CONSENSYS
Diligence

AragonOne — Aragon Network Presale Audit

Date	November 2019
Lead Auditor	Martin Ortner
Co-auditors	Sergii Kravchenko

1 Summary

ConsenSys Diligence conducted a security audit on a fork of the Aragon Fundraising Presale contract and template.

Note: This audit is an extension of the AragonBlack/Fundraising audit we recently performed. Please refer to the AragonBlack/Fundraising audit for a general overview of the system and its components, actors, key observations, security specification, and trust model.

2 Audit Scope

This two-day audit covered the following files from the source repository at GitHub: AragonOne/fundraising@ `f515e43a` :

File	
templates/externally_owned_presale_bonding_curve/contracts/EOPBCTemplate.sol	b3
apps/presale/contracts/IPresale.sol	76
apps/presale/contracts/BalanceRedirectPresale.sol	83

Files not listed are explicitly out of scope for this audit.

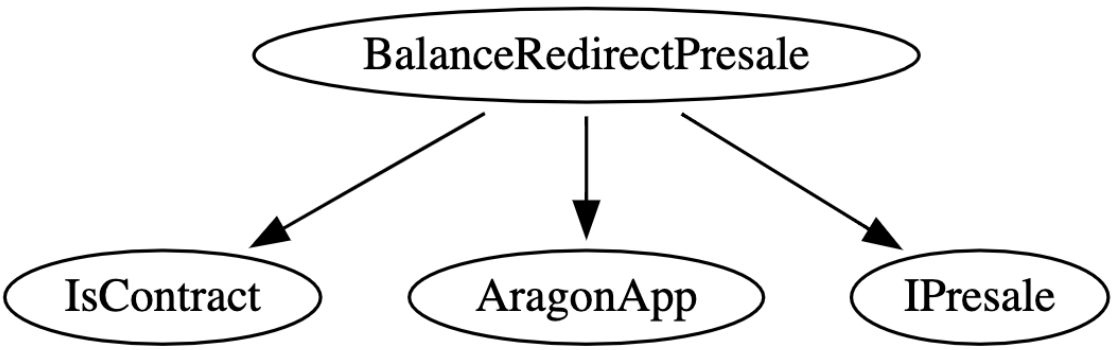
The audit activities can be grouped into the following three broad categories:

1. **Security:** Identifying security related issues within the contract.
2. **Architecture:** Evaluating the system architecture through the lens of established smart contract best practices.
3. **Code quality:** A full review of the contract source code. The primary areas of focus include:
    - Correctness
    - Readability
    - Scalability
    - Code complexity
    - Quality of test coverage

3 System Overview

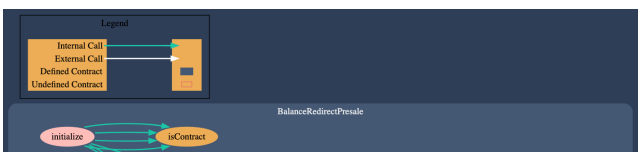
The system under audit for this review is a fork of the original AragonBlack/Fundraising `Presale` contract named `BalanceRedirectPresale`. This work is extending to a previous audit. Please refer to the AragonBlack/Fundraising audit for general information and details.

