Audit

By OCamlPro

August 9, 2021

Contents

Table of Issues

To edit this document

In the report.tex file, choose:

- **\soldraftfalse** to remove draft mode (watermarks, advises)
- **\solmodulestrue** to display modules by chapter instead of contracts
- **\soltablestrue** to display tables for parameters and returns
- **\solissuesfalse** to remove the table of issues

Issues can be entered with:

- **\issueCritical{title}{text}**
- **\issueMedium{title}{text}**
- **\issueLow{title}{text}**

Chapter 1

Contract Base

Contents

In file `Base.sol`

1.1 Constant Definitions

```
8   uint16 constant ERROR_DIFFERENT_CALLER =  211;

10  uint64 constant START_BALANCE       = 3 ton;

11  uint64 constant DEPLOYER_FEE        = 0.1 ton;

12  uint64 constant PROCESS_FEE         = 0.3 ton;

13  uint64 constant VOTE_FEE            = 1 ton;

14  uint64 constant DEPLOY_FEE          = START_BALANCE +
        DEPLOYER_FEE;

15  uint64 constant DEPLOY_PAY          = DEPLOY_FEE + PROCESS_FEE;

16  uint64 constant DEPLOY_PROPOSAL_FEE = 5 ton;

17  uint64 constant DEPLOY_PROPOSAL_PAY = DEPLOY_PROPOSAL_FEE +
        PROCESS_FEE;
```

```
18      uint64 constant DEPOSIT_TONS_FEE    = 1 ton;
19      uint64 constant DEPOSIT_TONS_PAY    = DEPOSIT_TONS_FEE +
            PROCESS_FEE;
20      uint64 constant DEPOSIT_TOKENS_FEE  = 0.5 ton +
            DEPOSIT_TONS_FEE;
21      uint64 constant DEPOSIT_TOKENS_PAY  = DEPOSIT_TOKENS_FEE +
            PROCESS_FEE;
22      uint64 constant TOKEN_ACCOUNT_FEE   = 2 ton;
23      uint64 constant TOKEN_ACCOUNT_PAY   = TOKEN_ACCOUNT_FEE +
            PROCESS_FEE;
24      uint64 constant QUERY_STATUS_FEE    = 0.02 ton;
25      uint64 constant QUERY_STATUS_PAY    = QUERY_STATUS_FEE +
            DEF_RESPONSE_VALUE;
27      uint64 constant DEF_RESPONSE_VALUE = 0.03 ton;
28      uint64 constant DEF_COMPUTE_VALUE = 0.2 ton;
```

1.2 Modifier Definitions

1.2.1 Modifier signed

```
30      modifier signed {
31          require(msg.pubkey() == tvm.pubkey(), Errors.INVALID_CALLER
                );
32          tvm.accept();
33          _;
34      }
```

1.2.2 Modifier accept

```
36      modifier accept {
37          tvm.accept();
38          _;
39      }
```

1.2.3 Modifier onlyContract

```
41      modifier onlyContract() {
42          require(msg.sender != address(0), Errors.ONLY_CONTRACT);
43          _;
44      }
```

1.2.4 Modifier onlyMe

```
46      modifier onlyMe {
47          require(msg.sender == address(this), ERROR_DIFFERENT_CALLER
                );
48          _;
49      }
```

Chapter 2

Contract Demiurge

Contents

In file `Demiurge.sol`

2.1 Contract Inheritance

Base	
PadawanResolver	
ProposalResolver	
IDemiurgeStoreCb	
IFaucetCb	

2.2 Constant Definitions

```
30    uint8 constant CHECK_PROPOSAL = 1;

31    uint8 constant CHECK_PADAWAN = 2;

33    uint128 constant TOTAL_EMISSION = 21000000;
```

