DAO Governance Contracts

Dan Boneh
Stanford University
Recap: Running a program on a blockchain (DAPP)

Consensus layer (beacon chain)

Compute layer (execution chain): The EVM

Create a DAPP

Program code

state₀ → Tx₁ → state₁ → Tx₂ → state₂ → ...
Recap: Ethereum World State

World state: set of accounts identified by 32-byte address.

Two types of accounts:

(1) owned accounts (EOA): controlled by a signing key pair (pk, sk).
   sk: owned by account owner

(2) contracts: controlled by code (set by creator)
## Recap: Data associated with an account

<table>
<thead>
<tr>
<th>Account data</th>
<th>Owned</th>
<th>Contracts</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>address</strong> (computed):</td>
<td>H(pk)</td>
<td>H(CreatorAddr, CreatorNonce) (different with CREATE2)</td>
</tr>
<tr>
<td><strong>balance</strong> (in Wei):</td>
<td>balance</td>
<td>balance (10^{18} Wei = 1 ETH)</td>
</tr>
<tr>
<td><strong>code:</strong></td>
<td>⊥</td>
<td>EVM bytecode</td>
</tr>
<tr>
<td><strong>storage</strong> (state):</td>
<td>⊥</td>
<td>S[0], ..., S[2^{256} - 1]</td>
</tr>
</tbody>
</table>
Recap: State transitions via Tx

Transactions: signed data by initiator

- **To:** 32-byte address of target (0 → create new account)
- **From, [Signature]:** initiator address and signature on Tx (if owned)
- **Value:** # Wei being sent with Tx
- **Tx fees (EIP 1559):** `gasLimit, maxFee, maxPriorityFee` (later)
- if `To ≠ 0:` **data** (what function to call & arguments)
- if `To = 0:` create new contract `code = (init, body)`
- …
mapping (address => uint256) internal _balances; // keeps track of balances

    // _balances[owner] is the balance of address owner

function transfer(address _to, uint256 _value) external returns (bool);

...

function totalSupply() external view returns (uint256);

function balanceOf(address _owner) external view returns (uint256);
contract ERC20 is IERC20 {

    mapping (address => uint256) internal _balances;

    function transfer(address _to, uint256 _value) external returns (bool) {
        require(_balances[msg.sender] >= _value, "ERC20_INSUFFICIENT_FUNDS");
        _balances[msg.sender] -= _value;
        _balances[_to] += _value;
        emit Transfer(msg.sender, _to, _value); // write log message
        return true;
    }
}

Tokens can be minted by a function mint(address _to, uint256 _value) onlyOwner;
DAPPs cannot keep secrets!

Anyone can read the storage array of a DAPP:
A bit about EVM mechanics
Write code in Solidity (or another front-end language)⇒ compile to EVM bytecode (some projects use WASM or BPF bytecode)⇒ validators use the EVM to execute contract bytecode in response to a Tx
The EVM

Stack machine
- code can **CREATE** or **CALL** another contract

In addition: several types of memory
- Persistent storage (on blockchain):  **SLOAD**, **SSTORE**  (expensive)
- Volatile memory (for single Tx):  **MLOAD**, **MSTORE**  (cheap)
- LOG0(data): write data to log (a part of every block)
- CallData: arguments in Tx  (persistent, but only readable by current Tx)
Every instruction costs gas, examples:

**MLOAD, MSTORE**: 3 gas  (cheap)

**SSTORE addr** (32 bytes), **value** (32 bytes)

- zero $\rightarrow$ non-zero: 20,000 gas
- non-zero $\rightarrow$ non-zero: 5,000 gas  (for a cold slot)
- non-zero $\rightarrow$ zero: 15,000 gas refund  (example)

**CREATE**: 32,000 + 200 $\times$ (code size) gas;

**CALL** gas, addr, value, args
Why charge gas?

- Tx fees (gas) prevents submitting Tx that runs for many steps.
- During high load: block proposer chooses set of Tx from mempool that maximize its income.

