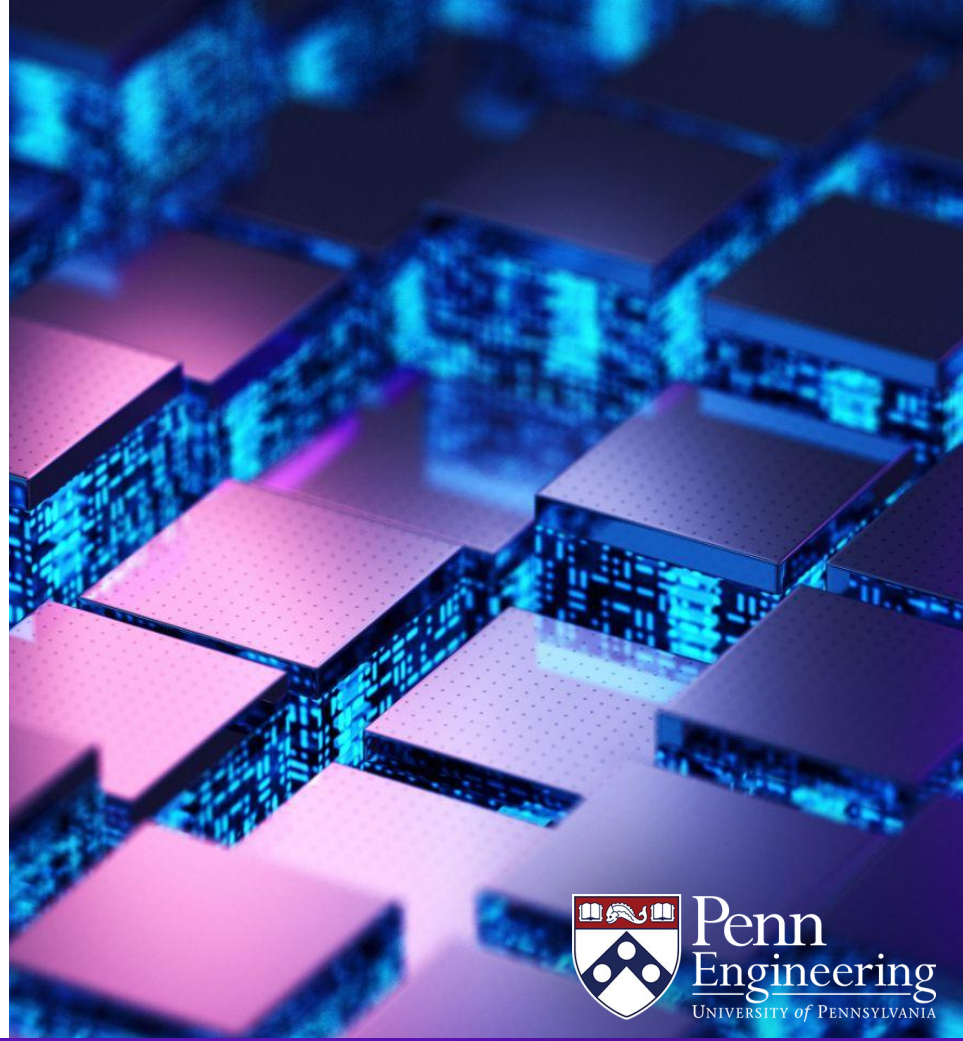
EAS 5830: BLOCKCHAINS

Bridges

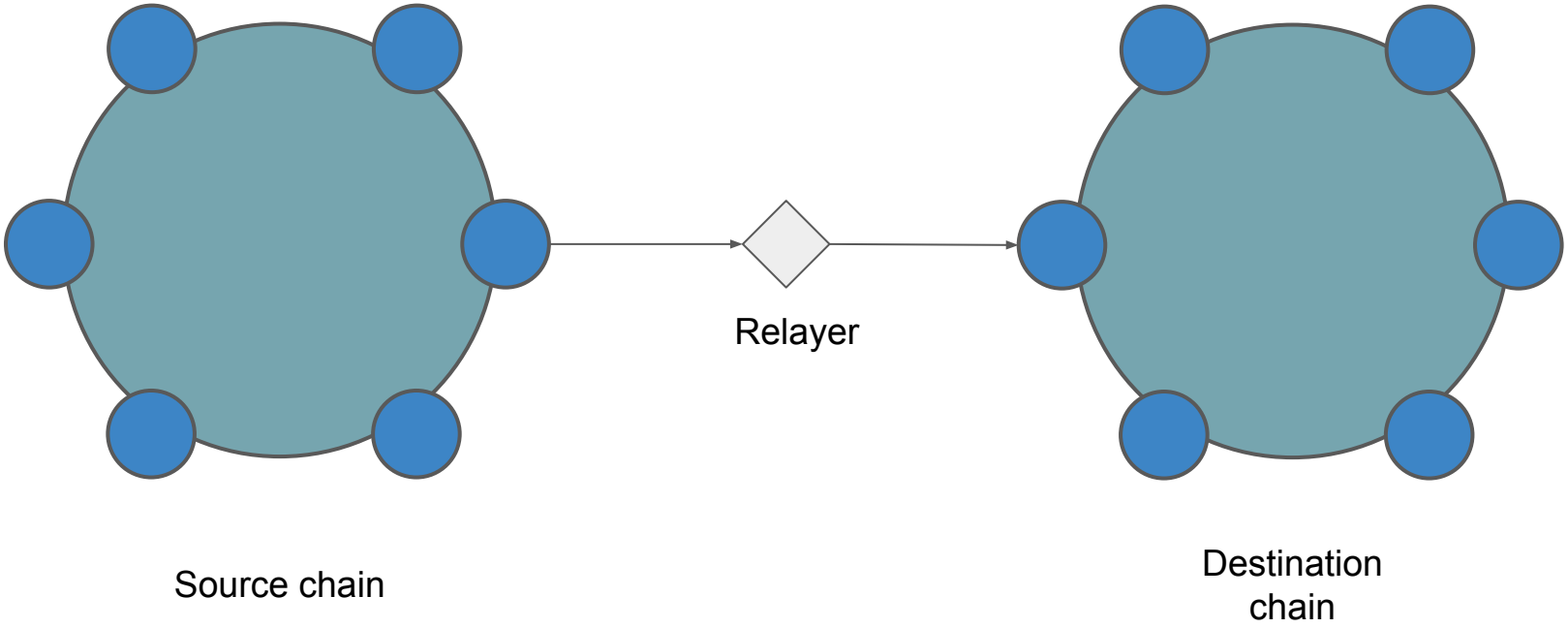