2.3 Variable Definitions

uint32	_deployedPadawansCounter	Initialized to 0
		used in @1.Demiurge.getStats
uint32	_deployedProposalsCounter	Initialized to 0
		used in @1.Demiurge.getStats
		assigned in @1.Demiurge._deployProposals
		used in @1.Demiurge._deployProposals
uint16	_version	Initialized to 3
		used in @1.Demiurge.getStats
address	_addrStore	
		used in @1.Demiurge.getStored
		used in @1.Demiurge.:constructor
		used in @1.Demiurge.:constructor
		used in @1.Demiurge.:constructor
		used in @1.Demiurge.:constructor
		used in @1.Demiurge.:constructor
		assigned in @1.Demiurge.:constructor
		used in @1.Demiurge.:constructor
address	_addrDensRoot	
		assigned in @1.Demiurge.updateAddr
		used in @1.Demiurge.updateAddr
		used in @1.Demiurge.getStored
		used in @1.Demiurge.deployReserveProposal
		used in @1.Demiurge._beforeProposalDeploy
address	_addrTokenRoot	
		assigned in @1.Demiurge.updateAddr
		used in @1.Demiurge.updateAddr
		used in @1.Demiurge.getStored
		used in @1.Demiurge.deployPadawan
address	_addrFaucet	
		assigned in @1.Demiurge.updateAddr
		used in @1.Demiurge.updateAddr
		used in @1.Demiurge.getStored
		used in @1.Demiurge._beforeProposalDeploy

```
    uint32 _deployedPadawansCounter = 0;

    uint32 _deployedProposalsCounter = 0;

    uint16 _version = 3;

    address _addrStore;

    address _addrDensRoot;

    address _addrTokenRoot;

    address _addrFaucet;

    uint8 _checkList;

    NewProposal[] public _newProposals;

    uint8 public _getBalancePendings = 0;

    uint128 public _totalVotes = 0;
```

2.4 Modifier Definitions

2.4.1 Modifier checksEmpty

```
    modifier checksEmpty() {
        require(_allCheckPassed(), Errors.NOT_ALL_CHECKS_PASSED);
        tvm.accept();
        _;
    }
```

2.4.2 Modifier onlyStore

```
    modifier onlyStore() {
        require(msg.sender == _addrStore);
        tvm.accept();
        _;
    }
```

2.5 Constructor Definitions

2.5.1 Constructor

Critical issue: Constructor for Demiurge (fake)

loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum

loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum

- TODO

```
82      constructor(address addrStore) public {
83          if (msg.sender == address(0)) {
84              require(msg.pubkey() == tvm.pubkey(), 101);
85          }
86          require(addrStore != address(0), Errors.
                STORE_SHOULD_BE_NOT_NULL);
87          tvm.accept();
88
89          if (addrStore != address(0)) {
90              _addrStore = addrStore;
91              DemiurgeStore(_addrStore).queryCode{value: 0.2 ton,
                    bounce: true}(ContractType.Proposal);
92              DemiurgeStore(_addrStore).queryCode{value: 0.2 ton,
                    bounce: true}(ContractType.Padawan);
93              DemiurgeStore(_addrStore).queryAddr{value: 0.2 ton,
                    bounce: true}(ContractAddr.DensRoot);
94              DemiurgeStore(_addrStore).queryAddr{value: 0.2 ton,
                    bounce: true}(ContractAddr.TokenRoot);
95              DemiurgeStore(_addrStore).queryAddr{value: 0.2 ton,
                    bounce: true}(ContractAddr.Faucet);
96          }
97
98          _createChecks();
99      }
```

2.6 Public Method Definitions

2.6.1 Function deployPadawan

- TODO

```
103     function deployPadawan(address owner) external onlyContract {
104         require(msg.value >= DEPLOY_FEE + 2 ton);
105         require(owner != address(0));
106         TvmCell state = _buildPadawanState(owner);
107         new Padawan{stateInit: state, value: START_BALANCE + 2 ton
                }(_addrTokenRoot);
108     }
```

2.6.2 Function deployReserveProposal

- TODO

```
112     function deployReserveProposal(
113         string title,
114         ReserveProposalSpecific specific
115     ) external onlyContract {
116         require(msg.value >= DEPLOY_PROPOSAL_FEE);
117         TvmBuilder b;
118         b.store(specific);
```

```
119         TvmCell cellSpecific = b.toCell();
120
121         NewProposal _newProposal = NewProposal(
122             0,
123             _addrDensRoot,
124             ProposalType.Reserve,
125             cellSpecific,
126             _codePadawan,
127             _buildProposalState(title)
128         );
129         _newProposals.push(_newProposal);
130
131         _beforeProposalDeploy(uint8(_newProposals.length - 1));
132     }
```

2.6.3 Function getStats

- TODO

```
214     function getStats() public view returns (uint16 version, uint32
            deployedPadawansCounter, uint32 deployedProposalsCounter)
        {
215         version = _version;
216         deployedPadawansCounter = _deployedPadawansCounter;
217         deployedProposalsCounter = _deployedProposalsCounter;
218     }
```

2.6.4 Function getStored

- TODO

```
198     function getStored() public view returns (
199         TvmCell codePadawan,
200         TvmCell codeProposal,
201         address addrStore,
202         address addrDensRoot,
203         address addrTokenRoot,
204         address addrFaucet
205     ) {
206         codePadawan = _codePadawan;
207         codeProposal = _codeProposal;
208         addrStore = _addrStore;
209         addrDensRoot = _addrDensRoot;
210         addrTokenRoot = _addrTokenRoot;
211         addrFaucet = _addrFaucet;
212     }
```

