aragonOS Code Review

Jordi Baylina <jordi@baylina.cat>
Barry Whitehat<barrywhitehat@protonmail.com>
Adrià Massanet <adria@codecontext.io>
Arthur Lunn <Quazia@giveth.io>
Oswald Jones<oz@giveth.io>
RJ Ewing <perissology@protonmail.com>
Griff Green <griff@giveth.io>

Release
V2.0 / 2018-10-30

Executive Summary

In February, Aragon asked us to do a code review for the aragonOS framework as well as the Finance, Vault, Voting and Token Manager applications. We were very impressed with the quality of the code. It is without question one of the most advanced smart contract systems in the space and makes extensive use of many new functionalities within Solidity and at the EVM level. Especially notable is the secure way to deploy a core controller that adds upgradability, the access control pattern (ACL) and its flexible execution engines.

We spent 3 weeks in March reviewing the code and found 1 critical issue, 3 high severity issues, 4 medium severity issues and 27 low severity issues. We also made 39 comments to the code about things that could be improved or at least things that we believe require a clarification or a deeper look. The critical issue, if exploited, could stop all Aragon DApps deployed if it was not corrected.

After our deep dive into the code, we discussed the issues with the Aragon team and they worked to fix all the issues throughout April. We reviewed these fixes and can say that the Aragon team has corrected all of the important security issues that we found.

The biggest worry we have with this framework is the possible misunderstanding of this framework by the developers that decide to use it to deploy their DApps. We strongly recommend that all Aragon DApp developers, especially those that are early pioneers, review the code, try to understand how it works, and do not treat it as a black box. We recommend that the Aragon core team and community contributors make a special effort to add useful clarifications in the code. Better documentation will greatly contribute to the high level of security that this framework is designed to provide.

From the architectural perspective, we believe that this code base is an incredibly well designed first iteration. However, after the first set of DApps are developed on top of this framework, it is likely that more improvements and refactors will need to be done to accommodate the desired usage of the DApp developers.

On the version 2.0 of this document, an additional review on changes introduced in AragonOS4.0.0-beta2 are analyzed (Annex B)

Table of Contents

1. Introduction

1.1 About this review

The primary focus of this audit was to analyze the aragonOS contracts. We focused our efforts on the Solidity contracts located at https://github.com/aragon/aragonOS/tree/v3.0.1/contracts (bf434fc6a0720bff091a5d3abfc1190e5ec3f342)

For the aragon-apps we started at commit:
https://github.com/aragon/aragon-apps/tree/3542fea317c641f801cef6ca365f52fda97c7323

We concluded the audit by reviewing only the changes related to the initially found issues up to https://github.com/aragon/aragonOS/tree/be4522b264d2224279fbfe582f424f3c6c8d3c9b and
https://github.com/aragon/aragon-apps/tree/05bf9e858e5234429f13bda22b7db9f79cba2f9e.

An additional review on changes introduced for AragonOS4.0.0-beta2 can be found in the Annex B.

We assembled a multidisciplinary team to get deep into the project and find as many issues as we could in the three weeks we set aside to review this code.

Scope: Solidity Contracts

> We critically analyzed the aragonOS contracts and aragon-app contracts to understand the system. We ran local test deployments and created proof of concept apps to try out the aragon system first hand.
>
> Solidity audits are something our core team is most experienced in. We followed our pattern of searching for bugs and performing a critical review to ensure the code matches the intentions of the developers.

Out of Scope:

> As the name implies, the aragonOS is meant to be a foundation layer for business application built on its ACL and kernel components. The following was not in scope:
>
> - Actual business applications and real world scenarios.
> - Documentation for business use cases (still underway when this review began)
> - User interface and user experience.
> - Deployment process.
> - Even though the team went deep into solidity, we can not assume there are zero errors in the solidity compiler.

1.2 Disclaimer

This document includes opinions and recommendations. All the opinions are our own and independent of the Aragon team. We found issues in the Solidity code and general design of system but that does not mean that we found all the issues.

1.3 Team Introduction

Jordi Baylina <jordi@baylina.cat>
@jbaylina
Over 30 years experience as a developer, and over two years spent researching Ethereum. One of the strongest Solidity developers in the world. Co-founder of the White Hat Group which played a major role in rescuing funds from TheDAO and Parity Multisig Hacks, and author of the MiniMe token contract, the elliptic curves Solidity library as well as numerous other established contracts in the space. Currently working towards formalizing the ERC-777 token standard.

Barry WhiteHat <barrywhitehat@protonmail.com>
@barrywhitehat
An anonymous white hat.

Adrià Massanet <adria@codecontext.io>
@codecontext
Over 18 years experience within the areas of security, cryptography, and digital identity software development. An established computer science engineer with several security audits conducted over the past year.

Oswald Jones <oz@giveth.io>
@oz
Blockchain problem solver with a decade of experience working on interesting projects in the startup and enterprise world. Giveth contributor.

Arthur Lunn <alf40k@gmail.com>
@arthur.lunn
Currently lead of social_coding with Giveth. Has contributed as an independent contractor and open source contributor to multiple projects in the space.

RJ Ewing <perissology@protonmail.com>
@perissology
Solidity and full-stack developer working with Giveth.

Griff Green <griff@giveth.io>
@griff
A well connected and respected person in the Ethereum space. Community manager for TheDAO, led the cleanup effort for TheDAO Hack from every angle, co-founder of the White Hat Group and Giveth. Holds a Masters Degree in Digital Currency and a frequent contributor to many projects and security audits in the space.