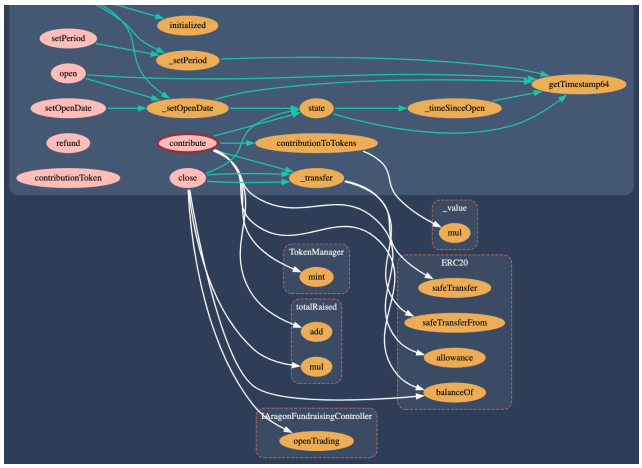
Inheritance Structure



Inheritance graph of the BalanceRedirectPresalePresale contract

Call Graph

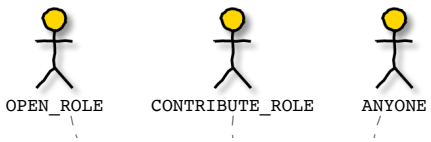


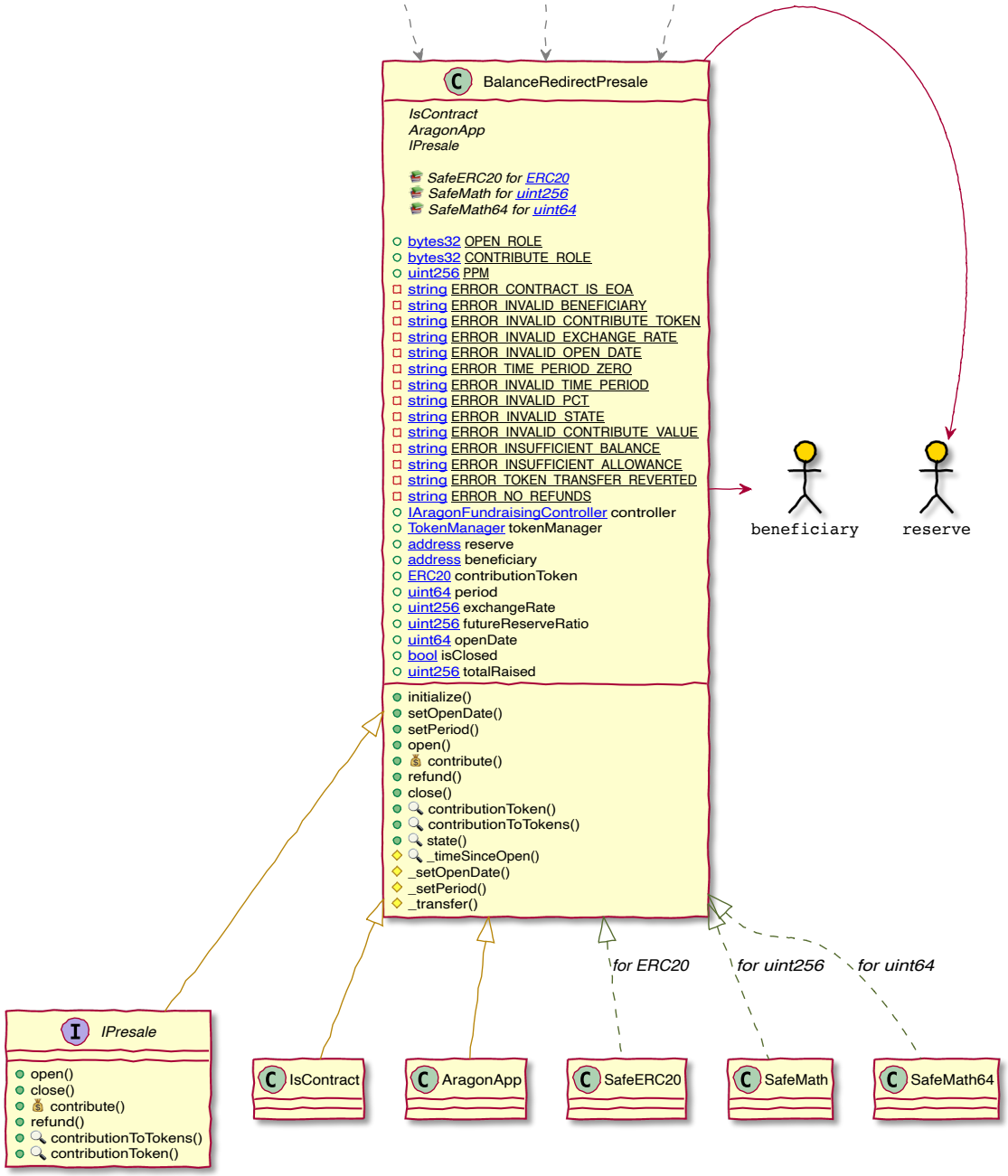


Call graph of the BalanceRedirectPresalePresale Contract

3.1 BalanceRedirectPresale

This contract is a simplified version of the `Presale` contract found at Github: AragonBlack/fundraising. It allows the `OPEN_ROLE` to configure and start a time-limited token presale for a Fundraising Campaign. Token are sold at a fixed rate and they vest immediately. The presale can only be started once and always succeeds as the presale does not define a minimum number of tokens to be reached. Contributors must have the `CONTRIBUTOR_ROLE` which is usually assigned to the `ANY_ENTITY`. Contributions cannot be refund.





Contract outline of the BalanceRedirectPresalePresale contract

3.2 EOPBCTemplate

The fundraising DAO template is based on the Company-Board default DAO-Template. This default DAO-Template has been audited as part of the Aragon DAO-Templates Audit. Please refer to the references report for general security information. A security analysis of the template is provided as part of section 5 - Security Specification.

The template is a simplification of the AragonBlack/FundraisingMultisigTemplate. Wherever possible it makes use of functionality provided by the Aragon `BaseTemplate` which is part of the Aragon default DAO-Templates repository

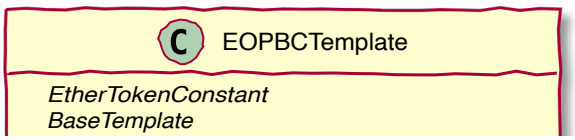
`BaseTemplate` which is part of the Aragon default DAO Templates repository.
New fundraising enabled DAOs can be deployed in one step:

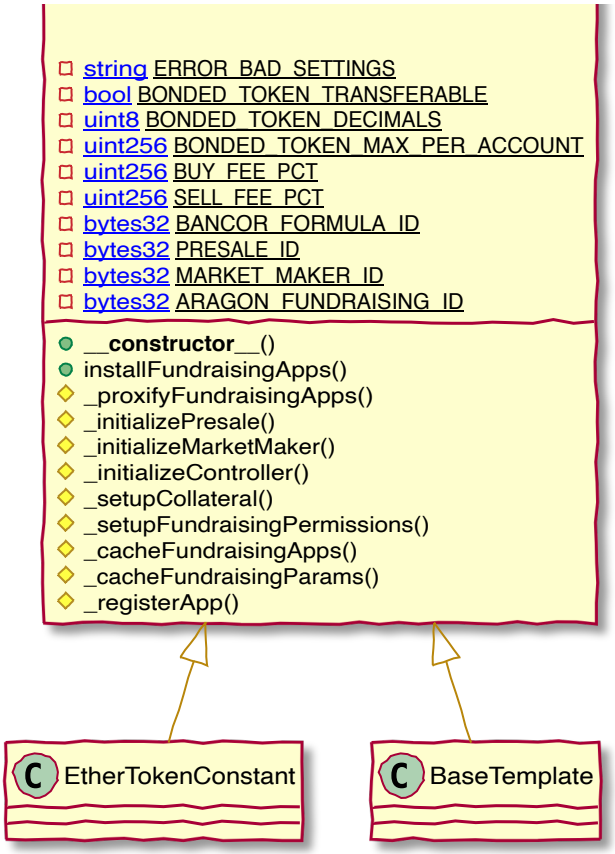
1. `installFundraisingApps`
    - creates a new DAO (Kernel, ACL) and assigns permissions to the template
    - installs fundraising applications (Agent/Reserve, Presale, MarketMaker, TapDisabled, Controller, TokenManager_BOND)
    - sets up the fundraising params in a struct
    - initializes Presale
    - initializes MarketMaker
    - initializes Controller
    - sets up fundraising permissions (TokenManager, Presale, Agent, MarketMaker, Controller)
    - sets up collateral
    - temporarily assigns `ADD_COLLATERAL_TOKEN_ROLE` to the template
    - transfers `ADD_COLLATERAL_TOKEN_ROLE` to `owner`
    - transfers root permissions (Kernel, ACL) from template to `owner`
    - registers application id

- **Note:** the template does not create Evm Scripts Registry Permissions.
- **Note:** the template takes an external `bondedToken`. The `MiniMeTokenFactory` is never used to create new tokens.

The `aragonID`, `DAOFactory`, `ENS Registry`, and `MiniMeTokenFactory` contract addresses are specified when deploying the template contract. Users should make sure the initial configuration of the template is safe (no malicious factory or collateral token contracts) when using a 3rd party template contract to deploy a new DAO.

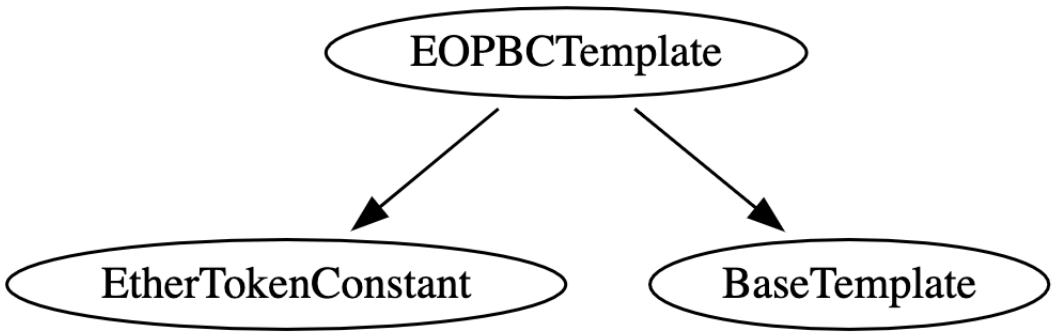
A visual representation of the permission setup deployed with the DAO can be found in section 5 - Security Specification.