2.6.5 Function getTotalDistributedCb

- TODO

```
148     function getTotalDistributedCb(
149         uint128 totalDistributed
150     ) public override {
151         _totalVotes = totalDistributed;
152         _getBalancePendings -= 1;
153         _deployProposals();
154     }
```

2.6.6 Function updateAddr

- TODO

```
174     function updateAddr(ContractAddr kind, address addr) external
            override onlyStore {
175         require(addr != address(0));
176         if (kind == ContractAddr.DensRoot) {
177             _addrDensRoot = addr;
178         } else if (kind == ContractAddr.TokenRoot) {
179             _addrTokenRoot = addr;
180         } else if (kind == ContractAddr.Faucet) {
181             _addrFaucet = addr;
182         }
183     }
```

2.6.7 Function updateCode

- TODO

```
185     function updateCode(ContractType kind, TvmCell code) external
            override onlyStore {
186         tvm.accept();
187         if (kind == ContractType.Proposal) {
188             _codeProposal = code;
189             _passCheck(CHECK_PROPOSAL);
190         } else if (kind == ContractType.Padawan) {
191             _codePadawan = code;
192             _passCheck(CHECK_PADAWAN);
193         }
194     }
```

2.7 Internal Method Definitions

2.7.1 Function _allCheckPassed

- TODO

```
62     function _allCheckPassed() private view inline returns (bool) {
63         return (_checkList == 0);
64     }
```

2.7.2 Function _beforeProposalDeploy

- TODO

```
134     function _beforeProposalDeploy(
135         uint8 i
136     ) private {
137         uint256 hashState = tvm.hash(_newProposals[i].state);
138         address addrProposal = address.makeAddrStd(0, hashState);
139         IClient(_addrDensRoot).onProposalDeploy
140             {value: 1 ton, bounce: true}
141             (addrProposal, _newProposals[i].proposalType,
                    _newProposals[i].specific);
142 
143         IFaucet(_addrFaucet).getTotalDistributed
144             {value: 0.2 ton, flag: 1, bounce: false}();
145         _getBalancePendings += 1;
146     }
```

2.7.3 Function _createChecks

- TODO

```
54     function _createChecks() private inline {
55         _checkList = CHECK_PADAWAN | CHECK_PROPOSAL;
56     }
```

2.7.4 Function _deployProposals

- TODO

```
156     function _deployProposals() private {
157         if(_getBalancePendings == 0) {
158             for(uint8 i = 0; i < _newProposals.length; i++) {
159                 new Proposal {stateInit: _newProposals[i].state,
                        value: START_BALANCE}(
160                     _totalVotes,
161                     _newProposals[i].addrClient,
162                     _newProposals[i].proposalType,
163                     _newProposals[i].specific,
164                     _newProposals[i].codePadawan
165                 );
166                 _deployedProposalsCounter++;
167             }
168             delete _newProposals;
169         }
170     }
```

2.7.5 Function _passCheck

- TODO

```
58      function _passCheck(uint8 check) private inline {
59          _checkList &= ~check;
60      }
```

Chapter 3

Contract DemiurgeStore

Contents

In file `DemiurgeStore.sol`

3.1 Contract Inheritance

Base	

3.2 Variable Definitions

mapping (uint8 => address)	_addrs	
		assigned in @4.DemiurgeStore.setTokenRootAddr
		used in @4.DemiurgeStore.setTokenRootAddr
		assigned in @4.DemiurgeStore.setFaucetAddr
		used in @4.DemiurgeStore.setFaucetAddr
		assigned in @4.DemiurgeStore.setDensRootAddr
		used in @4.DemiurgeStore.setDensRootAddr
		used in @4.DemiurgeStore.queryAddr
mapping (uint8 => TvmCell)	_codes	
		assigned in @4.DemiurgeStore.setProposalCode
		used in @4.DemiurgeStore.setProposalCode
		assigned in @4.DemiurgeStore.setPadawanCode
		used in @4.DemiurgeStore.setPadawanCode
		used in @4.DemiurgeStore.queryCode

```
11      mapping(uint8 => address) public _addrs;

12      mapping(uint8 => TvmCell) public _codes;
```

3.3 Public Method Definitions

3.3.1 Function queryAddr

- TODO

```
43      function queryAddr(ContractAddr kind) public view {
44          address addr = _addrs[uint8(kind)];
45          IDemiurgeStoreCb(msg.sender).updateAddr{value: 0, flag: 64,
                bounce: false}(kind, addr);
46      }
```