1.4 Methodology

In order to provide a focused analysis on the provided smart contracts we undertook the following steps:

- Understand the provided source code
- Define the primary and secondary objectives of the work, regarding the current document deliverable and the kind of analysis to be done
- Read and analyze the previous security audits done
- Iterate over a frozen code base
    - internal analysis
    - share findings with the Aragon team
    - clarify possible misunderstandings with them
    - analyze their fixes, when applicable
- Consolidate, write and deliver the report

The following table describes the analysis performed to the current code, and the depth to which each analysis has been done. When full, we feel the analysis is complete; when partial, we feel further analysis would be beneficial, and when supplemental we feel that the task is not in the scope of analysis but it has been also reviewed

Type	Description	Depth
Standard code review	includes the common issues that are usually reviewed in all applications: code readability, specification matching, duplicated code, parameter checking, precondition assertions, code conventions, documentation, testing, code coverage, overflow/underflows, unused variables, bounds checking, etc...	Full
Ethereum security specifics	Specific issues related to Solidity and the Ethereum execution environment, see Appendix A.	Full
Standards	Standards fulfillment	Full
Standard security review	Includes defining assets to be protected and checking STRIDE attacks on the system[1]	Partial
Application logic	Check if code logic matches with the expected behaviour defined in the documentation	n/a
(Crypto)Economics	Checks for (crypto)economic points of failure, and weakness in punish/incentivization systems	Supplemental
Decentralization	Checks for centralized points of failure	Supplemental

[1] https://en.wikipedia.org/wiki/STRIDE_(security)

Simulation	Simulate the system	Supplemental

Our findings are presented in the form V0##-XX, R00#-XX, and are specified in the Review Results section of the document, using the following pattern:

Description

Description of the finding

Location

Where is located (if applicable)

Severity

The threats have been classified into five groups (Critical, High, Medium, Low, Comment) depending on the impact and the likelihood of the event. Sometimes it is really complicated to guess the likelihood or the impact of the finding. We use OWASP threat classification.

	Unknown Likelihood	Low Likelihood	Medium Likelihood	High Likelihood
High Impact	Medium	Medium	High	Critical
Medium impact	Medium	Low	Medium	High
Low impact	Comment	Comment	Low	Medium
Unknown Impact	Medium	Medium	Medium	Medium

Status

“Fixed” means that the reviewing team considers that the issue has been repaired. “Not fixed” means that the issue is still present.

Comments

Our thoughts on the issue. The recommendations provided are not based on a detailed and deep analysis of the underlying problem, therefore they should be considered as a reference and not a definitive solution.

2. Review Results

ID	Description	Type	Status
V001-CR	Delegate Executors can be killed by anybody	Critical	Fixed
V002-HI	Irrevocable permissions	High	Fixed
V003-HI	Function receiveApproval can exploit orgs/users with 0xFFF approval	High	Fixed
V004-HI	Possible overflow in voteTime	High	Fixed
V005-MD	Cannot easily view permissions	Medium	Fixed
V006-MD	AppManager can upgrade a pinned app	Medium	Not fixed
V007-MD	CallScript buffer overflows	Medium	Fixed
V008-MD	Can create vote before initialize	Medium	Fixed
V009-LW	Issues if Solidity is upgraded	Low	Not fixed
V010-LW	Previous repo releases are not fixable	Low	Not fixed
V011-LW	ACL does not always return false when calling faulty ACLOracle	Low	Fixed
V012-LW	Min period of 1 sec is too short	Low	Fixed
V013-LW	_reference is lost for a single payment	Low	Fixed
V014-LW	Race condition exists for payment recipients	Low	Not fixed
V015-LW	Forward is too permissive	Low	Not fixed
V016-LW	Should not permit anything if no Kernel or ACL is defined	Low	Not fixed
V017-LW	Create permissions granted indefinitely to factories	Low	Fixed
V018-LW	Potential overflow when setting new period	Low	Fixed
V019-LW	Incorrect percentage check	Low	Fixed
V020-LW	Solidity forced to 0.4.18	Low	Not fixed
V021-LW	Potential infinite loop in ACL logic	Low	Not fixed
V022-LW	encodeParams can conflict w/ the reserved param ids (200+)	Low	Not fixed
V023-LW	No way to revoke all granted permissions for a role	Low	Not fixed
V024-LW	No way to restrict deposits in Finance app	Low	Not fixed
V025-LW	tokenFallback can clutter with gas payment	Low	n /a
V026-LW	Finance app doesn’t check if vault & etherToken are valid contracts	Low	Fixed
V027-LW	No easy way to debug evalParams	Low	Fixed
V028-LW	No repeat number is recorded for payments	Low	Fixed
V029-LW	ACL is based off of user input	Low	Not fixed
V030-LW	TokenManager can be given a non-transferrable token	Low	Fixed
V031-LW	No historical data for revoking vesting	Low	Fixed
V032-LW	Anyone can call TokenManager.onTransfer and create a log	Low	Fixed
V033-LW	TokenManager.onTransfer marked as view	Low	Fixed
V034-LW	Can't call send or transfer on aragon app	Low	Fixed
V035-CM	Failsafe mode	Comment	Not fixed
V036-CM	Move ACL param logic to oracle	Comment	Not fixed
V037-CM	Isolate apps that are granted powerful permissions	Comment	Not fixed

V038-CM	Cache variables are in AragonApp's storage	Comment	Fixed
V039-CM	Two step ACL changePermissionManager	Comment	Not fixed
V040-CM	Vault requestAllowance	Comment	Fixed
V041-CM	No need to encode/decode in executors	Comment	Not fixed
V042-CM	Return execute payment after payments are complete	Comment	Not fixed
V043-CM	Amount is not included in authP	Comment	Fixed
V044-CM	No role to delete name in APMRegistry	Comment	Not fixed
V045-CM	ACLOracle.canPerform doesn't receive how param	Comment	Fixed
V046-CM	Payments are not always expenses	Comment	Not fixed
V047-CM	Budget setting is dependant on period	Comment	Not fixed
V048-CM	No escapeHatch for ether in Finance App.	Comment	Fixed
V049-CM	Data used for Finance app tests is not realistic	Comment	Not fixed
V050-CM	Provide reason for imports used for child classes	Comment	Fixed
V051-CM	Distinguish between pinned or upgradable	Comment	Fixed
V052-CM	Truffle console should be truffle dev	Comment	Fixed.
V053-CM	Inconsistent tests between aragonOS and aragon-apps	Comment	Fixed
V054-CM	Redundant hasPermission functions in ACL	Comment	Not fixed
V055-CM	EVMScriptRegistry manager naming	Comment	Not fixed
V056-CM	setPaymentDisabled can also enable the payment	Comment	Not fixed
V057-CM	Consistent naming between newAppProxy and newAppProxyUpgradable	Comment	Fixed
V058-CM	getBudget returns remainingBudget	Comment	Fixed
V059-CM	Confusing "payment struct" name	Comment	Not fixed
V060-CM	Confusing ttl param name	Comment	Fixed
V061-CM	Redundant conditional	Comment	Fixed
V062-CM	TokenManager roles are a bit confusing	Comment	Not fixed
V063-CM	Simplify conditional	Comment	Fixed
V064-CM	Non-transferrable tokens are not clear	Comment	Fixed.
V065-CM	Dangling variable in AppProxyUpgradeable	Comment	Fixed
V066-CM	Vault assert should be require	Comment	Fixed
V067-CM	Use forced casting	Comment	Not fixed
V068-CM	Possible to be granted unexpected permissions	Comment	Not fixed
V069-CM	Use STATICCALL when calling ACLOracle	Comment	Not Fixed
V070-CM	ACL role assignment is unclear	Comment	Not Fixed
V071-CM	APMRegistry should to be an isolated dao	Comment	Not fixed
V072-CM	Vault short circuit	Comment	Fixed
V073-CM	Use safe math	Comment	Fixed