Professor Brett Hemenway Falk

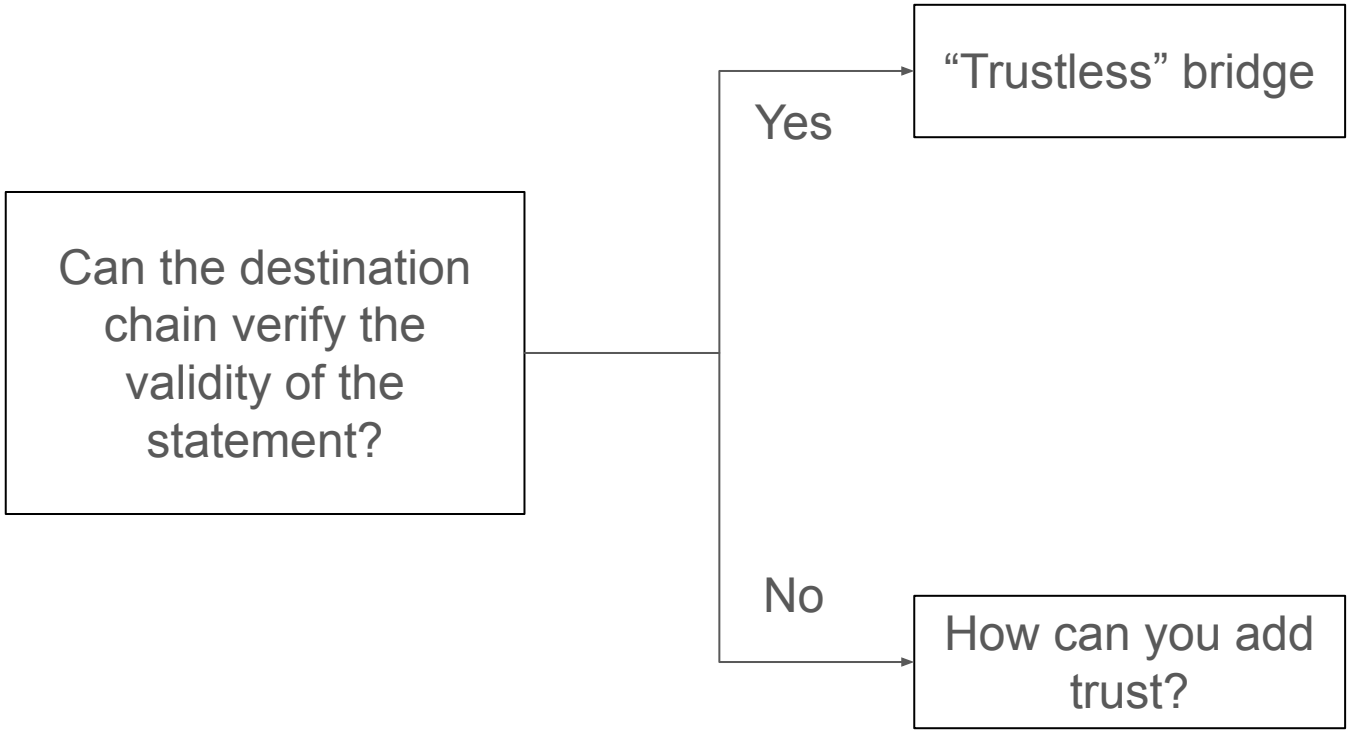


Problem

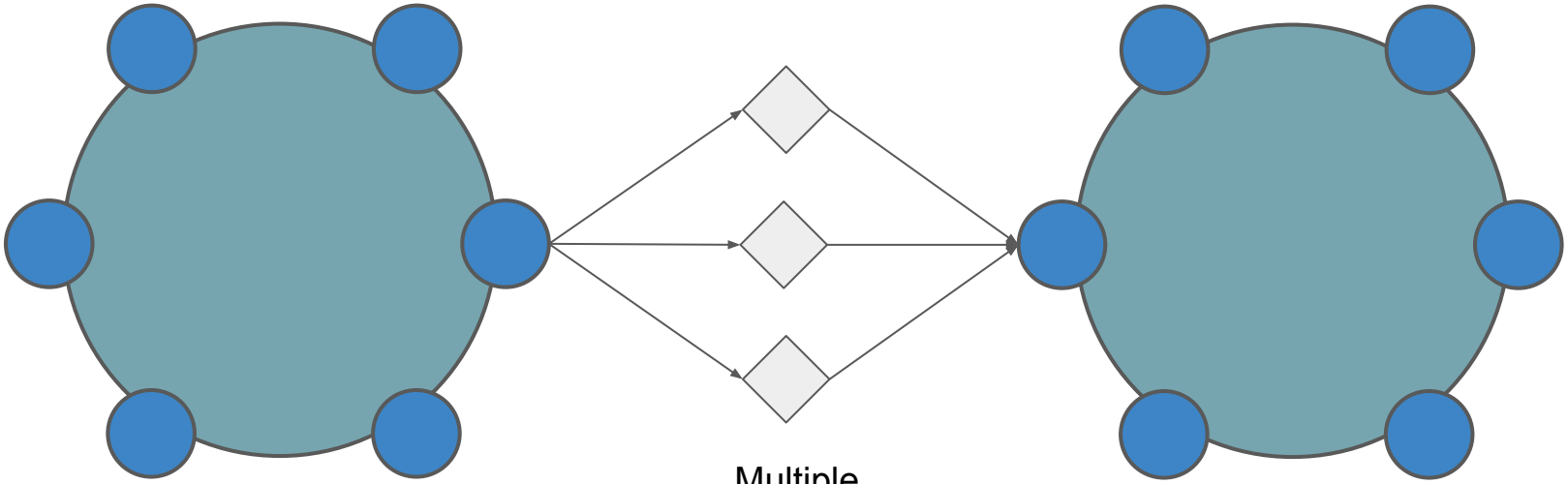