3.3.2 Function queryCode

- TODO

```
38      function queryCode(ContractType kind) public view {
39          TvmCell code = _codes[uint8(kind)];
40          IDemiurgeStoreCb(msg.sender).updateCode{value: 0, flag: 64,
                bounce: false}(kind, code);
41      }
```

3.3.3 Function setDensRootAddr

- TODO

```
21      function setDensRootAddr(address addr) public signed {
22          require(addr != address(0));
23          _addrs[uint8(ContractAddr.DensRoot)] = addr;
24      }
```

3.3.4 Function setFaucetAddr

- TODO

```
29      function setFaucetAddr(address addr) public signed {
30          require(addr != address(0));
31          _addrs[uint8(ContractAddr.Faucet)] = addr;
32      }
```

3.3.5 Function setPadawanCode

- TODO

```
14      function setPadawanCode(TvmCell code) public signed {
15          _codes[uint8(ContractType.Padawan)] = code;
16      }
```

3.3.6 Function setProposalCode

- TODO

```
17      function setProposalCode(TvmCell code) public signed {
18          _codes[uint8(ContractType.Proposal)] = code;
19      }
```

3.3.7 Function setTokenRootAddr

- TODO

```
25      function setTokenRootAddr(address addr) public signed {
26          require(addr != address(0));
27          _addrs[uint8(ContractAddr.TokenRoot)] = addr;
28      }
```

Chapter 4

Contract Padawan

Contents

In file `Padawan.sol`

4.1 Contract Inheritance

Base	

4.2 Static Variable Definitions

address	_deployer	
		used in @2.Padawan.:constructor
address	_owner	
		used in @2.Padawan.vote
		used in @2.Padawan.rejectVote
		used in @2.Padawan.onTokenWalletDeploy
		used in @2.Padawan.onGetBalance
		used in @2.Padawan.getAddresses
		used in @2.Padawan.confirmVote

```
18     address static _deployer;

19     address static _owner;
```

4.3 Variable Definitions

address	_addrTokenRoot	
		used in @2.Padawan._createTokenAccount
		assigned in @2.Padawan.:constructor
		used in @2.Padawan.:constructor
TipAccount	_tipAccount	
		assigned in @2.Padawan.onTokenWalletDeploy
		used in @2.Padawan.onTokenWalletDeploy
		assigned in @2.Padawan.onGetBalance
		used in @2.Padawan.onGetBalance
		used in @2.Padawan.onGetBalance
		used in @2.Padawan.getTipAccount
		used in @2.Padawan.getAll
		used in @2.Padawan.depositTokens
		used in @2.Padawan.depositTokens
		used in @2.Padawan._unlockDeposit
address	_returnTo	
		assigned in @2.Padawan.reclaimDeposit
		used in @2.Padawan.reclaimDeposit
		assigned in @2.Padawan._unlockDeposit
		used in @2.Padawan._unlockDeposit
		used in @2.Padawan._unlockDeposit
mapping (address => uint32)	_activeProposals	
		assigned in @2.Padawan.vote
		used in @2.Padawan.vote
		used in @2.Padawan.vote
		assigned in @2.Padawan.updateStatus
		used in @2.Padawan.updateStatus
		used in @2.Padawan.updateStatus
		assigned in @2.Padawan.rejectVote
		used in @2.Padawan.rejectVote
		used in @2.Padawan.rejectVote
		used in

```
    address _addrTokenRoot;

    TipAccount _tipAccount;

    address _returnTo;

    mapping(address => uint32) _activeProposals;

    uint32 _requestedVotes;

    uint32 _totalVotes;

    uint32 _lockedVotes;
```

4.4 Modifier Definitions

4.4.1 Modifier onlyOwner

```
    modifier onlyOwner() {
        require(msg.sender == _owner, Errors.
            NOT_AUTHORIZED_CONTRACT);
        _;
    }
```

4.4.2 Modifier onlyTokenRoot

```
    modifier onlyTokenRoot() {
        require(msg.sender == _addrTokenRoot, Errors.INVALID_CALLER
            );
        _;
    }
```

4.5 Constructor Definitions

4.5.1 Constructor

Critical issue: Constructor for Padawan (fake)

loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum

loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum

- TODO

```
    constructor(address addrTokenRoot) public onlyContract {
        require(_deployer == msg.sender, Errors.ONLY_DEPLOYER);
        _addrTokenRoot = addrTokenRoot;
        _createTokenAccount();
    }
```