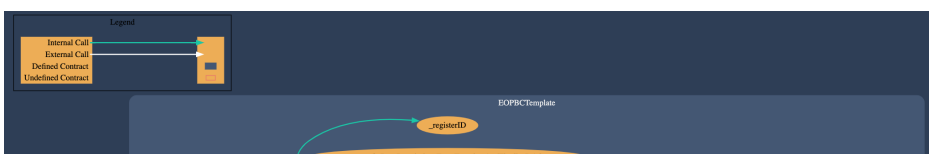
Contract outline of the EOPBCTemplate contract

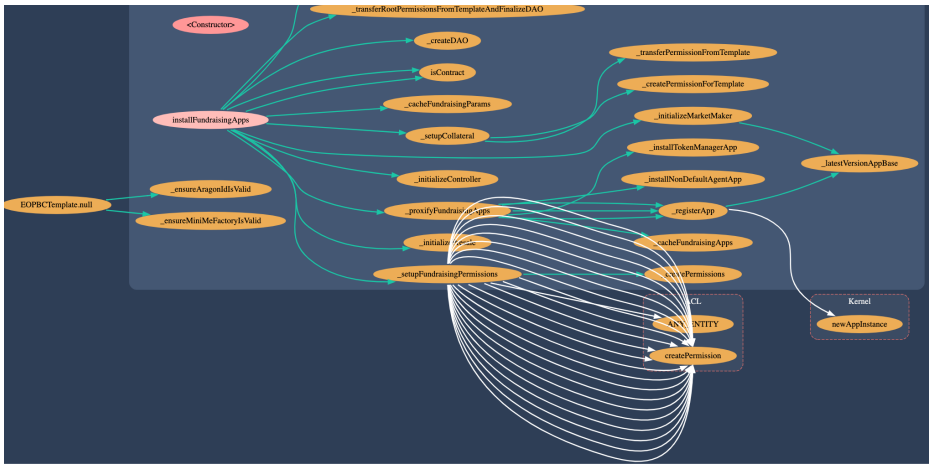
Inheritance Structure



Inheritance graph

Call Graph





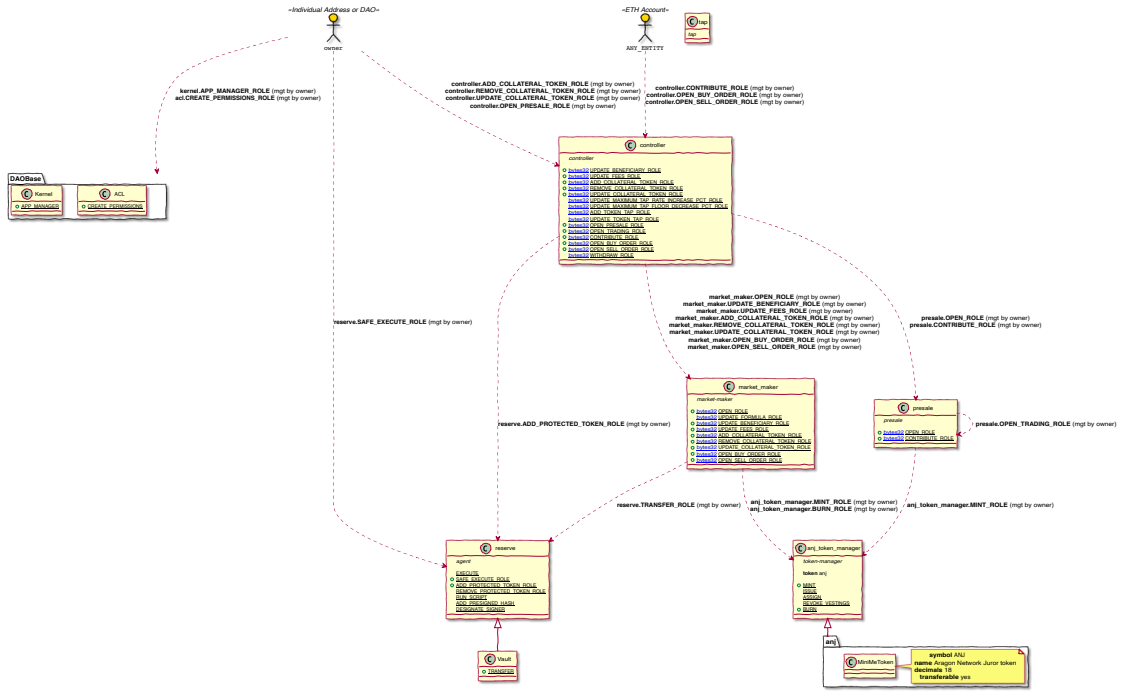
Call graph

4 Key Observations/Recommendations

- The project team provided system documentation and auditable specification documents. It is typically suggested to make documentation available that clearly outlines at least the following information:
    - Application Name, Version and Outline
    - Roles & Capabilities ideally grouped into logical actors (e.g. Investors, Project Managers, …)
    - Set-Up and initialization details
    - Caveats & Limitations
    - Security Considerations, Common Pitfalls or Secure Setup information
    - A description of an example Application Lifecycle
    - A reference to an example kit to deploy the Aragon Black Fundraising application with a DAO
- The project is a fork of the AragonBlack/Fundraising application. It is less complex, removes vesting, does not require a goal to be hit, does not all contributions to be refunded, provides more freedom when setting opening date and period and does not allow `ETH` contributions.
- The project does not make use of the `Tap` functionality. The tap amount is not restricted. Permissions to withdraw collateral from the agent is not assigned.
- The project provides a presale interface `IPresale` that can be used for both presale variants.

5 Security Specification

This section describes the behavior of the system under audit from a security perspective. It is best combined with the overview given in section 3 - System Overview. Please note that this document is not a substitute for documentation. The purpose of this section is to identify specific security properties that were validated by the audit team. Furthermore, the information contained in this section can be used for internal security activities and we recommend documenting and building-upon the trust model that has been established.