Blockchains can’t talk to each other

Bridging

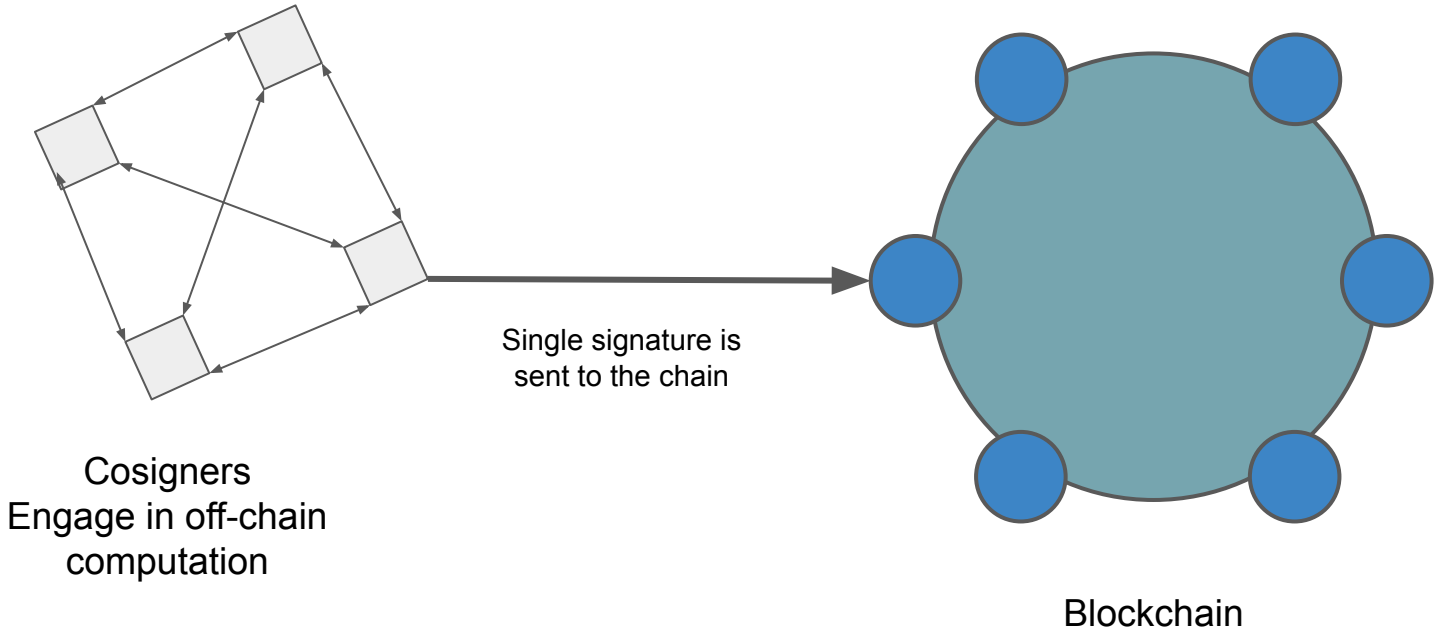



Can the destination chain verify the validity of the statement?
Yes
“Trustless” bridge
No
How can you add trust?

