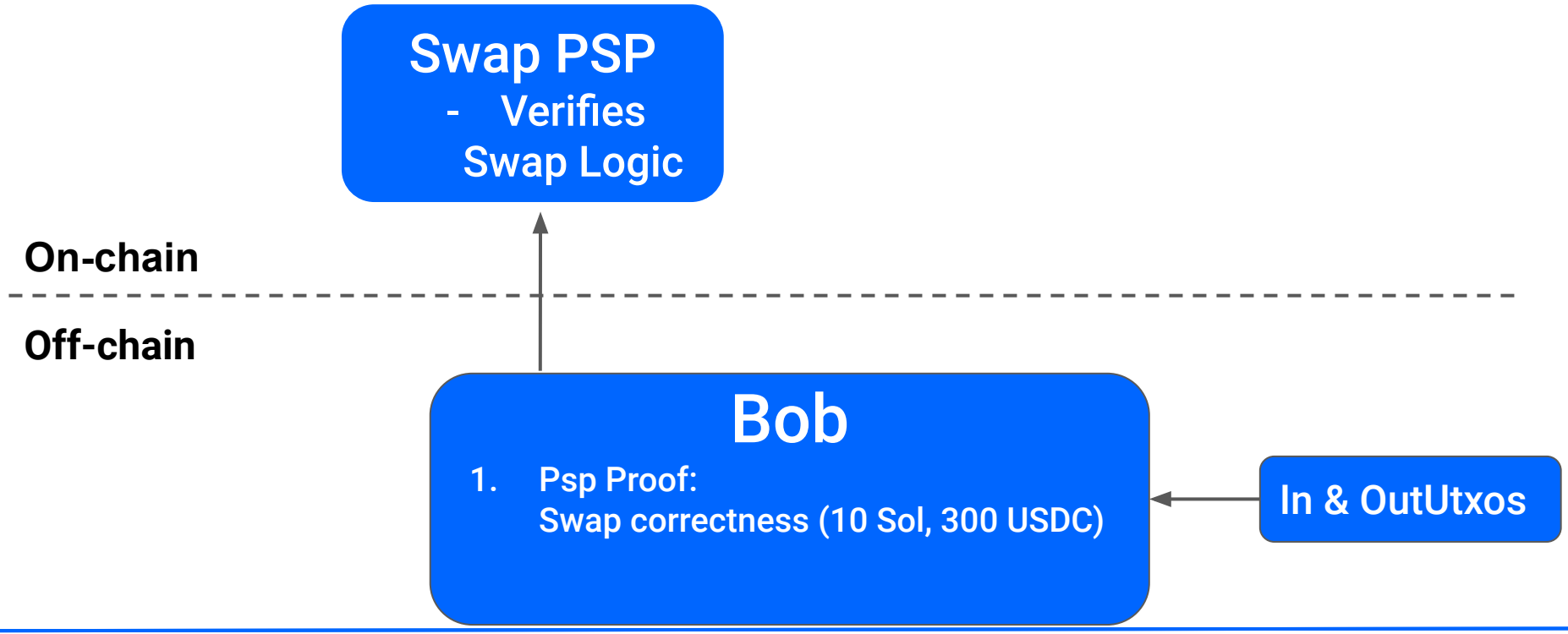
Alice



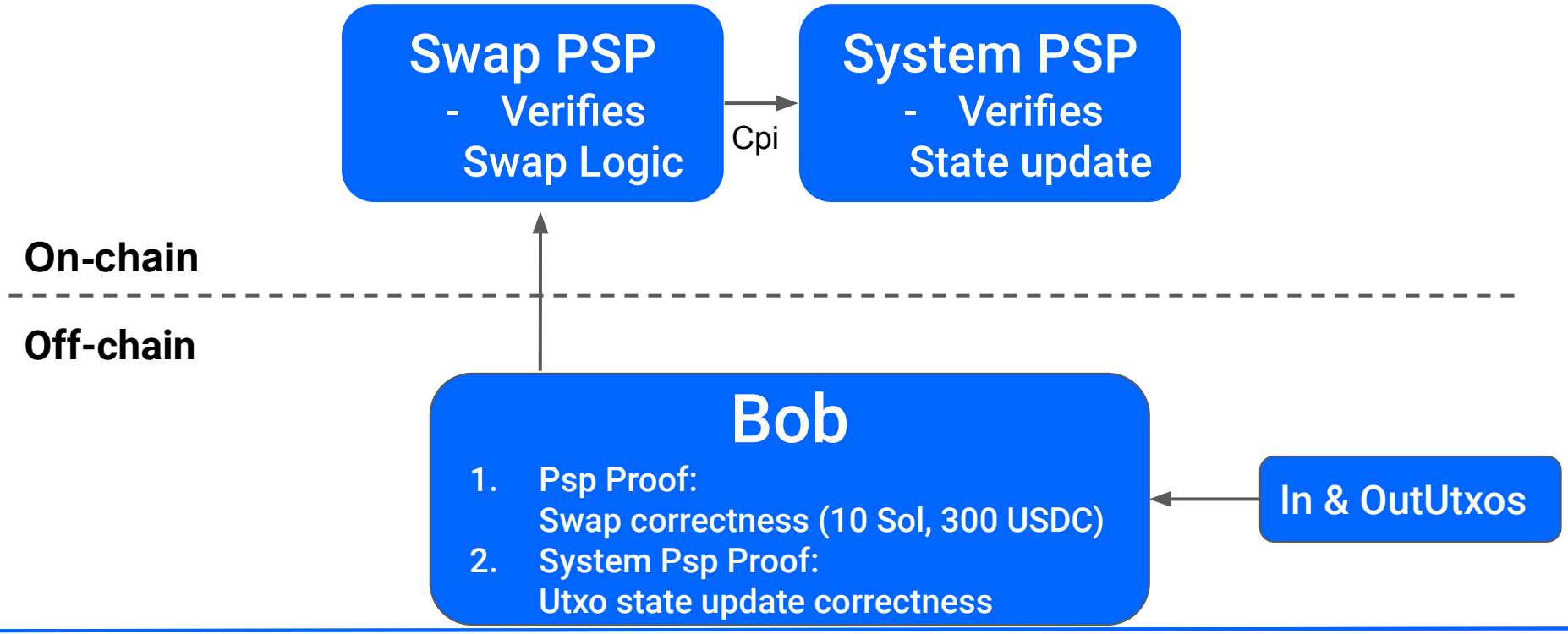
Shielded OTC Swap: Take Offer



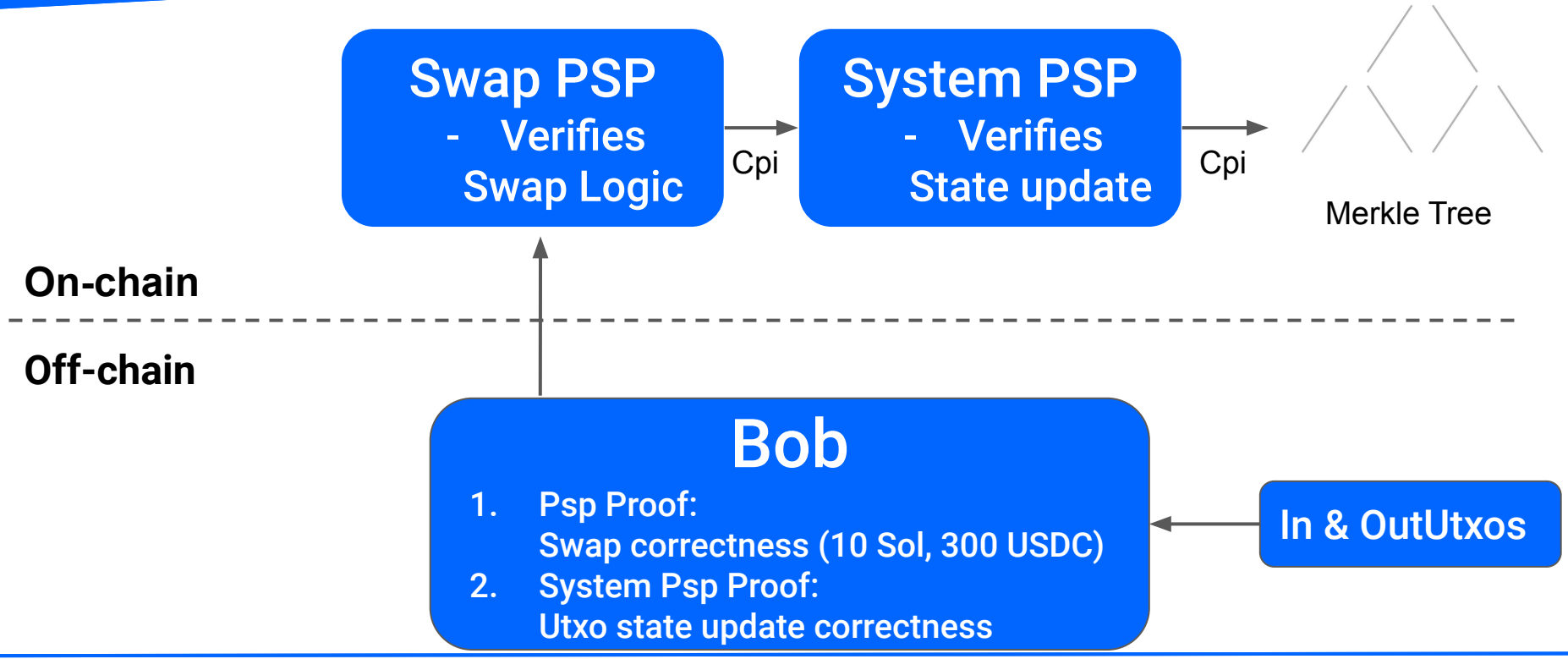
Swap PSP Proof



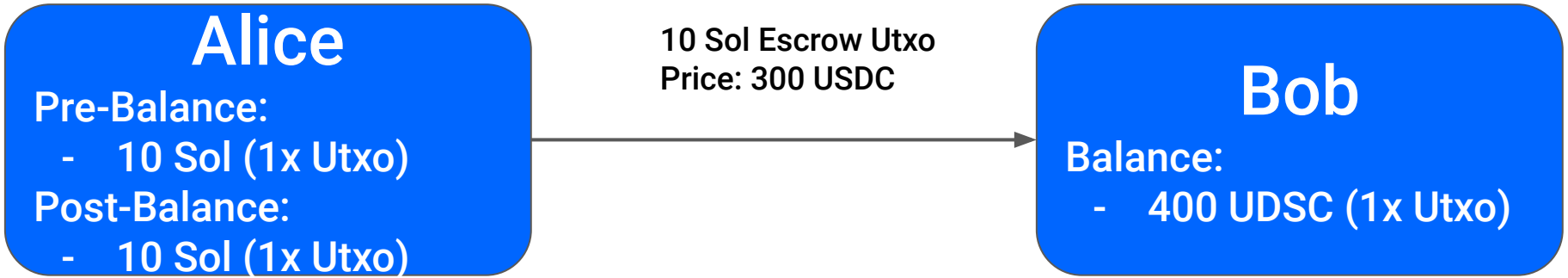
System Proof



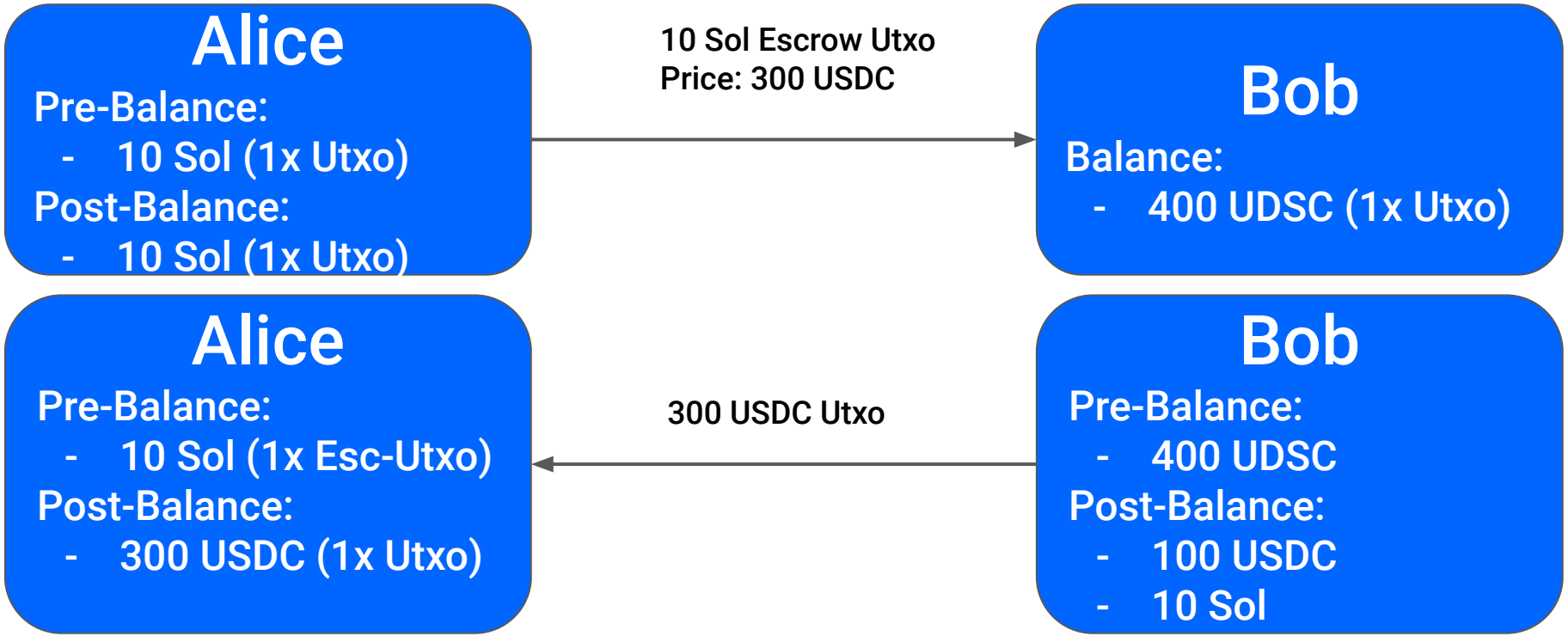
State Update



Shielded OTC Swap



Shielded OTC Swap



Practical Part

To code along:

- **git clone** https://github.com/Lightprotocol/breakpoint-workshop
- see readme for prerequisites

Practical Part

Build Commands:

- npm install
- npm run build

Repository Structure

1. Circuits
    a. Define PSP logic (.light)
    b. Written in macro-circom

2. Programs
    a. Verify PSP logic
    b. Anchor programs
    c. Generated files:
        i. verifying_key.rs
        ii. auto_generated_accounts.rs

3. Tests

Circuits: Instance

- Defines parameters for compiler