Template Permission Overview

5.1 Actors

The relevant actors are as follows:

- **EOPBCTemplate**
- **The DAO deployer**
- **Owner**
- **Beneficiary**
- **Shareholders**

- **Any Ethereum Account**

5.2 Trust Model

The trust model and security observations aim to bring transparency about security-relevant characteristics of the system, help to understand trust assumptions and describe potential high-level threats to the system. The goal is to spark security discussions, document them as part of a continuous process and use them as input for internal SDL security practices.

It is based on the permission setup provided with the `EOPBCTemplate`. The audit team would like to note that the system can be deployed with various configurations. Other templates (DAO scenarios) than the one audited as part of this work might not enforce secure defaults or a safe permission setup.

Deployment

Before the Fundraising DAO can be used it has to be deployed using a template contract. This template contract can be provided by Aragon or third parties. We would like to emphasize that both the template contract code and its initial configuration as well as its dependencies (especially Factory Contracts) must be verified and should be audited. The template contract nor factory contracts or a third party should remain in control of any of the newly deployed DAOs components.

- The **EOPBCTemplate** can be deployed by anyone. The template deployer does not remain in direct control of the template but it can be indirectly controlled via the templates default configuration and factories being used (e.g. DAOFactory, MiniMeFactory, TokenContracts).
    - The MiniMeFactory is unused by this template. The bonding token is external, passed as an argument to the deployment routine.
    - No tokens are minted by the template.
- The **DAO deployer** is an account that interacts with the **EOPBCTemplate** to deploy a new DAO. It is initiating the four DAO deployment steps outlined in section 3 - System Overview.
- In the course of the deployment of a DAO, permissions are assigned to the **EOPBCTemplate**. For example, `_createDAO` initially assigns

  `Kernell.APP_MANAGER_ROLE` and `Acl.CREATE_PERMISSIONS_ROLE` to the template. The **EOPBCTemplate** temporarily assigns `Controller.ADD_COLLATERAL_TOKEN_ROLE` to itself and transfers this

`CONTROLLER.ADD_COLLATERAL_TOKEN_ROLE` to itself and transfers this permission to **Owner** after whitelisting the specified collateral. When finalizing the new DAO the **EOPBCTemplate** transfers `Kernell.APP_MANAGER_ROLE` and `Acl.CREATE_PERMISSIONS_ROLE` to Voting_Share effectively revoking its access from the newly deployed DAO.

- The **DAO deployer** is not granted any permissions during the deployment of the new DAO but it is in control of configuration options (e.g. bonded token). DAO users must verify the configuration settings of the newly deployed DAO before participating in it.

- The external bonded token is not controlled by the DAO. It is unclear if someone already minted and assigned tokens before deploying the DAO. Stakes and voting power that might be derived from the bonded tokens can be tampered with before the DAO is deployed.

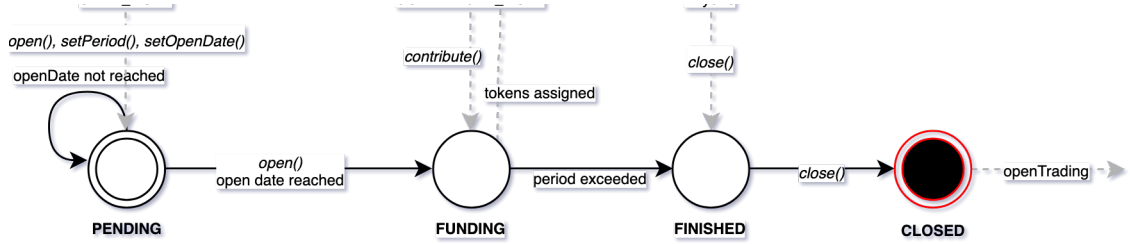
- The **Owner** is initially set as the **Beneficiary** for fees, presale tokens and tap withdrawal by the **FundraisingMulEOPBCTemplatetisigTemplate**.

- The setup is centralized towards the **Owner** address which might be an individual or another DAOs component.

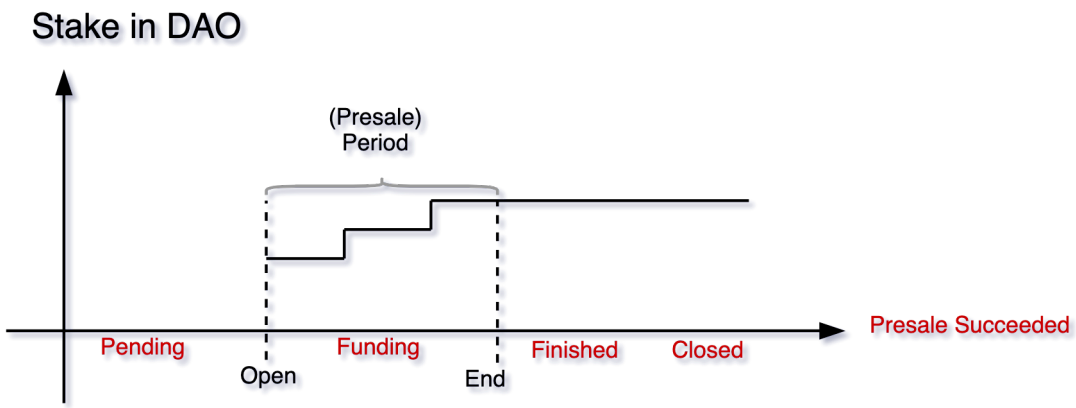
BalanceRedirectPresale

The fundraising campaign is preceded by a presale phase. With the scenario deployed by the **EOPBCTemplate** this step is mandatory. The presale always succeeds and lasts until the period has exceeded and only then ultimately enables investors to buy or sell tokens from the MarketMaker contract. The presale contract is a value store and keeps funds until the period is over. Closing the presale splits contributions to an amount initially assigned to the beneficiary with the rest being transferred to the fundraising reserve.

The following two images depict the presale stages and the timing configuration including the token vesting.



Presale Stages



Presale Timing

The presale proceeds in the following stages:

- `PENDING` - The presale is not yet open. Contributions are not yet accepted.
- `FUNDING` - The presale is open. Contributions are accepted.
- `FINISHED` - The presale duration ended and the goal has been reached. Waiting for someone to close the presale.
- `CLOSED` - The presale has reached its goal and it has been closed by someone, transferring a number of tokens to the beneficiary (Board Vault) and the rest to the Fundraising reserve. Shareholder tokens are minted and assigned vested to contributors. Trading with the MarketMaker is finally opened.

The following properties have been identified:

- The initial presale configuration is critical, set by the **DAO deployer** and must be verified by participants before contributing to a presale. For example, the **DAO deployer** may initially configure the presale to transfer 100% of contributed token to the **Beneficiary**/**Owner** account instead of providing it

  as collateral to the reserve. This will give the Board direct control over the funds instead of withdrawals being restricted by the `Tap` contract.