4.6 Public Method Definitions

4.6.1 Function confirmVote

- TODO

```
74      function confirmVote(uint32 votesCount) external onlyContract {
75          // TODO: better to check is it proposal or not
76          optional(uint32) optActiveProposal = _activeProposals.fetch
                (msg.sender);
77          require(optActiveProposal.hasValue());
78
79          _activeProposals[msg.sender] += votesCount;
80
81          _updateLockedVotes();
82
83          _owner.transfer(0, false, 64);
84      }
```

4.6.2 Function depositTokens

- TODO

```
172     function depositTokens() external onlyOwner view {
173         require(msg.value >= DEPOSIT_TOKENS_FEE, Errors.
                MSG_VALUE_TOO_LOW);
174         require(_tipAccount.addr != address(0), Errors.
                ACCOUNT_DOES_NOT_EXIST);
175
176         ITokenWallet(_tipAccount.addr).getBalance_InternalOwner
177             {value: 0, flag: 64, bounce: true}
178             (tvm.functionId(onGetBalance));
179     }
```

4.6.3 Function getActiveProposals

- TODO

```
228     function getActiveProposals() public view returns (mapping(
            address => uint32) activeProposals) {
229         activeProposals = _activeProposals;
230     }
```

4.6.4 Function getAddresses

- TODO

```
224     function getAddresses() public view returns (address
            ownerAddress) {
225         ownerAddress = _owner;
226     }
```

4.6.5 Function getAll

- TODO

```
207     function getAll() external view returns (TipAccount tipAccount,
            uint32 reqVotes, uint32 totalVotes, uint32 lockedVotes) {
208         tipAccount = _tipAccount;
209         reqVotes = _requestedVotes;
210         totalVotes =  _totalVotes;
211         lockedVotes = _lockedVotes;
212     }
```

4.6.6 Function getTipAccount

- TODO

```
214     function getTipAccount() external view returns (TipAccount
            tipAccount) {
215         tipAccount = _tipAccount;
216     }
```

4.6.7 Function getVoteInfo

- TODO

```
218     function getVoteInfo() external view returns (uint32 reqVotes,
            uint32 totalVotes, uint32 lockedVotes) {
219         reqVotes = _requestedVotes;
220         totalVotes =  _totalVotes;
221         lockedVotes = _lockedVotes;
222     }
```

4.6.8 Function onGetBalance

- TODO

```
181     function onGetBalance(uint128 balance) public onlyContract {
182         require(_tipAccount.addr == msg.sender, Errors.
                NOT_AUTHORIZED_CONTRACT);
183         _tipAccount.balance = balance;
184         _totalVotes = uint32(balance);
185         _owner.transfer(0, false, 64);
186     }
```

4.6.9 Function onTokenWalletDeploy

- TODO

```
192     function onTokenWalletDeploy(address ownerAddress) public
            onlyTokenRoot {
193         _tipAccount = TipAccount(ownerAddress, 0);
194         _owner.transfer(0, false, 64);
195     }
```

4.6.10 Function reclaimDeposit

- TODO

```
103     function reclaimDeposit(uint32 votes, address returnTo)
            external onlyOwner {
104         require(msg.value >= 3 ton, Errors.MSG_VALUE_TOO_LOW);
105         require(votes <= _totalVotes, Errors.NOT_ENOUGH_VOTES);
106         require(returnTo != address(0));
107         _returnTo = returnTo;
108         _requestedVotes = votes;
109
110         if (_requestedVotes <= _totalVotes - _lockedVotes) {
111             _unlockDeposit();
112         } else {
113             _requestedVotes = 0;
114         }
115
116         optional(address, uint32) optActiveProposal =
                _activeProposals.min();
117         while (optActiveProposal.hasValue()) {
118             (address addrActiveProposal,) = optActiveProposal.get()
                    ;
119             IProposal(addrActiveProposal).queryStatus
120                 {value: QUERY_STATUS_FEE, bounce: true, flag: 1}
121                 ();
122             optActiveProposal = _activeProposals.next(
                        addrActiveProposal);
123         }
124     }
```

4.6.11 Function rejectVote

- TODO

```
87      function rejectVote(uint32 votesCount, uint16 errorCode)
            external onlyContract {
88          votesCount; errorCode;
89
90          // TODO: better to check is it proposal or not
91          optional(uint32) optActiveProposal = _activeProposals.fetch
                (msg.sender);
92          require(optActiveProposal.hasValue());
```

```
93          uint32 activeProposalVotes = optActiveProposal.get();
94          if (activeProposalVotes == 0) {
95              delete _activeProposals[msg.sender];
96          }
97
98          _owner.transfer(0, false, 64);
99      }
```