Old EVM: (prior to EIP1559, live on 8/2021)

- Every Tx contains a gasPrice "bid" (gas $\rightarrow$ Wei conversion price)
- Producer chooses Tx with highest gasPrice \( \left( \max \ \sum (\text{gasPrice} \times \text{gasUsed}) \right) \)
  \( \Rightarrow \) not an efficient auction mechanism (first price auction)
Gas prices spike during congestion

Average Tx fee in USD

congestion
EIP1559 goals (informal):

• users incentivized to bid their true utility for posting Tx,
• block proposer incentivized to not create fake Tx, and
• disincentivize off chain agreements.

[ Transaction Fee Mechanism Design, by T. Roughgarden, 2021 ]
Gas calculation: EIP1559  (since 8/2021)

Every block has a “baseFee”:

the **minimum** gasPrice for all Tx in the block

baseFee is computed from **total gas** in earlier blocks:

- earlier blocks at gas limit (30M gas) $\Rightarrow$ base fee goes up 12.5%
- earlier blocks empty $\Rightarrow$ base fee decreases by 12.5%

If earlier blocks at “target size” (15M gas) $\Rightarrow$ base fee does not change
EIP1559 Tx specifies three parameters:

- **gasLimit**: max total gas allowed for Tx
- **maxFee**: maximum allowed gas price (max gas → Wei conversion)
- **maxPriorityFee**: additional “tip” to be paid to block proposer

Computed **gasPrice** bid:

\[
gasPrice \leftarrow \min(maxFee, \ baseFee + maxPriorityFee)\]

Max Tx fee: \( \text{gasLimit} \times gasPrice \)
Gas calculation

\[ \text{gasUsed} \leftarrow \text{gas used by Tx} \]

- Send \( \text{gasUsed} \times (\text{gasPrice} - \text{baseFee}) \) to block proposer
- BURN \( \text{gasUsed} \times \text{baseFee} \)

\( \Rightarrow \) total supply of ETH can decrease
Back to DAOs
Decentralized Orgs (DAO)

What is a DAO?

• A Dapp deployed on-chain at a specific address

• Anyone (globally) can send funds to DAO treasury

• Anyone can submit a proposal to DAO
  ➔ participants vote
  ➔ approved → proposal executes

snapshot.org
Examples of DAOs

- There are over 12000 DAOs managed on Snapshot
- **Collector DAOs:** PleasrDAO, flamingoDAO, ConstitutionDAO, ...
  (see art collection at [https://gallery.so/pleasrdao](https://gallery.so/pleasrdao))

**PleasrDAO:** 103 members.
- Manages a treasury, has full time employees.
- Deliberations over what to acquire over telegram.
Examples of DAOs

- There are over 12000 DAOs managed on Snapshot
- Collector DAOs: PleasrDAO, flamingoDAO, ConstitutionDAO, ...
- **Grants DAOs**: gitcoin (83K members), ...

Proposal ID 21: This proposal looks to ratify the allocation of 30,000 GTC from the Community Treasury to the MMM workstream.
Examples of DAOs

• There are currently about 12000 DAOs managed on Snapshot
• Collector DAOs: PleasrDAO, flamingoDAO, ConstitutionDAO, ...
• Grants DAOs: gitcoin, ...
• **Protocol DAOs**: manages operation of a specific protocol
  Uniswap DAO (74K), Compound DAO (8K), ...
• **Social DAOs**: FWB, ...
• **Investment DAOs**: many
How to build a DAO

Three key decisions:

• What is the community for the DAO?

• How is membership managed?
  Many available tools: Syndicate, Juicebox, Colony, ...
  can anyone join, or does the community vote?

• How to do governance? What is controlled by governance?
Many DAO governance experiments

Who can vote?    How to vote?    What voting mechanism?