2.1 Critical Severity Issues

[V001-CR] Delegate Executors can be killed by anybody

Location	Severity	Issue	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/evmscript/executors/DelegateScript.sol#L24	Critical	os#250	Fixed. Aragon team will temporarily remove this functionality in the next version.

Description

About https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/evmscript/executors/DelegateScript.sol#L24

There is only one `DelegateScript` instance globally created in `EVMScriptRegistryFactory`. This DelegateScript has the public everybody callable method `execScript(bytes _script, bytes _input, address[] _blacklist)`. If the delegatecall-ed contract method contains a `SELFDESTRUCT`, the destructed contract will be the `EVMScriptRegistryFactory` `baseDel` instance which cannot be redeployed since it's not upgradable.

About https://github.com/aragon/aragonOS/issues/250 fix

The fix, based on the assumption that a function that `SELFDESTRUCTs` called from a `delegatecall` never will return any data (`RETURNDATASIZE` is 0), can be bypassed by doing an extra `delegatecall` jump, see the following exploit, based on current test mocks:

```
contract DyingDelegatorHelper {
    function die() public returns (bool) {
       selfdestruct(0x0);
       return true;
    }
}

contract DyingDelegator is DelegateScriptTarget {
    function exec() public returns (bool) {
       DyingDelegatorHelper helper = new DyingDelegatorHelper();
       address(helper).delegatecall(DyingDelegatorHelper(0).die.selector);
       return true;
    }
}
```

Comments

Consider limiting `DelegateScript` callers to the `EVMScriptRunner`.

2.2 High Severity Issues

[V002-HI] Irrevocable permissions

Location	Severity	Issue	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/acl/ACL.sol#L121	High	os#211	Fixed

Description

By using require for permissions you can reach a situation where the permission depends on the parameters and you cannot revoke the permission because you don't check the specific parameters.

Comments

Should remove `require` permissions.

[V003-HI] Function receiveApproval can exploit orgs/users with 0xFFF approval

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/blob/master/apps/finance/contracts/Finance.sol#L157	High	apps#83	Fixed.

Description

Currently, any user can call the `receiveApproval` function, moving another user's tokens if they have given a 0xFFF approval to the org. Because this is really a hook called by the token contract, only the token should be able to call `receiveApproval`. It is up to the token to have the necessary protections.

Comments

Consider adding a `require(msg.sender == _token)`.

[V004-HI] Possible overflow in voteTime

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/blob/master/apps/voting/contracts/Voting.sol#L265	High	apps#187	Fixed

Description

The `voteTime` parameter is the seconds that a vote will be open for token holders. When checking `_isVoteOpen`, it is possible that `vote.startDate + voteTime` will overflow.

2.3 Medium Severity Issues

[V005-MD] Cannot easily view permissions

Location	Severity	Issue	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/acl/ACL.sol	Medium	os#251	Fixed

Description

For security and recovery situations a fast way to view permissions is necessary. Imagine a rogue employee starts moving funds. It would be good to be able to quickly see what other permissions that user has. Or if something strange is happening, being able to easily retrieve all permissions is helpful.

Comments

A `getPermissionParams` function would be very handy. Given entity, app, and role, tell me the parameters.

[V006-MD] AppManager can upgrade a pinned app

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/kernel/Kernel.sol#L62	Medium	Not fixed Future improvement: aOS 4.0

Description

There are two ways to get the `baseCode` for an app. Typically they would use `AppProxy.getCode()`. However if the proxy instance is registered with the kernel, you can fetch the app addr with `kernel.getApp()`. Thus if you register a `AppProxyPinned` instance with the kernel, then anywhere `kernel.getApp` is called to fetch the app address, it can be replaced with a new address by calling `kernel.setApp`. So the app isn't really pinned because it is replaceable.

[V007-MD] CallScript buffer overflows

Location	Severity	Issue	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/evmscript/executors/CallsScript.sol#L41	Medium	os#263	Fixed

Description

Since no bounding checks are performed for `_script` pointer, you can read data beyond the limit of the length.

Comments

Check that `calldataStart + calldataLength < _script.length` to prevent buffer overflows.

[V008-MD] Can create vote before initialize

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/voting/contracts/Voting.sol	Medium	apps#188	Fixed

Description

A `newVote` can be created and automatically executed without taking a vote if the voting contract has not been initialized. The worry is that this odd feature could be exploited by attackers. Here is an example:

1. A deployer deploys a contract
2. Attacker creates a proposal to send all funds to address X. The quorum for this will be zero.
3. The attacker votes for this proposal using another address so that it can be executed later. But also needs to vote for it in the same block as the vote deadline is also 0.
4. The deployer initializes the contract. and gives it access to move funds from a vault.
5. The attacker executes the vote that they have already passed and takes all the funds.

Comments:

Have a `afterInitalize` modifier. Or use a factory to deploy.

2.4 Low Severity Issues

[V009-LW] Issues if Solidity is upgraded

Location	Severity	Status
Example: https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/kernel/Kernel.sol#L11	Low	Not Fixed Documentation and tooling issue.

Description

Even though the likelihood of this event is “very low”, we consider this issue of medium severity, to give an special attention to it.

There are concerns about criteria changes in future solidity versions. The control that developers have on where the variables are stored is quite limited. Dependency on the multi-inheritance order on the definition of the class may variate the way state variables are stored.