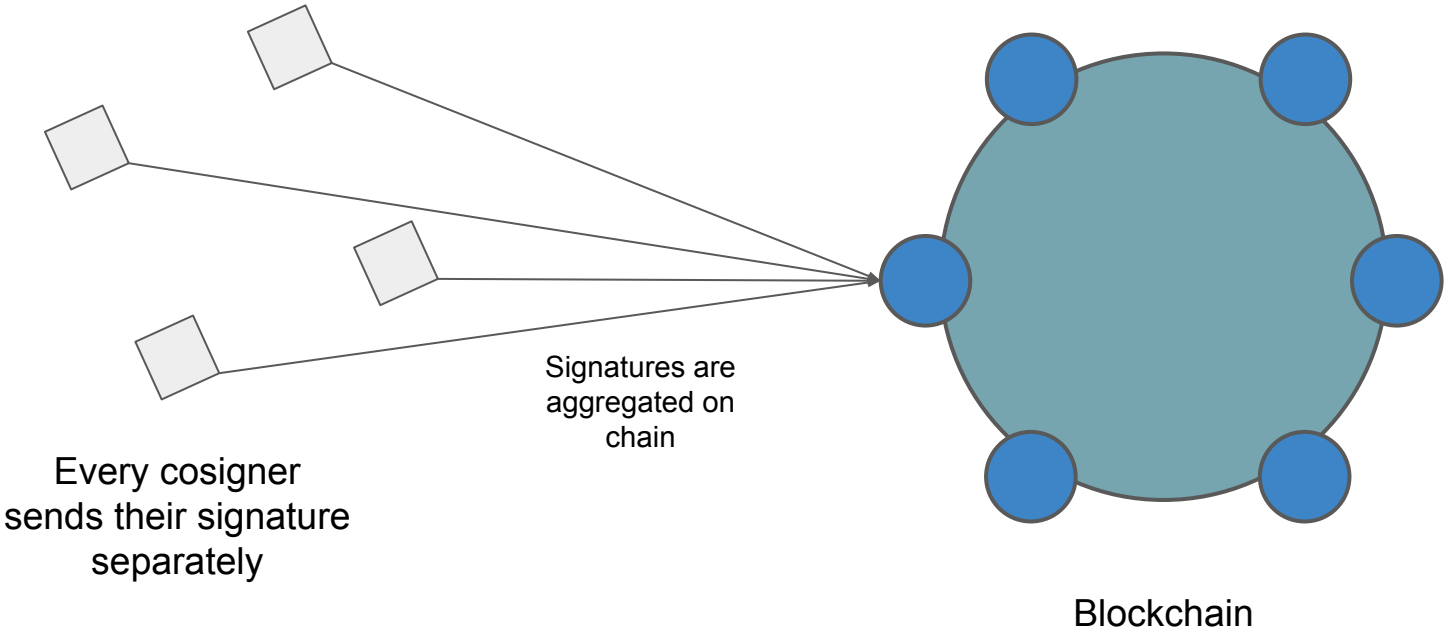
Adding trust



Threshold Signatures



Multi-sig


Signatures are aggregated on chain
Every cosigner sends their signature separately
Blockchain

Signatures

Threshold Signatures (TSS)

- Each participant has a “share” of a signing key
- Participants engage in (off-chain) secure computation protocol to generate a signature
- Resulting signature is indistinguishable from a regular signature
- E.g. Avalanche Bridge