- The presale phase can be set to open at a specific date or when opening it manually. After the defined presale period trading with the Fundraising MarketMaker can be opened.
- **Anyone** can contribute to the presale via `Controller` (`CONTRIBUTE_ROLE`).
- Investors (**Shareholders**) cannot request a refund. This may make it particular hard for early investors to invest in the presale as the return might uncertain.
- **Shareholder** tokens are minted when processing the contribution. Tokens vest immediately and are therefore directly available to contributors. If the bonded token is used to derive stake for a contributor then this might give contributors voting power even before the presale ends.
- **Shareholder** do not have any control of the fundraising DAO.
- Permissions are managed by the **Owner**.
- Contract upgraded must be performed via the DAO/ `AragonApp` update mechanisms. Upgrades can be performed by the **Owner** without **Shareholder** consensus.
- **Beneficiary** can be updated by **Owner**.
- Unauthenticated state-changing functionality is protected by a reentrancy guard.
- Investors might have an incentive to monitor the presale and only contribute close to the end of the period to make sure the presale is a success.
- Depending on the amount of Shareholder (bonded), token available, permissions and power associated with the bonded token, single individuals might become a majority Stakeholder at a fixed rate from the presale as there is no limit for individual buyers. Investors might want to consider that before or even after they invest someone might be able to buy close to all available tokens at minimal risk of losing funds.

- The bonded token is not created and controlled by the system. The token might be configured in a way that allows an entity malicious activity. The template under audit does not enforce security on this token

template under audit does not enforce security on this token.

6 Issues

Each issue has an assigned severity:

- **Minor** issues are subjective in nature. They are typically suggestions around best practices or readability. Code maintainers should use their own judgment as to whether to address such issues.
- **Medium** issues are objective in nature but are not security vulnerabilities. These should be addressed unless there is a clear reason not to.
- **Major** issues are security vulnerabilities that may not be directly exploitable or may require certain conditions in order to be exploited. All major issues should be addressed.
- **Critical** issues are directly exploitable security vulnerabilities that need to be fixed.

6.1 EOPBCTemplate - permission documentation inconsistencies Major ✓ Fixed

Resolution
Fixed with aragonone/fundraising@ `bafe100` by adding the undocumented and deviating permissions to the documentation.

Description

Undocumented

The template documentation provides an overview of the permissions set with the template. The following permissions are set by the template contract but are not documented in the accompanied `fundraising/templates/externally_owned_presale_bonding_curve/README.md`.

TokenManager
code/fundraising/templates/externally_owned_presale_bonding_curve/contracts/EO
L221

```
_createPermissions(_acl, grantees, _fundraisingApps.bondedTokenManage
_acl.createPermission(_fundraisingApps.marketMaker, _fundraisingApps.
```

code/fundraising/templates/externally_owned_presale_bonding_curve/eopbc.yaml:L L44

```
- app: anj-token-manager
  role: MINT_ROLE
  grantee: market-maker
  manager: owner
- app: anj-token-manager
  role: MINT_ROLE
  grantee: presale
  manager: owner
- app: anj-token-manager
  role: BURN_ROLE
  grantee: market-maker
  manager: owner
```

Inconsistent

The following permissions are set by the template but are inconsistent to the outline in the documentation:

Controller

`owner` has the following permissions even though they are documented as not being set.

App	Permission	Grantee	Manager
Controller	UPDATE_BENEFICIARY	NULL	NULL
Controller	UPDATE_FEES	NULL	NULL
Controller	ADD_COLLATERAL_TOKEN	Owner	Owner
Controller	REMOVE_COLLATERAL_TOKEN	Owner	Owner
Controller	UPDATE_COLLATERAL_TOKEN	Owner	Owner
Controller	UPDATE_MAXIMUM_TAP_RATE_INCREASE_PCT	NULL	NULL
Controller	UPDATE_MAXIMUM_TAP_FLOOR_DECREASE_PCT	NULL	NULL
Controller	ADD_TOKEN_TAP	NULL	NULL

Controller	UPDATE_TOKEN_TAP	NULL	NULL
Controller	OPEN_PRESALE	Owner	Owner
Controller	OPEN_TRADING	Presale	Owner
Controller	CONTRIBUTE	Any	Owner
Controller	OPEN_BUY_ORDER	Any	Owner
Controller	OPEN_SELL_ORDER	Any	Owner
Controller	WITHDRAW	NULL	NULL

code/fundraising/templates/externally_owned_presale_bonding_curve/contracts/EO
L240

```
_acl.createPermission(_owner, _fundraisingApps.controller, _fundraisi
_acl.createPermission(_owner, _fundraisingApps.controller, _fundraisi
```

Recommendation

For transparency, all permissions set-up by the template must be documented.

6.2 EOPBCTemplate - AppId of `BalanceRedirectPresale` should be different from AragonBlack/Presale namehash to avoid collisions Major ✓ Fixed

Resolution

Fixed with aragonone/fundraising@ `bafe100` by generating a unique APMNameHash for `BalanceRedirectPresale` that does not collide with the one from `Presale` .

Description

The template references the new presale contract with `apmNamehash` `0x5de9bbdeaf6584c220c7b7f1922383bcd8bbcd4b48832080afd9d5ebf9a04df5` . However, this namehash is already used by the aragonBlack/Presale contract. To

avoid confusion and collision a unique `apmNamehash` should be used for this variant of the contract.

Note that the contract that is referenced from an `apmNamehash` is controlled by the `ENS` resolver that is configured when deploying the template contract. Using the same namehash for both variants of the contract does not allow a single registry to simultaneously provide both variants of the contract and might lead to confusion as to which application is actually deployed. This also raises the issue that the `ENS` registry must be verified before actually using the contract as a malicious registry could force the template to deploy potentially malicious applications.

aragonOne/Fundraising:

code/fundraising/templates/externally_owned_presale_bonding_curve/contracts/EO

```
bytes32   private constant PRESALE_ID                    = 0x5de9bbdea
```

aragonBlack/Fundraising:

templates/multisig/contracts/FundraisingMultisigTemplate.sol:L35

```
bytes32   private constant PRESALE_ID              = 0x5de9bbdeaf6584c2
```

`bytes32 private constant PRESALE_ID = 0x5de9bbdeaf6584c220c7b7f1922383bcd8bbcd4b48832080afd9d5ebf9a04df5;`

Recommendation

Create a new `apmNamehash` for `BalanceRedirectPresale`.