As an example, in `Kernel`, If we swap `KernelStorage` and `Initializable` it would break compatibility. This can easily happen in user defined apps and errors would be very difficult to detect causing all kinds of problems to those upgraded apps.

[V010-LW] Previous repo releases are not fixable

Location	Severity	Issue	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/apm/Repo.sol#L78	Low	os#253	Not Fixed Future improvement for aOS 4.0

Description

Concerning semantic version checks with `isValidBump` in APM `Repo` contract, if you release the version 3.0.0 after the 2.7.0, you will be not able to release a 2.7.1 patch.

Comments

May be a good idea if the `isValidBump` does not check the major version to allow patch fixing of previous ones.

[V011-LW] ACL does not always return false when calling faulty ACLOracle

Location	Severity	Issue	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/acl/ACL.sol#L261	Low	os#252	Fixed

Description

When calling the ACL to check params, the `evalParam` function will always return `false` except when the `ACLOracle` is called which will potentially `revert`.

Comments

Using an assembly call to the `ACLOracle` and checking if the call succeeded or failed, will provide consistent handling of the params. May want to limit the `gas` provided to the `ACLOracle` as well.

[V012-LW] Min period of 1 sec is too short

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/blob/master/apps/finance/contracts/Finance.sol#L119	Low	apps#240	Fixed

Description

Min period of 1 sec for finance purposes makes no sense and it is possible to get into race conditions where a payment can not be made (due to the fact that you can only advance 20 periods per tx).

Comments

A min period of 1 day should be more than enough.

[V013-LW] _reference is lost for a single payment

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/blob/master/apps/finance/contracts/Finance.sol#L220	Low	apps#219	Fixed

Description

For a single payment in Finance app, you lose the `_reference` param.

Comments

Consider passing `_reference` to `_makePaymentTransaction` so the `_reference` is logged.

[V014-LW] Race condition exists for payment recipients

Location	Severity	Status
https://github.com/aragon/aragon-apps/blob/master/apps/finance/contracts/Finance.sol	Low	Not Fixed No action will be taken. Inherent ethereum flaw.

Description

If an organization using the Finance app owes more money out to receivers than it has, then receivers may begin racing each other hoping to get to any funds first as soon as they are available using the `receiverExecutePayment` function.

Comments

Avoiding these kinds of races can be done by defining and implementing a better policy for payments when there is not enough money. Maybe prioritising older receivers or the ones with more reputation. Or may be allowing only to withdraw a portion of the funds. Any of these solutions may require a large development effort for something that may

be complex and not worthy. So keeping the code as is, is probably the best decision at this stage, but users of this contract should be aware of this flaw.

[V015-LW] Forward is too permissive

Location	Severity	Status
https://github.com/aragon/aragon-apps/blob/master/apps/token-manager/contracts/TokenManager.sol#L182	Low	Not Fixed No action will be taken. Documentation issue.

Description

There are some great use cases for forwarding evm script functions. But this functionality should not be included as a general default with the TokenManager. It violates the single responsibility principle. It allows a function to return very different outputs.

Comments

This functionality might be better placed outside the TokenManager in a contract specifically designed for the specific use cases it is intended for. Creating multiple instances of employees or IDs are two examples.

[V016-LW] Should not permit anything if no Kernel or ACL is defined

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/apps/AragonApp.sol#L29	Low	Not Fixed See: https://github.com/aragon/aragonOS/issues/325 Documentation issue.

Description

The Kernel and ACL are central to the operation of Aragon apps and so apps should be able to expect that they are defined. If the Kernel or ACL are not defined, then nothing should be allowed.

Comments

Aragon's argument about this issue is that an application should be able to run in standalone mode (Without an ACL defined). This seems a little dangerous (but not an issue by itself) since this permits any operation when this variable is set to zero (the initial value). In any case, this should be very well documented in the code and in the usage handbook.

[V017-LW] Create permissions granted indefinitely to factories

Location	Severity	Issue	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/factory/DAOFactory.sol#L51 https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/factory/EVMScriptRegistryFactory.sol#L40	Low	os#217	Fixed We recommend adding a test for this fix

Description

If `regFactory` is provided, then `initialRoot` is set to `this`. `this` is then passed to `dao.initialize` which will grant the create permissions role to the factory contract.

Comments

We should revoke the `acl.CREATE_PERMISSIONS_ROLE` permission in both the dao and evm registry factories.

[V018-LW] Potential overflow when setting new period

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/blob/master/apps/finance/contracts/Finance.sol#L130 https://github.com/aragon/aragon-apps/blob/master/apps/finance/contracts/Finance.sol#L446	Low	apps#254	Fixed

Description

It's possible to overflow when initializing the Finance app. During initialization, the call to `_newPeriod` could overflow where `endTime` is set.

Comments

The `setPeriodDuration` function should have a max `_periodDuration` to prevent overflows.

[V019-LW] Incorrect percentage check

Location	Severity	PR	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/voting/contracts/Voting.sol#L64	Low	apps#99	Fixed

Description

Percentage integers can have a value of 1 so should not check `> 1`.

Comments

Should be `>0` instead of `>1`.

[V020-LW] Solidity forced to 0.4.18

Location	Severity	Status
All contracts.	Low	Not Fixed No action. Will upgrade solc if necessary in future deployments / upgrades

Description

If there is an error in solidity version 0.4.18 and an upgrade is needed, this will need to be updated. It's not clear for aragonOS users (developers) what they need to do in this case.

[V021-LW] Potential infinite loop in ACL logic

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/acl/ACL.sol#L285	Low	Not Fixed Future improvement: aOS 4.0

Description

Potential infinite loop in ACL logic.

Comments

Ensure that condition is always advancing to prevent getting into an infinite loop.

[V022-LW] encodeParams can conflict w/ the reserved param ids (200+)

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/acl/ACL.sol#L207	Low	Not Fixed Future improvement: aOS 4.0

Description

In `ACL._saveParams` suggest limit `_encodeParams` to 50-100 params. This will help prevent consuming too much gas as well as prevent conflicts with the reserved param ids.

[V023-LW] No way to revoke all granted permissions for a role

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/acl/ACL.sol	Low	Not Fixed Future improvement: aOS 4.0

Description

There is no easy way to `revokeAll grantedPermissions` for an app, role combination. This functionality is usually needed in emergency situations.