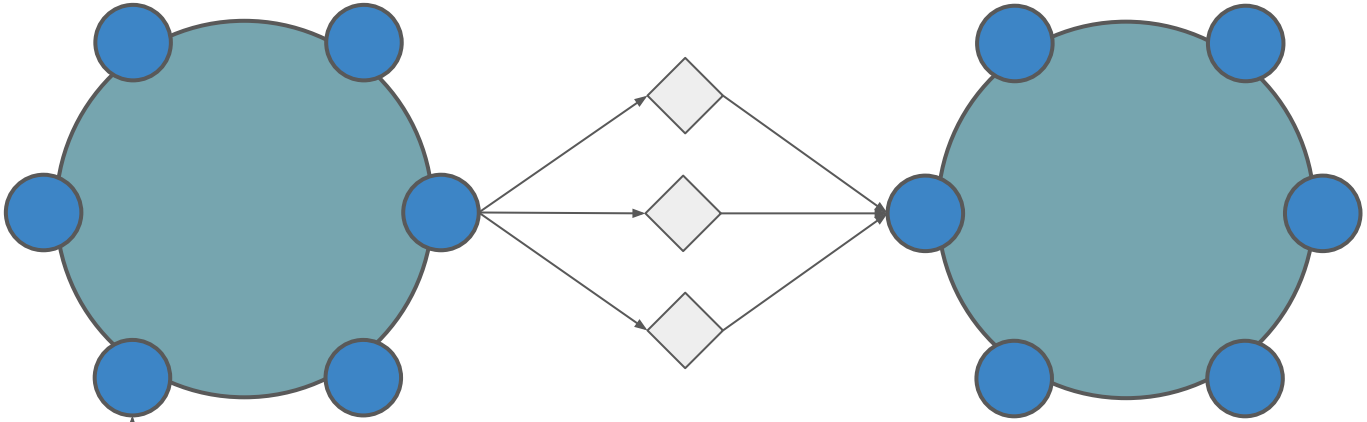
Multisig

- Each participant has a regular signing key
- There is an on-chain contract that aggregates signatures
- Underlying blockchain must support multi-sig accounts, or general contracts
- E.g. Wormhole bridge

Rotating relayers

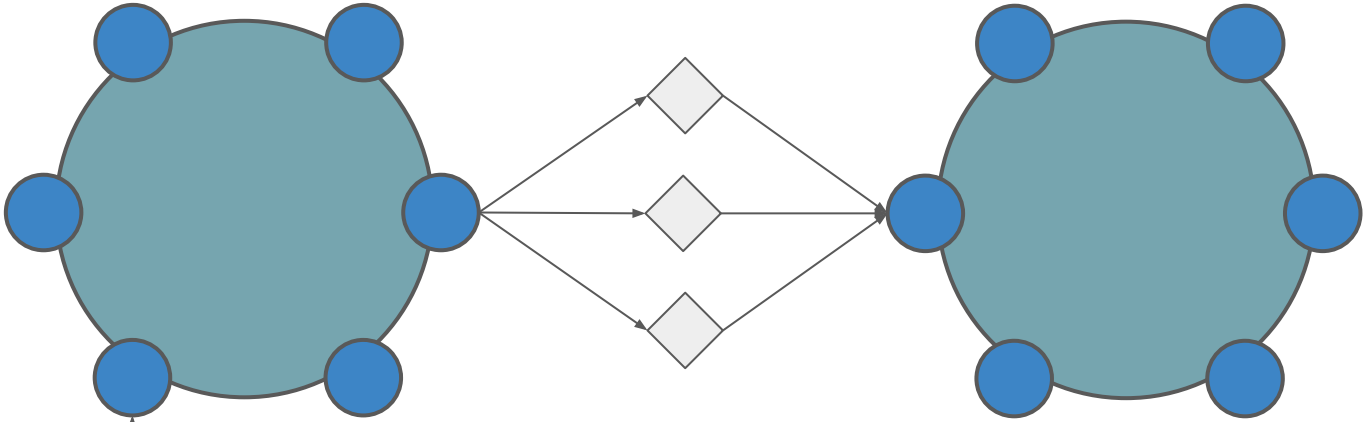
- Axelar
- Zetachain
- Thorchain
  - Doesn’t issue wrapped tokens
  - Trade native assets cross-chain