4.6.12 Function updateStatus

- TODO

```
127     function updateStatus(ProposalState state) external
            onlyContract {
128         optional(uint32) optActiveProposal = _activeProposals.fetch
                (msg.sender);
129         require(optActiveProposal.hasValue());
130         tvm.accept();
131
132         if (state >= ProposalState.Ended) {
133             delete _activeProposals[msg.sender];
134             _updateLockedVotes();
135         }
136
137         if (_requestedVotes != 0 && _requestedVotes <= _totalVotes
                - _lockedVotes) {
138             _unlockDeposit();
139         }
140     }
```

4.6.13 Function vote

- TODO

```
55      function vote(address proposal, bool choice, uint32 votes)
            external onlyOwner {
56          require(msg.value >= VOTE_FEE, Errors.MSG_VALUE_TOO_LOW);
57          optional(uint32) optActiveProposal = _activeProposals.fetch
                (proposal);
58
59          uint32 activeProposalVotes = optActiveProposal.hasValue() ?
                 optActiveProposal.get() : 0;
60          uint32 availableVotes = _totalVotes - activeProposalVotes;
61          require(votes <= availableVotes, Errors.NOT_ENOUGH_VOTES);
62
63          // TODO: better to remove
64          if (activeProposalVotes == 0) {
65              _activeProposals[proposal] = 0;
66          }
67
68          IProposal(proposal).vote
69              {value: 0, flag: 64, bounce: true}
70              (_owner, choice, votes);
71      }
```

4.7 Internal Method Definitions

4.7.1 Function _createTokenAccount

- TODO

```
197     function _createTokenAccount() private view {
198         ITokenRoot(_addrTokenRoot).deployEmptyWallet
199             {value: 2 ton, flag: 1, bounce: true}
200             (tvm.functionId(onTokenWalletDeploy), 0, 0, address(
                    this).value, 1 ton);
201     }
```

4.7.2 Function _unlockDeposit

- TODO

```
146     function _unlockDeposit() private {
147         ITokenWallet(_tipAccount.addr).transfer
148             {value: 0.1 ton + 0.1 ton}
149             (_returnTo, _requestedVotes, 0.1 ton);
150         _totalVotes -= _requestedVotes;
151         _requestedVotes = 0;
152         _returnTo = address(0);
153     }
```

4.7.3 Function _updateLockedVotes

- TODO

```
155     function _updateLockedVotes() private inline {
156         optional(address, uint32) optActiveProposal =
                _activeProposals.min();
157         uint32 lockedVotes;
158         while (optActiveProposal.hasValue()) {
159             (address addr, uint32 votes) = optActiveProposal.get();
160             if (votes > lockedVotes) {
161                 lockedVotes = votes;
162             }
163             optActiveProposal = _activeProposals.next(addr);
164         }
165         _lockedVotes = lockedVotes;
166     }
```

Chapter 5

Contract PadawanResolver

Contents

In file `PadawanResolver.sol`

5.1 Variable Definitions

TvmCell	_codePadawan	
		assigned in @1.Demiurge.updateCode
		used in @1.Demiurge.updateCode
		used in @1.Demiurge.getStored
		used in @1.Demiurge.deployReserveProposal
		used in @3.Proposal._buildPadawanState
		assigned in @3.Proposal.:constructor
		used in @3.Proposal.:constructor
		used in @11.PadawanResolver._buildPadawanState

```
8     TvmCell _codePadawan;
```

5.2 Public Method Definitions

5.2.1 Function resolvePadawan

- TODO

```
10      function resolvePadawan(address owner) public view returns (
            address addrPadawan) {
11          TvmCell state = _buildPadawanState(owner);
12          uint256 hashState = tvm.hash(state);
13          addrPadawan = address.makeAddrStd(0, hashState);
14      }
```

5.3 Internal Method Definitions

5.3.1 Function _buildPadawanState

- TODO

```
16      function _buildPadawanState(address owner) internal virtual
            view returns (TvmCell) {
17          return tvm.buildStateInit({
18              contr: Padawan,
19              varInit: {_deployer: address(this), _owner: owner},
20              code: _codePadawan
21          });
22      }
```

Chapter 6

Contract Proposal

Contents

In file `Proposal.sol`

6.1 Contract Inheritance

Base	
PadawanResolver	
IProposal	

6.2 Event Definitions

```
23      event ProposalFinalized(ProposalResults results);
```

6.3 Static Variable Definitions

address	_deployer	
		used in @3.Proposal._buildPadawanState
		used in @3.Proposal.:constructor
string	_title	
		used in @3.Proposal.:constructor

```
13      address static _deployer;
```

```
14      string static _title;
```