Comments

Easy fix is to have an "era" for each `roleHash`. To `revokeAll` for a `roleHash`, just increase the era.

[V024-LW] No way to restrict deposits in Finance app

Location	Severity	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/finance/contracts/Finance.sol#L140	Low	Not Fixed Fix is UI level

Description

In Finance app there is a possible attack to clutter transactions by sending one wei transactions. The problem here is that you could overflow the UI.

Comments

Should have a deposit role in Finance app to restrict the 3 deposit functions.

[V025-LW] tokenFallback can clutter with gas payment

Location	Severity	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/finance/contracts/Finance.sol#L182	Low	Not Fixed Fix is UI level

Description

In the Finance app, the `tokenFallback` can log an event with only gas payment, leading to cluttering of the UI.

Comments

Prevent the event from being triggered if `msg.value == 0`.

[V026-LW] Finance app doesn't check if vault & etherToken are valid

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/finance/contracts/Finance.sol#L116	Low	apps#256	Fixed

Description

Finance app doesn't check if `vault` and `etherToken` are valid.

Comments

The Finance app's `initialize` function should check `vault` and `etherToken != 0`. Alternatively, it may be better to use kernel for `vault` and `etherToken`.

In addition code has been changed since time of initial audit to no longer rely on an ethertoken as the vault has been generalized. The generalized changes to the vault have not been audited.

[V027-LW] No easy way to debug evalParams

Location	Severity	Issue	Status
https://github.com/aragon/aragonOS/blob/e271a2a2e3d4528fe00d79b5dbbc7aefeea90b27/contracts/acl/ACL.sol#L228	Low	os#251	Fixed

Description

No easy way to debug `evalParams`.

Comments

Making the ACL `evalParams` function external will help with debugging.

[V028-LW] No repeat number is recorded for payments

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/finance/contracts/Finance.sol#475	Low	apps#257	Fixed

Description

Payment repeat number is not recorded in the transaction

Comments

The ordinal number of payment is useful information for specifying one payment among the others that come before and after.

[V029-LW] ACL is based off of user input

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/e271a2a2e3d4528fe00d79b5dbbc7aefeea90b27/contracts/acl/ACL.sol#L285	Low	Not Fixed User documentation

Description

ACL can make buggy permissions that can bypass the permission.

Comments

ACL is based off of user input and test coverage does not extend to modifiers.

Analyze the following example: create a parameter with `LOGIC_OP_PARAM_ID = param.id`. But with `_param.op` not equal to `Op.IF_ELSE, Op.Not , Op.OR, Op.And,Op.XOR` will result in `r2` being returned when `evalLogic` gets called from `evalParam` bypassing the `param.id` check. This is a good one for the underhanded aragonOS competition.

[V030-LW] TokenManager can be given a non-transferrable token

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/token-manager/contracts/TokenManager.sol#L5	Low	apps#280	Fixed

9			

Description

The TokenManager app is designed to handle transferability of the token. This requires that transfers are enabled on the token.

Comments

Check `token.transfersEnabled` on initialize in TokenManager.

[V031-LW] No historical data for revoking vesting

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/token-manager/contracts/TokenManager.sol#L158	Low	apps#281	Fixed

Description

When revoking a vesting, there is no easy way to see historical data of revoked `TokenVesting` such as `_cliff`, `_start`, `amount`, etc.

[V032-LW] Anyone can call TokenManager.onTransfer and create a log

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/token-manager/contracts/TokenManager.sol#L204	Low	apps#282	Fixed

Description

The TokenManager function `onTransfer` is a hook provided by the MiniMe.

Comments

Thus it should check `msg.sender == token` to prevent anyone from creating logs if `logHolders` is true.

[V033-LW] TokenManager.onTransfer marked as view

Location	Severity	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/token-manager/contracts/TokenManager.sol#L208	Low	Fixed

Description

If `logHolders == true`, the `onTransfer` function in the `TokenManager` will create a log.

Comments

Thus the function should not be marked as `view`.

[V034-LW] Can't call send or transfer on aragon app

Location	Severity	Issue	Status
Anywhere a delegate proxy is used	Comment	OS#281	Fixed (with new functionality that may need deeper review)

Comment

Send or transfer only permits 2300 gas but delegate proxy requires 10000 + execution costs. This means that you can not use the re-entrancy safe `transfer` and `send` methods provided by solidity to call an AragonApp. You would need to manually limit the gas with `vault.call.gas(16000).value(msg.value)()`.

This issue, has been fixed recently in https://github.com/aragon/aragonOS/pull/281 . Please note that this PR also includes the token recoverability functionality.

The way it has been solved is by always accepting `send` and `transfer` calls with ETH and then creating a mechanism for token recovery including ETH. This mechanism may complicate the contracts aimed to receive ETH by default, and any other application that will have to define a token. Aragon App developers must be very aware of this mechanism.

This mechanism may also bring some issues with clients using the `eth_estimateGas` function.

Please, also note that this functionality was not part of this code review and has not been fully reviewed.

3. Recommendations

3.1 Architecture Comments

[V035-CM] Failsafe mode

Location	Severity	Status
n/a	Comment	Not Fixed Future improvement: aOS 4.0

Comment

Currently, if a fatal error occurs while deploying a new ACL or a faulty ACL is deployed, the kernel can become inoperative without the possibility of redeploying the ACL app.

[V036-CM] Move ACL param logic to oracle

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/	Comment	Not Fixed

acl/ACL.sol#285		

Comment

The ACL param logic adds a lot of complexity to the ACL. In order to simplify such a critical component, consider moving the param logic to an ACLOracle. This way, the complexity is “opt in” and a decision can be made on a DAO by DAO basis if they would like to include the ACL language or use a custom ACLOracle to handle permissions.

[V037-CM] Isolate apps that are granted powerful permissions

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/acl/ACL.sol	Comment	Not Fixed Future improvement: aOS 4.0

Comment

Related to V037-CM, there are a few permissions that are critical to the operation of the DAO. If these permissions fall into the wrong hands, they can break the entire DAO and any apps that share an ACL/Kernel. Sometimes these permissions are given to Apps within the DAO. A bug in 1 of these “privileged apps” can affect the entire DAO. Consider deploying Apps that are granted powerful permissions with a separate kernel, when possible, in order to provide some sandboxing.

