Flow Cadence 101

Resource-oriented programming

Hsuan Lee

2022/03/16

Who am I?

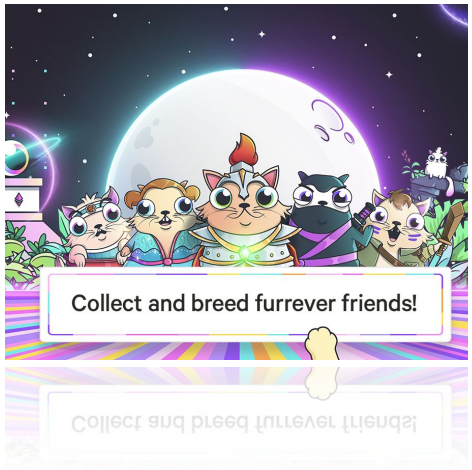
1. Co-founder & CEO @ portto
2. VP of Engineering @ Cobinhood & DEXON
3. Software Engineer @ 17 Media, Agoda, Yahoo
4. Bachelor & Master @ NTU, Electrical Engineering

Outline of this Sharing

1. Flow blockchain introduction
2. Resource-oriented programming
3. Flow accounts
4. Cadance contracts & transactions
5. Code examples


Flow

Dapper Labs



CryptoKitties

NBA Top Shot

Flow advantages

Security

Performance

DX

UX

Flow blockchain

Proof of Work

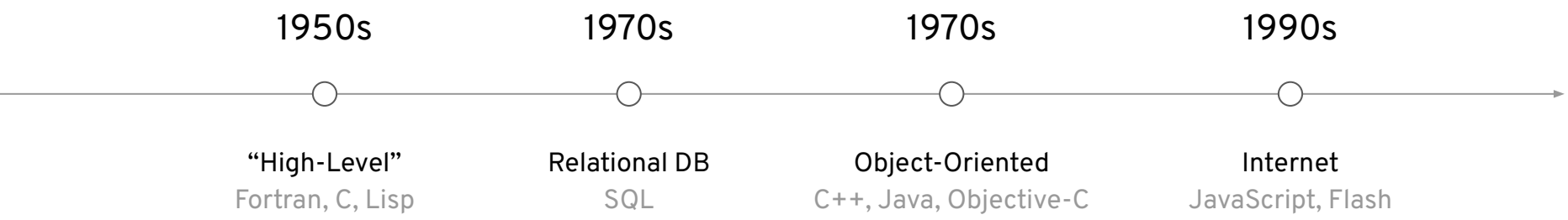
↓

Proof of Stake

Resource-oriented programming

- Introduced by Libra Move
- Made popular by Flow
- Picked up by Aptos (ex Libra/Diem)

History of programming languages



Programming on blockchain

- Introduced by Ethereum
- General purpose programming
- Suitable for
  - Transfer scarce assets
  - Control access
  - Provide auditable execution
  - Provide traceable proof??

What's wrong with current model?

- Centralized ledger on contracts
- Reduce chance of parallelism
- Data structure does not reflect ownership
- Huge attack surface
- Difficult to audit & analyze

Common attacks

1. Reentrance
    a. DAO hack:
    https://quantstamp.com/blog/what-is-a-re-entrancy-attack

    b. ERC777 + Uniswap / Lendf.me:
    https://www.abmedia.io/detailed-explanation-of-uniswaps-erc777-reentry-risk/

2. Abuse authorization
    a. Parity wallets got locked:
    https://github.com/openethereum/openethereum/issues/6995

    b. Centralized ERC20:
    https://etherscan.io/address/0xc12d1c73ee7dc3615ba4e37e4abfdbddfa38907e

Access Control
Who you are (list)

Scarce Assets
Data structure

Security
Contract level

Existing

v.s.