6.4 Variable Definitions

address	_addrClient	
		used in @3.Proposal._finalize
		assigned in @3.Proposal.:constructor
		used in @3.Proposal.:constructor
ProposalInfo	_proposalInfo	
		assigned in @3.Proposal.vote
		used in @3.Proposal.vote
		assigned in @3.Proposal.vote
		used in @3.Proposal.vote
		used in @3.Proposal.vote
		used in @3.Proposal.vote
		used in @3.Proposal.queryStatus
		used in @3.Proposal.getVotingResults
		used in @3.Proposal.getInfo
		used in @3.Proposal.getCurrentVotes
		used in @3.Proposal.getCurrentVotes
		used in @3.Proposal.getAll
		used in @3.Proposal._wrapUp
		used in @3.Proposal._wrapUp
		used in @3.Proposal._wrapUp
		used in @3.Proposal._wrapUp
		used in @3.Proposal._wrapUp
		used in @3.Proposal._tryEarlyComplete
		used in @3.Proposal._tryEarlyComplete
		used in @3.Proposal._softMajority
		used in @3.Proposal._softMajority
		used in @3.Proposal._softMajority
		used in @3.Proposal._softMajority
		used in @3.Proposal._finalize
		used in @3.Proposal._finalize
		used in @3.Proposal._finalize
		used in @3.Proposal._finalize
		assigned in @3.Proposal._changeState
		used in @3.Proposal._changeState
		assigned in @3.Proposal.:constructor
		used in @3.Proposal.:constructor
		assigned in @3.Proposal.:constructor
		used in @3.Proposal.:constructor

```
    address public _addrClient;
```

```
    ProposalInfo public _proposalInfo;
```

```
    ProposalResults _results;
```

```
    VoteCountModel _voteCountModel;
```

6.5 Constructor Definitions

6.5.1 Constructor

Critical issue: Constructor for Proposal (fake)

loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum
loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum loren ipsum

- TODO

```
    constructor(
        uint128 totalVotes,
        address addrClient,
        ProposalType proposalType,
        TvmCell specific,
        TvmCell codePadawan
    ) public {
        require(_deployer == msg.sender);

        _addrClient = addrClient;

        _proposalInfo.title = _title;
        _proposalInfo.start = uint32(now);
        _proposalInfo.end = uint32(now + 60 * 60 * 24 * 7);
        _proposalInfo.proposalType = proposalType;
        _proposalInfo.specific = specific;
        _proposalInfo.state = ProposalState.New;
        _proposalInfo.totalVotes = totalVotes;

        _codePadawan = codePadawan;

        _voteCountModel = VoteCountModel.SoftMajority;
    }
```

6.6 Public Method Definitions

6.6.1 Function getAll

- TODO

```
168     function getAll() public view override returns (ProposalInfo
            info) {
169         info = _proposalInfo;
170     }
```

6.6.2 Function getCurrentVotes

- TODO

```
181     function getCurrentVotes() external override view returns (
            uint32 votesFor, uint32 votesAgainst) {
182         return (_proposalInfo.votesFor, _proposalInfo.votesAgainst)
                ;
183     }
```

6.6.3 Function getInfo

- TODO

```
177     function getInfo() public view returns (ProposalInfo info) {
178         info = _proposalInfo;
179     }
```

6.6.4 Function getVotingResults

- TODO

```
172     function getVotingResults() public view returns (
            ProposalResults vr) {
173         require(_proposalInfo.state > ProposalState.Ended, Errors.
                VOTING_HAS_NOT_ENDED);
174         vr = _results;
175     }
```

6.6.5 Function queryStatus

- TODO

```
162     function queryStatus() external override {
163         IPadawan(msg.sender).updateStatus(_proposalInfo.state);
164     }
```

6.6.6 Function vote

- TODO

```
55      function vote(address addrPadawanOwner, bool choice, uint32
            votesCount) external override {
56          address addrPadawan = resolvePadawan(addrPadawanOwner);
57          uint16 errorCode = 0;
58
59          if (addrPadawan != msg.sender) {
60              errorCode = Errors.NOT_AUTHORIZED_CONTRACT;
61          } else if (now < _proposalInfo.start) {
62              errorCode = Errors.VOTING_NOT_STARTED;
63          } else if (now > _proposalInfo.end) {
64              errorCode = Errors.VOTING_HAS_ENDED;
65          }
66
67          if (errorCode > 0) {
68              IPadawan(msg.sender).rejectVote{value: 0, flag: 64,
                    bounce: true}(votesCount, errorCode);
69          } else {
70              IPadawan(msg.sender).confirmVote{value: 0, flag: 64,
                    bounce: true}(votesCount);
71              if (choice) {
72                  _proposalInfo.votesFor += votesCount;
73              } else {
74                  _proposalInfo.votesAgainst += votesCount;
75              }
76          }
77
78          _wrapUp();
79      }
```