6.3 BalanceRedirectPresale - Presale can be extended indefinitely Major Won't Fix

Resolution
This issue was addressed with the following statement:

> *It is a very reasonable concern, but this is the intended behavior. That modification is permissioned and that `OPEN_ROLE` is going to be held by the Aragon Network Dao, so we expect a reasonable use of it. We may document it and make it clear that this is possible.*

Description

The `OPEN_ROLE` can indefinitely extend the Presale even after users contributed funds to it by adjusting the presale period. The period might be further manipulated to avoid that token trading in the MarketMaker is opened.

code/fundraising/apps/presale/contracts/BalanceRedirectPresale.sol:L136-L138

```
function setPeriod(uint64 _period) external auth(OPEN_ROLE) {
    _setPeriod(_period);
}
```

code/fundraising/apps/presale/contracts/BalanceRedirectPresale.sol:L253-L257

```
function _setPeriod(uint64 _period) internal {
    require(_period > 0, ERROR_TIME_PERIOD_ZERO);
    require(openDate == 0 || openDate + _period > getTimestamp64(), ER
    period = _period;
}
```

Recommendation

Do not allow to extend the presale after funds have been contributed to it or only allow period adjustments in `State.PENDING`.

6.4 Repository structure - Create a clean repository containing one Aragon Application unless changes are contributed upstream Medium ✓ Deferred

upstream Medium ✓ Deferred

Resolution
The issue has been deferred pending internal discussion.

Description

The repository is a fork of AragonBlack/fundraising. The main development repository for Aragon Fundraising is the origin repository at AragonBlock. This repository duplicates a state of the upstream repository that can quickly get out of sync and therefore hard to maintain.

It is unclear if both repositories will live side-by-side or if the `BalanceRedirectPresale` variant is contributed upstream.

Recommendation

In case changes are not planned to be contributed upstream it is recommended to create a clean Aragon Application from scratch removing any unused or duplicated files.

6.5 BalanceRedirectPresale - Tokens vest during the Presale phase Medium Won't Fix

Resolution
The issue was addressed with the following statement: > *This presale version is intended to be used along with the Externally Owned Presale and Bonding Curve Template, which doesn't have a Voting app, therefore contributors doesn't have any voting power. The use case is the deployment of Aragon Network Jurors Token* > *(ANJ) for the Aragon Court, which is not going to be active before the presale starts, so we don't see any potential issue here.*

Description

Tokens are directly minted and assigned to contributors during the Presale. While this might not be an issue if the minted token does not give any voting power of some sort in a DAO it can be a problem for scenarios where contributors get stake in return for contributions.

Recommendation

Vest tokens for contributors after the presale finishes. In case this is the expected we suggest to add a note to the documentation to make potential users aware of this behaviour that might have security implications if contributors get stake in return for their investments.

6.6 BalanceRedirectPresale - `setPeriod` uint64 overflow in validation check Medium ✓ Fixed

Resolution

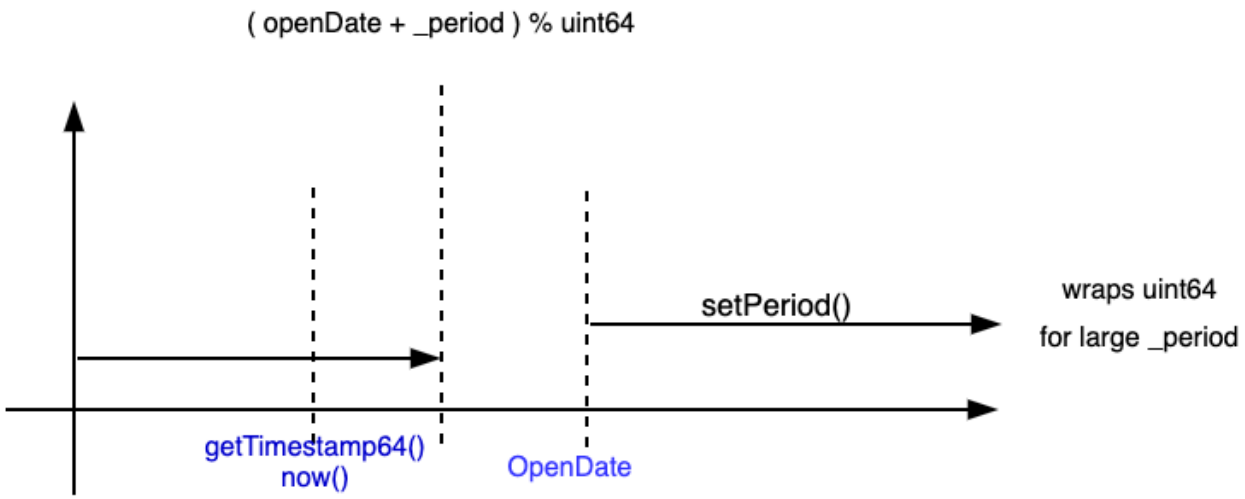
Fixed with aragonone/fundraising@ `bafe100` by performing the addition using `SafeMath`.

Description

`setPeriod()` allows setting an arbitrary Presale starting date. The method can be called by an entity with the `OPEN_ROLE` permission. Providing a large enough value for `uint64 _period` can overflow the second input validation check. The result is unwanted behaviour where for relatively large values of `period` the require might fail because the overflow `openDate + _period` is less than or equal the current timestamp (`getTimestamp64()`) but if high enough it still might succeed because `openDate + _period` is higher than the current timestamp. The overflow has no effect on the presale end as it is calculated against `_timeSinceOpen`.

code/fundraising/apps/presale/contracts/BalanceRedirectPresale.sol:L253-L257

```
function _setPeriod(uint64 _period) internal {
    require(_period > 0, ERROR_TIME_PERIOD_ZERO);
    require(openDate == 0 || openDate + _period > getTimestamp64(), EP
    period = _period;
}
```



Recommendation

Use `SafeMath` which is already imported to protect from overflow scenarios.

6.7 EOPBCTemplate - misleading method names `_cacheFundraisingApps` and `_cacheFundraisingParams`

Minor ✓ Fixed

Resolution
Fixed with aragonone/fundraising@ `0ce7c72` by renaming the functions.