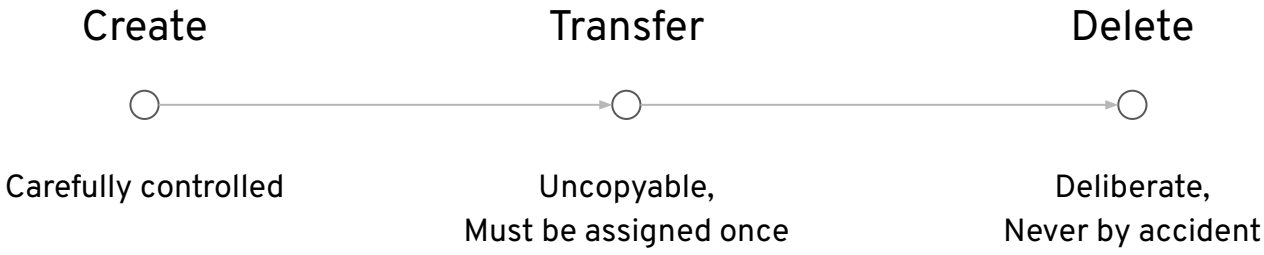
Access Control
What you have

Scarce Assets
Resources

Security
VM level

ROP

Resource lifecycle

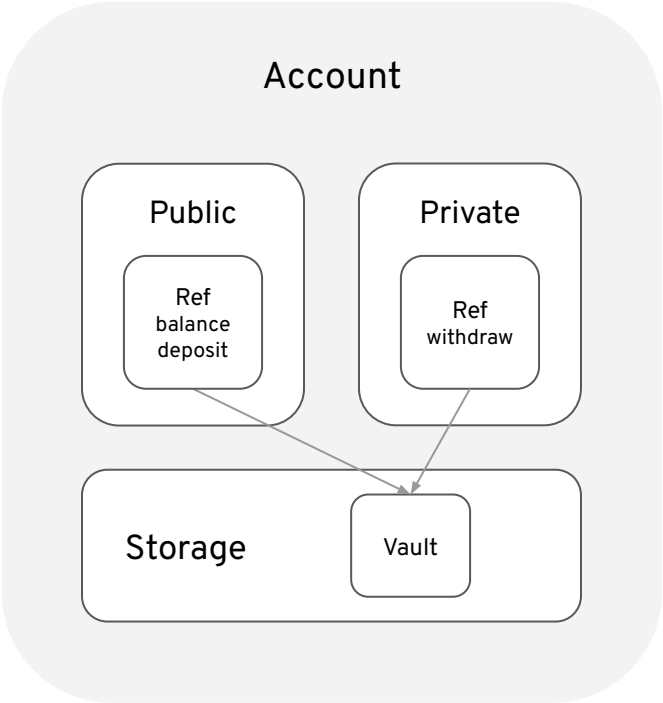


Ethereum fungible token

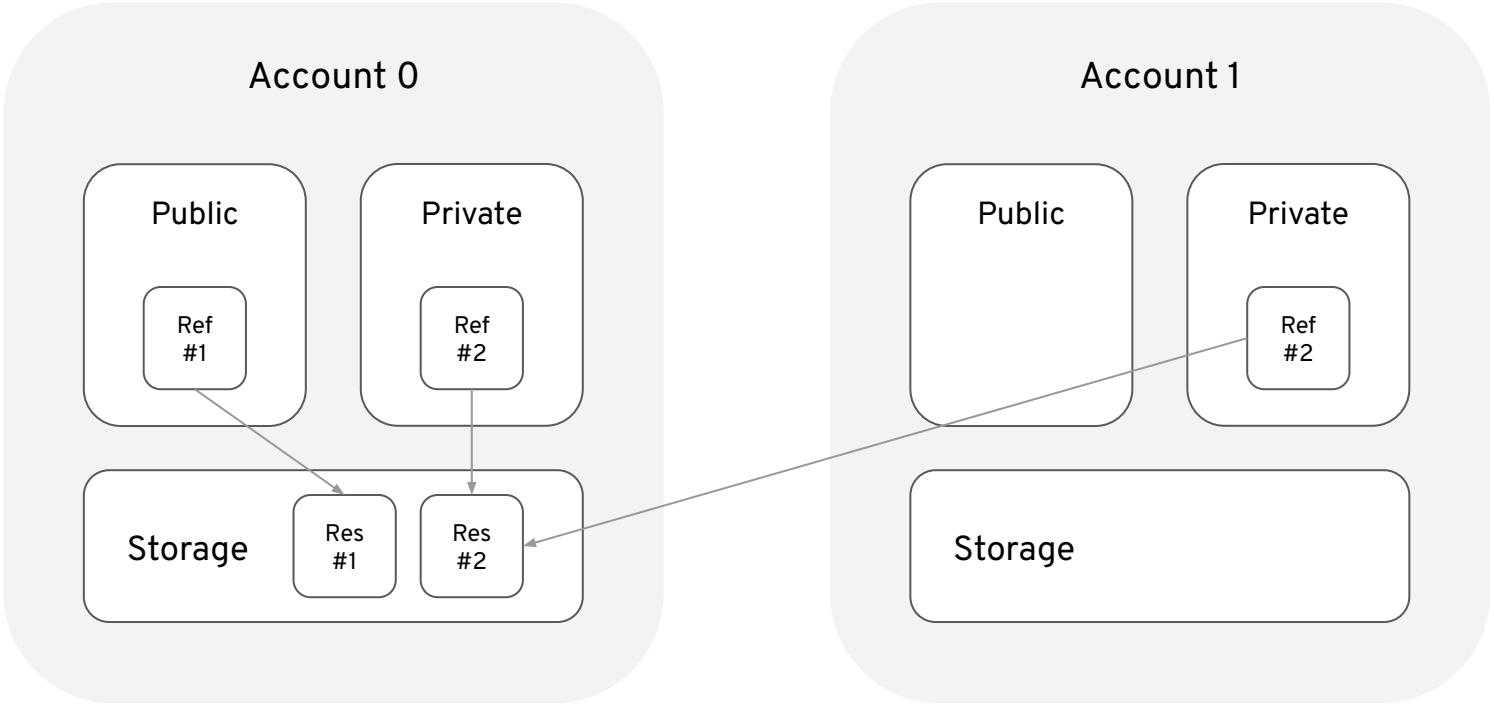
```
contract ERC20 {
   mapping (address => uint256) private _balances;

   function _transfer(address sender, address recipient, uint256 amount) {
      // ensure the sender has a valid balance
      require(_balances[sender] >= amount);

      // subtract the amount from the senders ledger balance
      _balances[sender] = _balances[sender] - amount;

      // add the amount to the recipient's ledger balance
      _balances[recipient] = _balances[recipient] + amount
   }
}
```