6.6.7 Function wrapUp

- TODO

```
49      function wrapUp() external override {
50          _wrapUp();
51          msg.sender.transfer(0, false, 64);
52      }
```

6.7 Internal Method Definitions

6.7.1 Function _buildPadawanState

- TODO

```
154     function _buildPadawanState(address owner) internal view
            override returns (TvmCell) {
155         return tvm.buildStateInit({
156             contr: Padawan,
```

```
157             varInit: {_deployer: _deployer, _owner: owner},
158             code: _codePadawan
159         });
160     }
```

6.7.2 Function _calculateVotes

- TODO

```
132     function _calculateVotes(
133         uint32 yes,
134         uint32 no
135     ) private view returns (bool) {
136         bool passed = false;
137         passed = _softMajority(yes, no);
138         return passed;
139     }
```

6.7.3 Function _changeState

- TODO

```
150     function _changeState(ProposalState state) private inline {
151         _proposalInfo.state = state;
152     }
```

6.7.4 Function _finalize

- TODO

```
81      function _finalize(bool passed) private {
82          _results = ProposalResults(
83              uint32(0),
84              passed,
85              _proposalInfo.votesFor,
86              _proposalInfo.votesAgainst,
87              _proposalInfo.totalVotes,
88              _voteCountModel,
89              uint32(now)
90          );
91
92          ProposalState state = passed ? ProposalState.Passed :
                ProposalState.NotPassed;
93
94          _changeState(state);
95
96          IClient(address(_addrClient)).onProposalPassed{value: 1 ton
                } (_proposalInfo);
97
98          emit ProposalFinalized(_results);
99      }
```

6.7.5 Function _softMajority

- TODO

```
141     function _softMajority(
142         uint32 yes,
143         uint32 no
144     ) private view returns (bool) {
145         bool passed = false;
146         passed = yes >= 1 + (_proposalInfo.totalVotes / 10) + (no *
                ((_proposalInfo.totalVotes / 2) - (_proposalInfo.
                totalVotes / 10))) / (_proposalInfo.totalVotes / 2);
147         return passed;
148     }
```

6.7.6 Function _tryEarlyComplete

- TODO

```
101     function _tryEarlyComplete(
102         uint32 yes,
103         uint32 no
104     ) private view returns (bool, bool) {
105         (bool completed, bool passed) = (false, false);
106         if (yes * 2 > _proposalInfo.totalVotes) {
107             completed = true;
108             passed = true;
109         } else if(no * 2 >= _proposalInfo.totalVotes) {
110             completed = true;
111             passed = false;
112         }
113         return (completed, passed);
114     }
```

6.7.7 Function _wrapUp

- TODO

```
116     function _wrapUp() private {
117         (bool completed, bool passed) = (false, false);
118 
119         if (now > _proposalInfo.end) {
120             completed = true;
121             passed = _calculateVotes(_proposalInfo.votesFor,
                    _proposalInfo.votesAgainst);
122         } else {
123             (completed, passed) = _tryEarlyComplete(_proposalInfo.
                    votesFor, _proposalInfo.votesAgainst);
124         }
125 
126         if (completed) {
127             _changeState(ProposalState.Ended);
128             _finalize(passed);
129         }
130     }
```

Chapter 7

Contract ProposalResolver

Contents

In file `ProposalResolver.sol`

7.1 Variable Definitions

TvmCell	_codeProposal	
		assigned in @1.Demiurge.updateCode
		used in @1.Demiurge.updateCode
		used in @1.Demiurge.getStored
		used in @12.ProposalResolver._buildProposalState

```
6     TvmCell _codeProposal;
```

7.2 Public Method Definitions

7.2.1 Function resolveProposal

- TODO

```
8       function resolveProposal(string title) public view returns (
            address addrProposal) {
9           TvmCell state = _buildProposalState(title);
10          uint256 hashState = tvm.hash(state);
11          addrProposal = address.makeAddrStd(0, hashState);
12      }
```

7.3 Internal Method Definitions

7.3.1 Function _buildProposalState

- TODO

```
14      function _buildProposalState(string title) internal view
            returns (TvmCell) {
15          return tvm.buildStateInit({
16              contr: Proposal,
17              varInit: {_deployer: address(this), _title: title},
18              code: _codeProposal
19          });
20      }
```