Bridging tokens



Send tokens to bridge (escrow) contract on the source chain

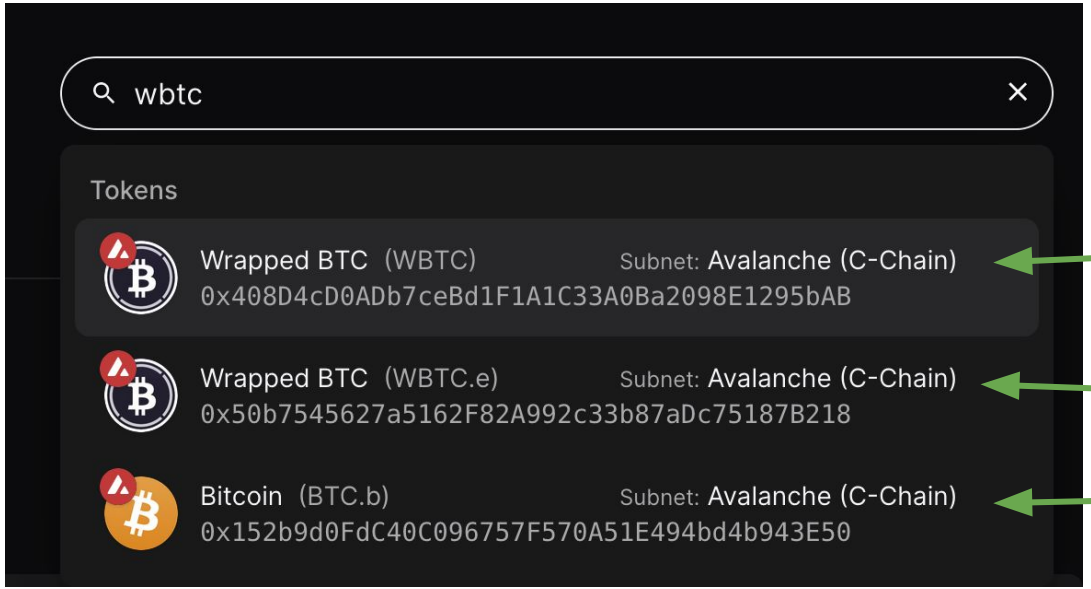
Bridging tokens



Send tokens to bridge (escrow) contract on the source chain

Mint new “wrapped” tokens on the destination chain

BTC on Avalanche



This was bridged from Ethereum by the old Avalanche bridge

WBTC.e is WBTC that was bridged from Ethereum

BTC bridged from Bitcoin

USDC on Avalanche

- USDC
  - “Real” USDC issued by Circle
- USDC.e
  - Wrapped USDC issued by the Avalanche bridge (backed by USDC on Ethereum)
- axlUSDC
  - Wrapped USDC issued by Axelar (backed by USDC on Ethereum)

```
/**
 * @dev Mint function used by bridge. Optional FeeAddress and FeeAmount parameters used to mint small percentage of transfered assets directly to bridge.
 * @param to Address to mint funds to.
 * @param amount Amount of funds to mint.
 * @param feeAddress Address to mint bridge fees to.
 * @param feeAmount Amount to mint as bridge fees.
 * @param feeAmount Amount to mint as bridge fees.
 * @param originTxId Transaction ID from external network that triggered this minting.
 */
function mint(
    address to,
    uint256 amount,
    address feeAddress,
    uint256 feeAmount,
    bytes32 originTxId
) public {
    require(bridgeRoles.has(msg.sender), "Unauthorized.");
    _mint(to, amount);
    if (feeAmount > 0) {
        _mint(feeAddress, feeAmount);
    }
    emit Mint(to, amount, feeAddress, feeAmount, originTxId);
}
```