Flow fungible token

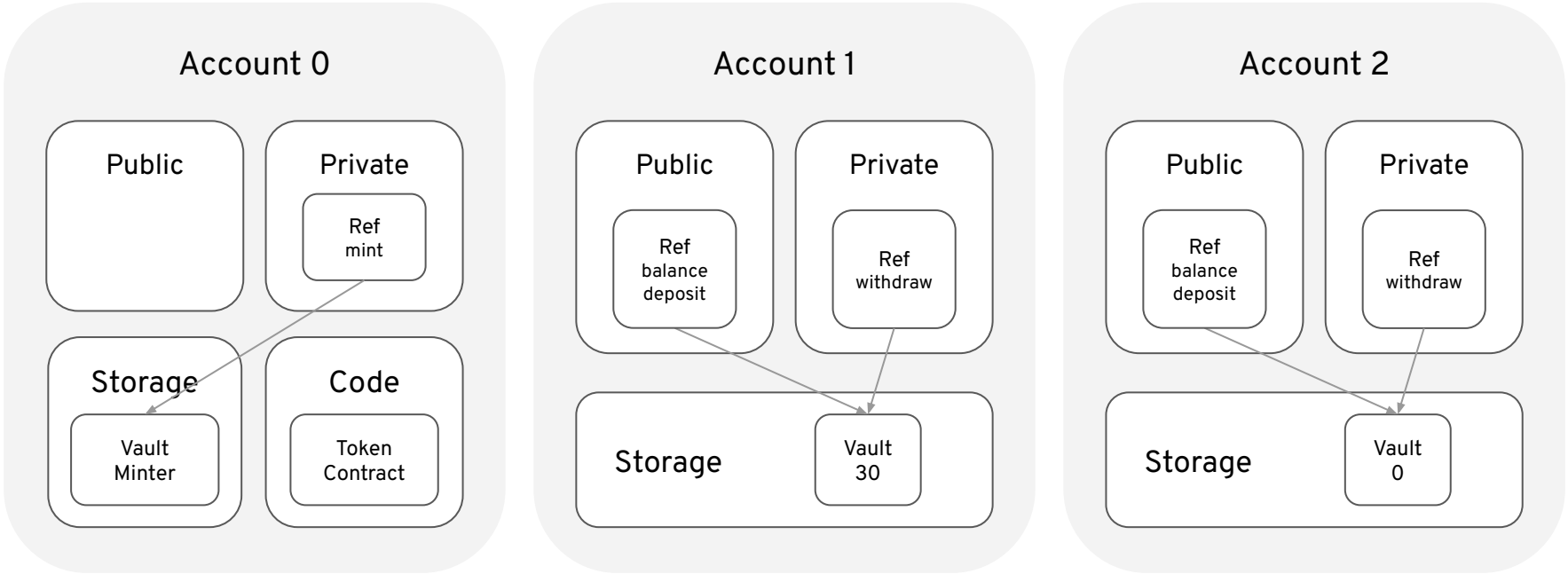
```
pub resource Vault: Provider, Receiver {
    pub var balance: UFix64

    init(balance: UFix64) {
        self.balance = balance
    }

    pub fun withdraw(amount: UFix64): @Vault {
        self.balance = self.balance - amount
        return ←create Vault(balance: amount)
    }

    pub fun deposit(from: @Vault) {
        self.balance = self.balance + from.balance
        destroy from
    }
}
```