**Lightspeed Democracy: What web3 organizations can learn from the history of governance**

by Andrew Hall and Porter Smith
June 29, 2022

DAOs: a platform for experimenting with governance mechanisms
Governance methods

One token one vote: (most common)
• Members receive tokens based on their contribution
• Everyone can vote

In reality: low participation rate (8%)

⇒ delegation

proposal 21 (tally.com)
Delegation example: element

≈300 addresses delegated tokens to this address
On-chain vs. Off-chain voting

On chain voting: (e.g., using Compound’s Governor bravo)
• Ensures secure vote tallying and proposal execution
  ... but casting every vote costs gas

Off chain voting: (e.g., using snapshot) [vote signaling]
• Proposals and votes are recorded on IPFS
• Problem: how to securely convey results to on-chain DAPP ??
  • one option: SafeSnap. [ Reality.eth ⇒ Gnosis safe ]
DAO Governance Contracts

For on-chain voting
Governor Bravo (compound)

**Governor alpha** (Jan. 2020) updated to **Bravo** (March 2021):

- A collection of DAPPs used for governing Compound
- Forked and used by many other protocols

Other alternatives:
- **OpenZeppelin Governor** protocol
The life cycle of a proposal

- **Proposal Created**: 2 Days
- **Voting Active**: 3 Days
- **Voting Ends**: Outcome Decided
- **Timelock**: 2 Days

States:
- **Review** → **Active**
- **Active** → **Succeeded** → **Queued**
- **Active** → **Defeated**
- **Canceled**
- **Executed**
Governor Bravo

Implements an ERC-20 governance token. Let’s call it GOV.

• An address that owns GOV tokens can vote on proposals

function castVote(uint proposalID, uint8 support)

(cannot vote partially)
Governor Bravo

Implements an ERC-20 governance token. Let’s call it GOV.

... or delegate its GOV tokens to another address

```
function delegate(address delegatee)
```

can re-delegate at any time by calling function again

An address can only delegate all its tokens to one address at a time

... but I can hold my GOV tokens at multiple addresses
Creating a proposal

```solidity
function propose(
    address[] targets,
    uint[] values,
    string[] signatures,
    bytes[] calldatas,
    string description
) returns (uint)
{
    // proposal description (human readable)
}
```

The function to call on governor contract

The ID of newly created proposal

If proposal passes, how to execute it:
- what functions to call (actions)
- and with what arguments
Global contract parameters

- **quorumVotes**: the minimum # of votes for a proposal to pass
- **proposalThreshold**: min # votes needed to create a proposal
- **votingDelay**: # blocks to wait until voting can begin after a proposal is created (e.g., two days)
- **votingPeriod**: # blocks when voting is open (e.g., three days)

All these parameters can be changed by governance in Bravo
Proposal execution

Once proposal gets enough votes and delays expire:

- anyone can call: `function execute(uint proposalID)`

  ⇒ prescribed functions in proposal get called
The limits of democracy

Anytime before execution

- “guardian” can call `cancel()` to cancel a proposal

Who is this guardian??

- set at Governor contract creation time
- can abdicate at any time by calling `_abdicate()`
Hot topics in blockchains
• **Private DAO participation:** keep membership list private
• **Private voting:** keep who voted how on each proposal private
• **Privately delegate** voting rights
• **Private treasury**

... while complying with all relevant laws.

Some of these questions are solved by general privacy platforms such as **Aztec, Aleo**, and others.
Private voting in Nouns

Mandated Round: Private Voting Research Sprint

Proposed by delegate.el4d.eth at 0x7b1ba

For 166
Against 75
Abstain 19

https://nouns.wtf/vote/216
Hot topics (what I work on)

- Scaling the blockchain
- Preventing theft by enabling asset recovery: ERC-20R and ERC-721R
- Privacy on the blockchain: businesses cannot use otherwise (with compliance)
- Efficient interoperability between blockchains
- Managing Maximal Extractable Value (MEV)
- Zero knowledge proofs of solvency
The interoperability problem

Can I use Joepegs??

... but today not easy

wrap NFT (using a bridge)

Ethereum

Avalanche

Cosmos

Joepegs Marketplace
END OF LECTURE