- Multiple instances per circuit (not supported yet)

```
#[instance]
{
    name: swaps,
}
```

Circuits: Entrypoint

- Defines entry point for circuit (main function)

```
#[entrypoint]
template <swaps> {

    ...

}
```

Circuits: Programming Model

```
utxoType SwapUtxo {...}

inUtxo offerUtxo { type: SwapUtxo, }

outUtxo offerRewardUtxo {type: native, ...}

#[entrypoint]
template swaps() {
        utxo offerUtxo;
        offerUtxo.check();
        utxo offerRewardUtxo;
        offerRewardUtxo.check();
}
```

Circuits: Utxo Type

- Defines a utxo type
- Multiple multiple utxos of the same type are supported

```
utxoType SwapUtxo {
    priceSol,
    priceSpl,
    splAsset,
    recipient,
    recipientEncryptionPublicKey,
}
```

Circuits: inUtxo

- Checks that one inUtxo is of type SwapUtxo

```
inUtxo offerUtxo {
        Type: SwapUtxo,
}
```

Circuits: outUtxo

- Checks that one outUtxo rewards the the seller as defined in the offer.

```
outUtxo offerRewardUtxo {
        type: native,
        enabled: takeOfferInstruction,
        checks: {
                amountSol == offerUtxo.priceSol,
                amountSpl == offerUtxo.priceSpl,
                assetSpl == offerUtxo.splAsset,
                publicKey == offerUtxo.recipient,
                blinding == offerUtxo.blinding,
        },
}
```

Circuits: outUtxo

- Checks that one inUtxo is an offer.
- Checks that one outUtxo rewards the the seller as defined in the offer.

```
#[entrypoint]
template swaps() {
    signal input takeOfferInstruction;
    utxo offerUtxo;
    offerUtxo.check();
    utxo offerRewardUtxo;
    offerRewardUtxo.check();
    …
}
```

Circuits: Instructions

- Zk-circuits always execute all possible paths.
- Multiply undesired paths with 0.
- Only one instruction variable is 1 all others are 0.

```
#[entrypoint]
template swaps() {
    signal input takeOfferInstruction;

    signal input takeCounterOfferInstruction;

    signal input cancelInstruction;
    ...
}
```

Tests: Take Offer

1. Create Alice and Bob as Users
2. Alice:
    a. Creates offer escrow utxo
3. Bob:
    a. Fetches offer escrow utxo
    b. Creates out utxos
    c. Generates system and PSP proofs
    d. Creates solana instructions
    e. Settles trade by invoking Swap PSP in 3 transactions

Tests: Code

Let’s dive into the code.

Questions

Questions?

Links

Thank you, for your Attention!

Links:

- **Examples:**
  https://github.com/lightprotocol/psp-examples
- **Docs:**
  https://docs.lightprotocol.com
- **Twitter:**
  @LightProtocol
  @ananas_light