Description

The methods `_cacheFundraisingApps` and `_cacheFundraisingParams` suggest that parameters are cached as state variables in the contract similar to the multi-

step deployment contract used for AragonBlack/Fundraising. However, the methods are just returning memory structs.

code/fundraising/templates/externally_owned_presale_bonding_curve/contracts/EOL300

```
function _cacheFundraisingApps(
    Agent          _reserve,
    Presale        _presale,
    MarketMaker    _marketMaker,
    Tap            _tap,
    Controller     _controller,
    TokenManager   _tokenManager
)
    internal
    returns (FundraisingApps memory fundraisingApps)
{
    fundraisingApps.reserve            = _reserve;
    fundraisingApps.presale            = _presale;
    fundraisingApps.marketMaker        = _marketMaker;
    fundraisingApps.tap                = _tap;
    fundraisingApps.controller         = _controller;
    fundraisingApps.bondedTokenManager = _tokenManager;
}

function _cacheFundraisingParams(
    address       _owner,
    string        _id,
    ERC20         _collateralToken,
    MiniMeToken   _bondedToken,
    uint64        _period,
    uint256       _exchangeRate,
    uint64        _openDate,
    uint256       _reserveRatio,
    uint256       _batchBlocks,
    uint256       _slippage
)
    internal
    returns (FundraisingParams fundraisingParams)
{
    fundraisingParams = FundraisingParams({
        owner:           _owner,
        id:               id,
```

```
        collateralToken: _collateralToken,
        bondedToken:     _bondedToken,
        period:          _period,
        exchangeRate:    _exchangeRate,
        openDate:        _openDate,
        reserveRatio:    _reserveRatio,
        batchBlocks:     _batchBlocks,
        slippage:        _slippage
    });
}
```

Recommendation

The functions are only called once throughout the deployment process. The structs can therefore be created directly in the main method. Otherwise rename the functions to properly reflect their purpose.

6.8 EOPBCTemplate - Pool should be Agent or Reserve Minor ✓ Fixed

Resolution

Fixed with aragonone/fundraising@ `bafe100` by replacing `Pool` for `Reserve` in the documentation.

Description

The documentation refers to an non-existent `Pool` application.

code/fundraising/templates/externally_owned_presale_bonding_curve/README.md: L68

```
| App  | Permission            | Grantee          | Manager
| ---- | --------------------- | ---------------- | ----------------

| Pool | SAFE_EXECUTE          | Owner            | Owner
| Pool | ADD_PROTECTED_TOKEN   | Controller       | Owner
| Pool | REMOVE_PROTECTED_TOKEN | NULL            | NULL
```

```
| Pool | EXECUTE                | NULL             | NULL
| Pool | DESIGNATE_SIGNER       | NULL             | NULL
| Pool | ADD_PRESIGNED_HASH     | NULL             | NULL
| Pool | RUN_SCRIPT             | NULL             | NULL
| Pool | TRANSFER               | MarketMaker      | Owner
```

Recommendation

`Pool` should be `Agent` or `Reserve`.

6.9 EOPBCTemplate - inconsistent storage location declaration

Minor ✓ Fixed

Resolution
Fixed with aragonone/fundraising@ `bafe100` by adding the missing storage location declaration.

Description

`_cacheFundraisingParams()` does not explicitly declare the return value memory location.

code/fundraising/templates/externally_owned_presale_bonding_curve/contracts/EO L286

```
function _cacheFundraisingParams(
    address       _owner,
    string        _id,
    ERC20         _collateralToken,
    MiniMeToken   _bondedToken,
    uint64        _period,
    uint256       _exchangeRate,
    uint64        _openDate,

    uint256       _reserveRatio,
    uint256       _batchBlocks,
    uint256        slippage
```

```
)
    internal
    returns (FundraisingParams fundraisingParams)
```

`_cacheFundraisingApps()` explicitly declares to return a copy of the storage struct.

code/fundraising/templates/externally_owned_presale_bonding_curve/contracts/EO L271

```
function _cacheFundraisingApps(
    Agent          _reserve,
    Presale        _presale,
    MarketMaker    _marketMaker,
    Tap            _tap,
    Controller     _controller,
    TokenManager   _tokenManager
)
    internal
    returns (FundraisingApps memory fundraisingApps)
{
    fundraisingApps.reserve            = _reserve;
    fundraisingApps.presale            = _presale;
    fundraisingApps.marketMaker        = _marketMaker;
    fundraisingApps.tap                = _tap;
    fundraisingApps.controller         = _controller;
    fundraisingApps.bondedTokenManager = _tokenManager;
}
```

Recommendation

Storage declarations should be consistent.

6.10 EOPBCTemplate - Keep the template as closely aligned to the audited `Company` DAO-Template provided by Aragon

Minor ✓ Fixed

Resolution

The issue was addressed with aragonone/fundraising@ `bafe100` changing the main deployment method from `installFundraisingApps` to `newInstance` aligned with the Aragon/DAO-templates naming.

Description

The EOPBCTemplate is a simplified variant of the AragonBlack/FundraisingMultisigTemplate. The FundraisingMultisigTemplate is initially based on the Aragon/DAO-templates/company-board template.

Please note that the DAO-templates provided by Aragon have recently been audited.

The EOPBCTemplate is similar to the setup established with Aragon/DAO-templates/company. The scenario deploys in one step. However, interface names are different to the audited DAO-template variant (`installFundraisingApps` vs `newInstance`). We recommend the template and interface names to be kept as close as possible to the audited `company` template which established the entry point for deploying a one-step template as `newInstance` .

Recommendation

Take the Aragon/DAO-templates/company template as a starting point and add relevant parts for the presale variant.

6.11 EOPBCTemplate - `EtherTokenConstant` is never used

Minor ✓ Fixed

Resolution

Fixed with aragonone/fundraising@ `bafe100` by removing the `EtherTokenConstant` dependency.

Description

The constant value `EtherTokenConstant.ETH` is never used.

code/fundraising/templates/externally_owned_presale_bonding_curve/contracts/EO

```
import "@aragon/os/contracts/common/EtherTokenConstant.sol";
```

code/fundraising/templates/externally_owned_presale_bonding_curve/contracts/EO

```
contract EOPBCTemplate is EtherTokenConstant, BaseTemplate {
```

Recommendation

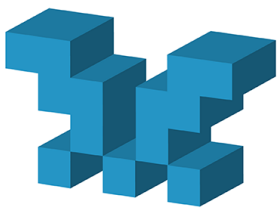
Remove all references to `EtherTokenConstant`.

7 Tool-Based Analysis

Several tools were used to perform an automated analysis of the reviewed contracts. These issues were reviewed by the audit team, and relevant issues are listed in the Issue Details section.

7.1 MythX

MythX is a security analysis API for Ethereum smart contracts. It performs multiple types of analysis, including fuzzing and symbolic execution, to detect many common vulnerability types. The tool was used for automated vulnerability discovery for all audited contracts and libraries. More details on MythX can be found at mythx.io.



7.2 Ethlint

Ethlint is an open-source project for linting Solidity code. Only security-related issues were reviewed by the audit team.