[V038-CM] Cache variables are in AragonApp’s storage

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/evmscript/executors/DeployDelegateScript.sol#L11	Comment	Fixed

Comment

`DeployDelegateScript` contains a storage variable `cache` calculated from indexed storage slot 0. Contracts implementing `EVMScriptRunner` (AragonApp) make a `delegatecall` to the scriptExecutor. When this `scriptExecutor` is the `DeployDelegateScript` executor this `cache` variable will potentially conflict with the calling contracts storage in slot 0, thus potentially leading to storage corruption and unexpected state.

With the current AragonStorage layout, there is no conflict because `cache` is a mapping and thus stored in slot `keccak256('0' . index)` and the first storage variable in AragonStorage is `IKernel` and stored in slot 0.

[V039-CM] Two step ACL changePermissionManager

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/acl/ACL.sol#L132	Comment	Not Fixed Future improvement: aOS 4.0

Comment

Use a 2 step process to change permission manager because changing the permission manager for the CREATE_PERMISSION_ROLE on the ACL itself is critical . For an example, see how owner is set here: https://github.com/OpenZeppelin/zeppelin-solidity/blob/master/contracts/ownership/Claimable.sol

[V040-CM] Vault requestAllowance

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/blob/d9dff36a84ff956d21628de0e294233d967404dc5/apps/vault/contracts/Vault.sol#L30	Comment	apps#129	Fixed

Comment

This should be passed the `spender`, but currently `spender == msg.sender`. Requesting an allowance should be for another entity to spend on your behalf. There is no need to `requestAllowance` for yourself, as you can just execute the `transfer`. When accepting `spender` as a param, this should be passed to `authP` as well.

[V041-CM] No need to encode/decode in executors

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/evmscript/EVMScriptRunner.sol#L39	Comment	Not Fixed

Comment

For delegate executor and `EVMScriptRunner.returnedDataDecoded` function, encode/decode is not needed. Simplify the executors return data.

3.2 Functionality Comments

[V042-CM] Return execute payment after payments are complete

Location	Severity	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/finance/contracts/Finance.sol#L272	Comment	Not Fixed

Comment

In Finance app, `executePayment` should only return if payments are complete and there are no more pending payments. Currently, `executePayment` will pay all payments owed (payments can repeat). However, because there is a limit of MAX_PAYMENTS_PER_TX (for gas reasons), you could call `executePayment` and still have payments you can collect.

[V043-CM] Amount is not included in authP

Location	Severity	Issue	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/finance/contracts/Finance.sol	Comment	apps#247	Fixed

Comment

For Finance app `executePayment`, include payment amount in `authP` modifier params.

[V044-CM] No role to delete name in APMRegistry

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/apm/APMRegistry.sol	Comment	Not Fixed No action. Add later if needed.

Comment

May want to add a DELETE_NAME_ROLE role permission.

[V045-CM] ACLOracle.canPerform doesn't receive how param

Location	Severity	Issue	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/acl/ACL.sol#L9	Comment	os#269	Fixed

Comment

Pass `_how` to `canPerform` function for ACL Oracle.

[V046-CM] Payments are not always expenses

Location	Severity	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/finance/contracts/Finance.sol	Comment	Not Fixed

Comment

Regarding the Finance app, outgoing payments are not always expenses. Loan payback is a good example.

[V047-CM] Budget setting is dependant on period

Location	Severity	Status
https://github.com/aragon/aragon-apps/blob/master/apps/finance/contracts/Finance.sol#L558	Comment	Not Fixed

Comment

Finance app's `getRemainingBudget` only checks current period. If you want to change a budget, you must do it before the period is created.

[V048-CM] no escapeHatch for ether in Finance App.

Location	Severity	Commit	Status
https://github.com/aragon/aragon-apps/blob/c44aaff2d59bbe579998e717a2a0040486329864/apps/finance/contracts	Comment	https://github.com/aragon/aragon-apps/co	Fixed. This commit has been refactored in the last source code version.

/Finance.sol		mmit/61e6c2f336745f3e9b84280afffecccc2464bc89#diff-b0ce20bbf524d03600ba9d9339a6e6abR105	

Comment

In Finance app, there is no `escapeHatch` for ether, but it exists for tokens. Consider to add Escape hatch for ether.

3.3 Developer Help Comments

[V049-CM] data used for Finance app tests is not realistic

Location	Severity	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/finance/test/finance.js	Comment	Not fixed

Comment

Currently some the data used for Finance app tests is not realistic. Finance tests need more realistic data. As an example, `repeats up to 10 times every 2 seconds` is not realistic in the context of current state-of-the-art ethereum blockchain.

[V050-CM] Provide reason for imports used for child classes

Location	Severity	PR	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/apps/AragonApp.sol#L5	Comment	os#270	Fixed

Comment

Should have annotation to provide reason for including imports that are only used by child classes.

[V051-CM] Distinguish between pinned or upgradable

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/factory/AppProxyFactory.sol#L8	Comment	Fixed.

Comment

Add parameter to `newAppProxy` event to distinguish if it's pinned or upgradable.

[V052-CM] Truffle console should be truffle dev

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/package.json	Comment	Fixed

Comment

In package.json, truffle console should be truffle dev.

[V053-CM] Inconsistent tests between aragonOS and aragon-apps

Location	Severity	PR	Status
https://github.com/aragon/ara	Comment	apps#255	Fixed

gon-apps/blob/master/shared/test-helpers/ganache-cli.sh https://github.com/aragon/aragonOS/blob/dev/scripts/ganache-cli.sh			

Comment

The ganache-sli.sh script is a different version in aragonOS and aragon-apps. The aragonOS one seems to work better.

[V054-CM] Redundant hasPermission functions in ACL

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/acl/ACL.sol#L157 https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/apps/AragonApp.sol#L25	Comment	Not Fixed

Description

There are 2 `hasPermission` functions in the `ACL`. One uses `bytes` for the `_how` param and the other uses `uint[]` for the `_how` param. In the `AragonApp` contract, `canPerform` converts from `uint[] -> bytes`. Then it is converted from `bytes -> uint[]` in the `ACL`.

Comments

This is not an issue, but we keep this comment here because the cast in assembly is a little strange to read and because we believe that using the same type of data would make the code more consistent.