Flow accounts and Resources



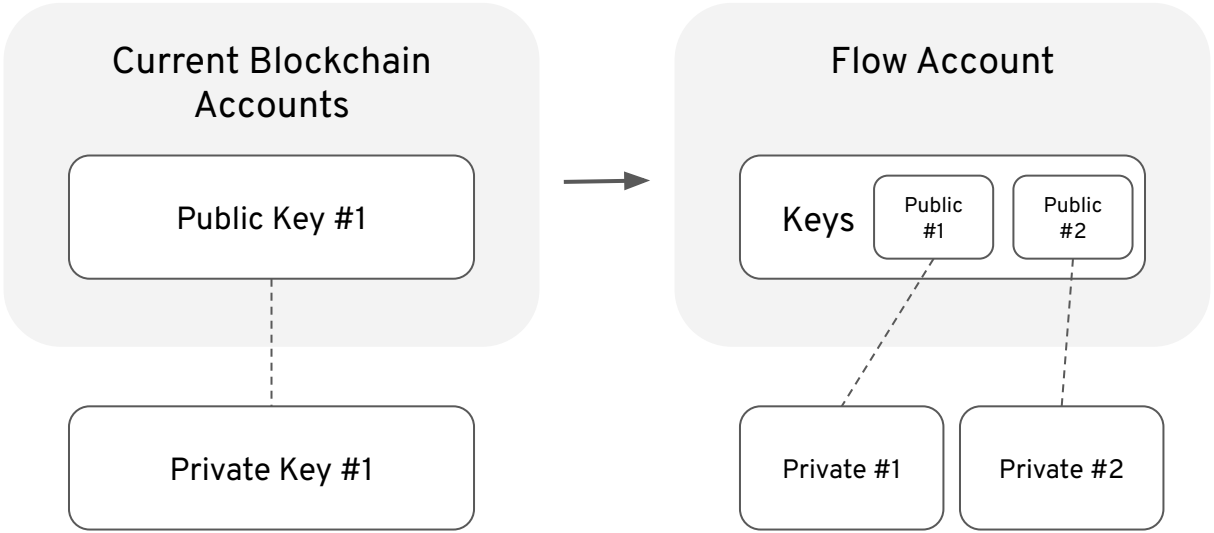
Flow fungible token



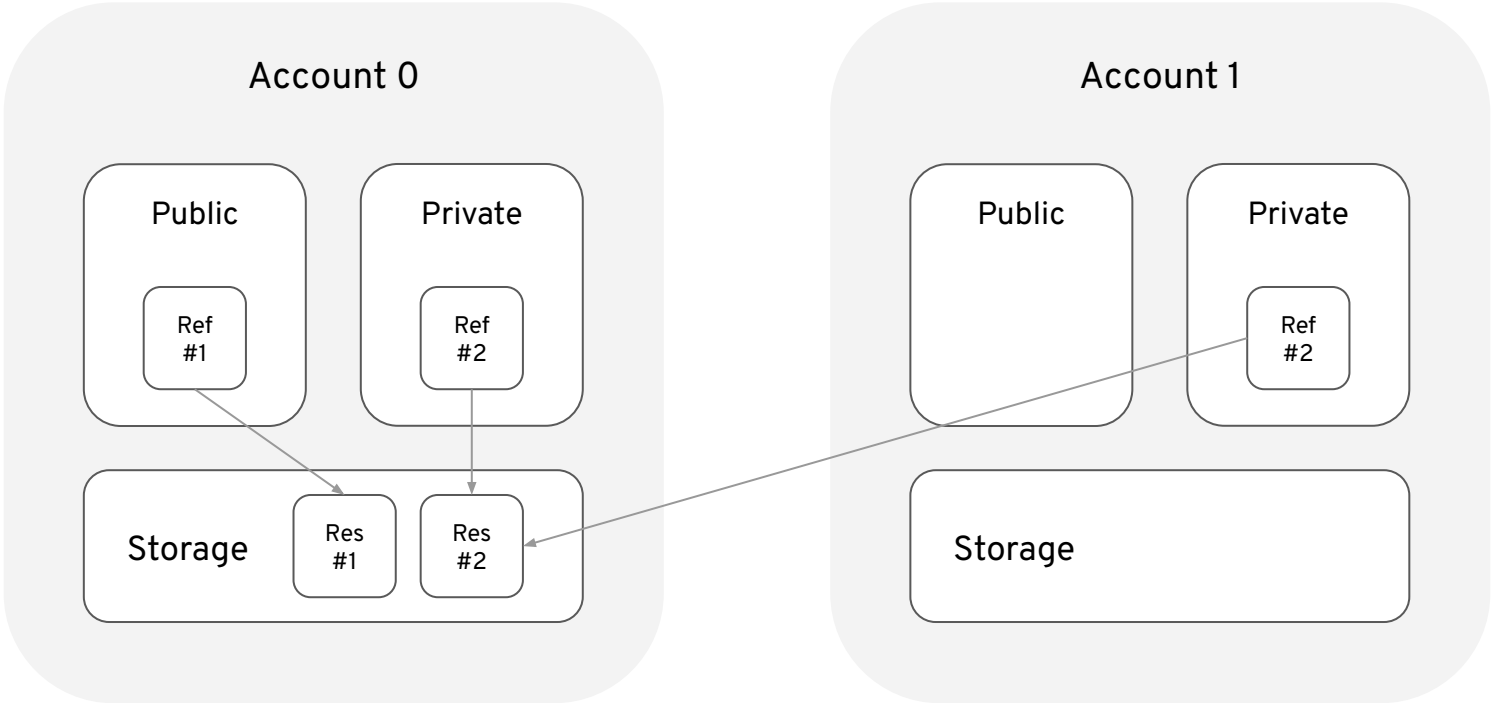
ROP Advantages

- Built-in security
- Less human error
- Better parallelism
- State rent made possible
- Resource hierarchy

Flow Accounts



Resources in Flow Accounts



Ethereum Transaction

0xa9059cbb000000000000000000000000090a805848b96c25488d9ed1f707b628dcaf665130000000000000000000000000000000000000000000000000001e7f483a

```
Function: transfer(address recipient, uint256 amount)

MethodID: 0xa9059cbb
[0]:  0000000000000000000000007dcda8fd073a1dab84976d7876634efbc61f5f15
[1]:  0000000000000000000000000000000000000000000000001b1ae4d6e2ef500000
```

View Input As Decode Input Data

Cadence Transactions

- Script
- Arguments
- Roles
  - Proposer
  - Authorizer(s)
  - Payer

Cadence TX Script

```
// Transfer Tokens
import ExampleToken from 0x02

transaction(amount: UFix64, to: Address) {
 var temporaryVault: @ExampleToken.Vault

 prepare(acct: AuthAccount) {
  let vaultRef = acct.borrow<&ExampleToken.Vault>(from: /storage/CadenceFungibleTokenTutorialVault)
    ?? panic("Could not borrow a reference to the owner's vault")

  self.temporaryVault ← vaultRef.withdraw(amount: amount)
 }

 execute {
  let recipient = getAccount(to)

  let receiverRef = recipient.getCapability(/public/CadenceFungibleTokenTutorialReceiver)
            .borrow<&ExampleToken.Vault{ExampleToken.Receiver}>()
            ?? panic("Could not borrow a reference to the receiver")

  receiverRef.deposit(from: ←self.temporaryVault)
 }
}
```

Cadence Contracts

- Import
- Resource
- Interface
- Initializer

Code Example

https://play.onflow.org/50745fb6-77d5-4510-adfc-cf448fb043e1

Flow Project + CLI

- ./contracts/*.cdc
- ./transactions/*.cdc
- ./scripts/*.cdc
- ./flow.json

Resource-oriented resources

- Getting Started With Move
- Cadence Language Reference
- Cadence Fungible Tokens
- Alchemy Flow API

Questions?

@hleetw