ETHLINT
Analyse Solidity for Style and Security

Below is the raw output of the Ethlint vulnerability scan:

```
$ solium --version
Solium version 1.2.5
```

```
$ solium -d contracts


contracts/EOPBCTemplate.sol
  118:8     warning    Line exceeds the limit of 145 characters
  208:8     warning    Line exceeds the limit of 145 characters
  221:8     warning    Line exceeds the limit of 145 characters
  224:8     warning    Line exceeds the limit of 145 characters
  231:8     warning    Line exceeds the limit of 145 characters
  232:8     warning    Line exceeds the limit of 145 characters
  233:8     warning    Line exceeds the limit of 145 characters
  234:8     warning    Line exceeds the limit of 145 characters
  235:8     warning    Line exceeds the limit of 145 characters
  236:8     warning    Line exceeds the limit of 145 characters
  237:8     warning    Line exceeds the limit of 145 characters
  265:8     warning    Assignment operator must have exactly single sp
  266:8     warning    Assignment operator must have exactly single sp
  267:8     warning    Assignment operator must have exactly single sp
  268:8     warning    Assignment operator must have exactly single sp
  269:8     warning    Assignment operator must have exactly single sp
  289:13    warning    Name 'owner': Only "N: V", "N : V" or "N:V" spa
  290:13    warning    Name 'id': Only "N: V", "N : V" or "N:V" spacin
  292:13    warning    Name 'bondedToken': Only "N: V", "N : V" or "N
  293:13    warning    Name 'period': Only "N: V", "N : V" or "N:V" sp
  294:13    warning    Name 'exchangeRate': Only "N: V", "N : V" or "N
  295:13    warning    Name 'openDate': Only "N: V", "N : V" or "N:V"
  296:13    warning    Name 'reserveRatio': Only "N: V", "N : V" or "N
  297:13    warning    Name 'batchBlocks': Only "N: V", "N : V" or "N
  298:13    warning    Name 'slippage': Only "N: V", "N : V" or "N:V"
```

7.3 Surya

Surya is a utility tool for smart contract systems. It provides a number of visual outputs and information about the structure of smart contracts. It also supports

querying the function call graph in multiple ways to aid in the manual inspection and control flow analysis of contracts.

Below is a complete list of functions with their visibility and modifiers (please use horizontal scroll to view all columns):

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	
BalanceRedirectPresale	Implementation	IsContract, AragonApp, IPres		
└	initialize	External ❗	🛑	onlyInit
└	setOpenDate	External ❗	🛑	auth
└	setPeriod	External ❗	🛑	auth
└	open	External ❗	🛑	auth
└	contribute	External ❗	💵	nonReentrant auth
└	refund	External ❗	🛑	isInitialized
└	close	External ❗	🛑	nonReentrant isInitialized
└	contributionToken	External ❗	NO❗	
└	contributionToTokens	Public ❗		isInitialized
└	state	Public ❗		isInitialized
└	_timeSinceOpen	Internal 🔒		
└	_setOpenDate	Internal 🔒	🛑	
└	_setPeriod	Internal 🔒	🛑	
└	_transfer	Internal 🔒	🛑	
IPresale	Interface				
└	open	External ❗	🛑	NO❗
└	close	External ❗	🛑	NO❗
└	contribute	External ❗	💵	NO❗
└	refund	External ❗	🛑	NO❗
└	contributionToTokens	External ❗		NO❗
└	contributionToken	External ❗		NO❗
EOPBCTemplate	Implementation	EtherTokenConstant, BaseTemplate			
└	\	Public ❗	🛑	BaseTemplate
└	installFundraisingApps	External ❗	🛑	NO❗
└	_proxifyFundraisingApps	Internal 🔒	🛑	
└	_initializePresale	Internal 🔒	🛑	
└	_initializeMarketMaker	Internal 🔒	🛑	
└	_initializeController	Internal 🔒	🛑	
└	_setupCollateral	Internal 🔒	🛑	

```
| └ | _setupFundraisingPermissions | Internal 🔒 | 🛑 | |
| └ | _cacheFundraisingApps | Internal 🔒 | 🛑 | |
| └ | _cacheFundraisingParams | Internal 🔒 | 🛑 | |
| └ | _registerApp | Internal 🔒 | 🛑 | |
```

```
### Legend
```

```
| Symbol | Meaning |
|:--------:|-----------|
| 🛑 | Function can modify state |
| 💵 | Function is payable |
```

Appendix 1 - Disclosure

ConsenSys Diligence (“CD”) typically receives compensation from one or more clients (the “Clients”) for performing the analysis contained in these reports (the “Reports”). The Reports may be distributed through other means, including via ConsenSys publications and other distributions.

The Reports are not an endorsement or indictment of any particular project or team, and the Reports do not guarantee the security of any particular project. This Report does not consider, and should not be interpreted as considering or having any bearing on, the potential economics of a token, token sale or any other product, service or other asset. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. No Report provides any warranty or representation to any Third-Party in any respect, including regarding the bugfree nature of code, the business model or proprietors of any such business model, and the legal compliance of any such business. No third party should rely on the Reports in any way, including for the purpose of making any decisions to buy or sell any token, product, service or other asset. Specifically, for the avoidance of doubt, this Report does not constitute investment advice, is not intended to be relied upon as investment advice, is not an endorsement of this project or team, and it is not a guarantee as to the absolute security of the project. CD owes no duty to any Third-Party by virtue of publishing these Reports.
PURPOSE OF REPORTS The Reports and the analysis described therein are created solely for Clients and published with their consent. The scope of our review is limited to a review of Solidity code and only the Solidity code we note as

review is limited to a review of Solidity code and only the Solidity code we note as being within the scope of our review within this report. The Solidity language itself remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond Solidity that could present security risks. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty.

CD makes the Reports available to parties other than the Clients (i.e., "third parties") – on its website. CD hopes that by making these analyses publicly available, it can help the blockchain ecosystem develop technical best practices in this rapidly evolving area of innovation.

LINKS TO OTHER WEB SITES FROM THIS WEB SITE You may, through hypertext or other computer links, gain access to web sites operated by persons other than ConsenSys and CD. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites' owners. You agree that ConsenSys and CD are not responsible for the content or operation of such Web sites, and that ConsenSys and CD shall have no liability to you or any other person or entity for the use of third party Web sites. Except as described below, a hyperlink from this web Site to another web site does not imply or mean that ConsenSys and CD endorses the content on that Web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the Reports. ConsenSys and CD assumes no responsibility for the use of third party software on the Web Site and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any outcome generated by such software.

TIMELINESS OF CONTENT The content contained in the Reports is current as of the date appearing on the Report and is subject to change without notice. Unless indicated otherwise, by ConsenSys and CD.