3.4 Naming Comments

[V055-CM] EVMScriptRegistry manager naming

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/evmscript/EVMScriptRegistry.sol#L14	Comment	Not Fixed Documentation

Comment

REGISTRY_MANAGER_ROLE name does not reflect the importance. This role could adversely affect the whole system. Manager role can make a malicious executor or remove a blacklist. Need to specify implications of every role.

[V056-CM] setPaymentDisabled can also enable the payment

Location	Severity	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/finance/contracts/Finance.sol#L306	Comment	Not Fixed

Comment

In Finance app, `setPaymentDisabled` can also enable the payment. Function name is misleading.

[V057-CM] Consistent naming between newAppProxy and newAppProxyUpgradable

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/apps/AppProxyBase.sol#L39	Comment	Fixed.

Comment

AppProxyBase has a fallback function. This is overridden in `AppProxyPinned`, but does the same thing minus one check. For consistency, either remove from `AppProxyPinned`, or move from base to `AppProxyUpgradeable`.

[V058-CM] getBudget returns remainingBudget

Location	Severity	PR	Status
https://github.com/aragon/aragon-apps/blob/c44aaff2d59bbe579998e717a2a0040486329864/apps/finance/contracts/Finance.sol#L417	Comment	apps#248	Fixed

Comment

In Finance app, `getBudget` returns `remainingBudget` as well.

[V059-CM] Confusing “payment struct” name

Location	Severity	Status
https://github.com/aragon/aragon-apps/tree/c44aaff2d59bbe579998e717a2a0040486329864/apps/finance/contracts/Finance.sol#L30	Comment	Not Fixed

Comment

The name "`Payment`" implies a single payment, but the `Payment` struct is only used for recurring payments. Suggest renaming.

[V060-CM] Confusing ttl param name

Location	Severity	PR	Status
https://github.com/aragon/aragon-apps/tree/c44aaff2d59bbe579998e717a2a0040486329864/apps/finance/contracts/Finance.sol#L330	Comment	apps#250	Fixed

Comment

In Finance app `tryTransitionAccountingPeriod`, `_ttl` should be `maxTransitions`.

3.5 Readability Comments

[V061-CM] Redundant conditional

Location	Severity	PR	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/factory/DAOFactory.sol#L35	Comment	os#271	Fixed

Comment

There is a superfluous ternary being used in the DAOFactory.sol function `newDAO` when only one condition is necessary. Using two separate conditionals that are exactly the same a few lines away from each other suggests that the code in between is imperative to that order, but this is not the case.

[V062-CM] TokenManager roles are a bit confusing

Location	Severity	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/token-manager/contracts/TokenManager.sol#L25	Comment	Not Fixed

Comment

In TokenManager app, `mint`, `issue`, `assign` roles are a bit confusing. It is important to document the purpose and behavior of all roles to help prevent misuse.

[V063-CM] Simplify conditional

Location	Severity	Status
https://github.com/aragon/aragon-apps/blob/c44aaff2d59bbe579998e717a2a0040486329864/apps/token-manager/contracts/TokenManager.sol#L213	Comment	Fixed.

Comment

In TokenManager app, `onTransfer` more clear to use or operator `||` instead negative and `&&` operators. Could instead be: `!transferable || !toCanReceive || (transferableBalance(_from, now) < _amount)`

[V064-CM] Non-transferrable tokens are not clear

Location	Severity	Status
https://github.com/aragon/aragon-apps/blob/0f04123f8a39a245f484f15eba092b19267973e1/apps/token-manager/contracts/TokenManager.sol#L	Comment	Fixed.

315		

Comment

In TokenManager app, non-transferrable tokens are not clear. The function `assign` is a token transfer that bypasses the transferable bool.

[V065-CM] Dangling variable in AppProxyUpgradeable

Location	Severity	PR	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/apps/AppProxyUpgradeable.sol#L7	Comment	os#260	Fixed

Comment

There is an orphan variable that is never referenced which should be removed.

3.6 Optimization Comments

[V066-CM] Vault assert should be require

Location	Severity	PR	Status
https://github.com/aragon/aragon-apps/blob/c44aaff2d59bbe579998e717a2a0040486329864/apps/vault/contracts/Vault.sol#L36	Comment	apps#129	Fixed Vault.sol refactored

Comment

Suggest using `require` instead of `assert`.

[V067-CM] Use forced casting

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/evmscript/ScriptHelpers.sol#L100	Comment	Not Fixed

Comment

ScriptHelpers function `toBytes` can use forced casting which is 2x as efficient. This paradigm is already used other places in codebase.

3.7 Security Comments

[V068-CM] Possible to be granted unexpected permissions

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/apps/AragonApp.sol#L15	Comment	Not Fixed

Comment

If the same role is used to protect more than one method. The parameters passed to the `authP` modifier MUST be in the same order as any other method which is protected by that role because the ACL permissions are all based off of the index of the `_how` params. If not, then it may be possible to have someone be granted unexpected permissions.

[V069-CM] Use STATICCALL when calling ACLOracle

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/acl/ACL.sol#L261	Comment	Not Fixed Future improvement: aOS 4.0

Comment

`STATICCALL` does not preserve `msg.sender` and does not allow for state modifications which can prevent potential side effects.

[V070-CM] ACL role assignment is unclear

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/acl/ACL.sol	Comment	Not Fixed Future improvement: aOS 4.0

Description

When a `permissionManager` is assigned to a role, the permission creator is giving up control on that permission and only the permissionManager can change the permissionManager. This is not made obvious.

[V071-CM] APMRegistry should be an isolated dao

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/v3.0.1/contracts/apm/APMRegistry.sol	Comment	Not Fixed (documentation issue)

Comment

Because the `APMRegistry` must have `CREATE_PERMISSIONS_ROLE` on the `ACL`, a buggy upgrade has the potential to mess up the entire dao. Thus, the `APMRegistry` should be the only app in a dao.

[V072-CM] Vault short circuit

Location	Severity	Status
https://github.com/aragon/aragon-apps/tree/d9dff36a84ff956d21628de0e29423d967404dc5/apps/vault/contracts/Vault.sol	Comment	Fixed (Vault.sol refactored)

Comment

In Vault app it is possible to create a short circuit if the current allowance is the same as the `_amount`. If the amount is 0 it's possible to repeatedly send events.

[V073-CM] Use safe math

Location	Severity	Status
https://github.com/aragon/aragon-apps/blob/c44aaff2d59bbe579998e717a2a0040486329864/apps/token-manager/contracts/TokenManager.sol#L312 https://github.com/aragon/aragon-apps/blob/c44aaff2d59bbe579998e717a2a0040486329864/apps/token-manager/contracts/TokenManager.sol#L224	Comment	Fixed

Comment

In TokenManager app, `vestedTokens` should use safe math on return.
In TokenManager app, use safe math for `isBalanceIncreaseAllowed` function.

Appendix A. Security checks

Assets

Considered as threat targets, smart contracts and user interface application logic should enforce using the underlying Ethereum technology to guarantee the self-protection and non-by-passability of the access controls for the following assets:

ETH	Stored/transferred ether asserts
TOKEN	Stored/transferred token asserts (like ERC20)
ACL	Deployed permissions
Upgradability	Components (smartcontracts) upgradeability
Core Isolation	Applications are not allowed to modify core data
App Isolation	Applications are not allowed to modify data from other applications, only explicit communication channels can be used to share information between them.

Threat Agents

We use the following categories to understand who might attack the smart contracts:

Accidental Discovery	An ordinary user stumbles across a functional mistake in your application, just using a web browser, and gains access to privileged information or functionality.
Privileged account mistake	A mistake (e.g. client library failures or fat finger errors) performed from an account that has special privileges on the smart contracts.
Automated Malware	Programs or scripts, which are searching for known vulnerabilities, and then report them back to a central collection site.
The Curious Attacker	A security researcher or ordinary user, who notices something wrong with the application, and decides to pursue further.
Script Kiddies	Common renegades, seeking to compromise or deface applications for collateral gain, notoriety, or a political agenda.
The Motivated Attacker	Potentially, a disgruntled staff member with inside knowledge or a paid professional attacker.

Whales	Big economic agents abusing blockchain economic models for financial gain.
Miners	Miners abusing blockchain protocols for financial gain.
Big Organizations	Highly trained professionals and criminals cracking or abusing blockchain smart contracts and protocols for financial gain, reputation destruction or geopolitical cyber warfare.

Operational Environment

We use the following operative scenarios when analyzing threats:

Normal operation	Ethereum ecosystem is stable
Network congestion	High amount of pending transactions makes a Ethereum network congestion
High volatility	ETH/$ price is having a high volatility
Fork	A contentious fork of Ethereum that produces 2 seperate networks.
Unexpected fatal bug / global settlement	An unexpected critical bug is found on the smart contracts
Subsystems failure	Auxiliary subsystems (e.g. oracles) fails
Operators failure	Smart contract operators fails or delays the execution
Whale attacks	Economic whales gone wild

Common Security Problems Checklist

Trivial Errors

- Shadowing
- Misnamed variables
- Missing require guards
- Variable confusion
- String Literals/Magic Numbers
- Standards Conformity
- Tx.origin use
- Erroneous Constant State Functions

Race Conditions
- Re-entrancy
- Recursion
- External/Cross Contract Calls

Front Running
Miner specific attacks
- Blocktime Attacks
- Transaction Reordering Attacks

DoS
- Revert
- Block Gas Limit/Dynamic Loops

Integer Overflow & Underflow
Template Misuse/Plugin Trust Issues
Dynamic Array Memory Manipulation
ABI Length Encoding Exploit
Forced Balance Manipulation
Malicious Higher Order Bit Parameters
Economic Attacks
Unsolved Solidity Bugs
Bad function access control
Replay/nonce attacks
Library initialization
Post-deployment ACLs

Appendix B. Audit update 4.0.0-beta2

This Annex contains an audit update of

> aragonOS
> up to commit `05ee60d20bf01e16b2012cf9e95a606a21b9a127`
> tagged as aragonOS@4.0.0-beta.2

and

> aragon-apps
> up to commit `cb199980467c83e7645733c1c27036f9d7344f7b`

This review is not a full review and has been done by checking the particular commits until the frozen audit version, so is not as exhaustive as (in terms of systemic wholeness) the first review done. Nonetheless a detailed search on errors and future problems has been done as expected, as also a review of relevant previous unclosed issues has been also done.

Summary

We spent 1 week in september/october reviewing the code and found 2 high severity issues, and 2 low severity issues. We also made 2 comments to the code about things that could be improved or at least things that we believe require a clarification or a deeper look.

The WHG Audit team sees in this last version a more mature code in terms of security, with a much better user documentation. Nonetheless we still strongly recommend to Aragon DApp developers, to review the code, to understand the security model that the aragonOS provides, and do not treat it as a black box.

ID	Description	Type	Status
V009-LW	Issues if Solidity is upgraded	Low	Fixed
V016-LW	Should not permit anything if no Kernel or ACL is defined	Low	Fixed
V069-CM	Use STATICCALL when calling ACLOracle	Low	Fixed
V074-HI V034-LW	Unexpected behaviour in payable apps when they reverts	High	Fixed
V075-HI	Explicitly authorize external call data	High	Fixed
V076-LW	TRANSFER_ROLE naming is too weak	Low	Fixed
V077-LW	Unusable vault transfer parameter	Low	Fixed
V078-CM	Use require messages to improve troubleshooting	Comment	Fixed
V079-CM	Comment optimizations based on non-evident invariants	Comment	Fixed

Fixed previous unsolved issues

[V009-LW] Issues if Solidity is upgraded

AragonOS is now using the variable name hash static slot assignment technique (`UnstructuredStorage.sol`) not delegating to the compiler the variables slot assignments.

[V016-LW] Should not permit anything if no Kernel or ACL is defined

Now, AragonOS does not authorize operations if a kernel is not set or the application has not been initialized.

[V069-CM] Use STATICCALL when calling ACLOracle

AragonOS is now using STATICCALL when calling external ACL oracles.

New issues found

[V074-HI/V034-LW] Unexpected behaviour in payable apps when they revert

Location	Severity	Status
https://github.com/aragon/aragonOS/blob/58f2c82536a4f8c09751d734c2534eaabb8f6f56/contracts/common/DepositableDelegateProxy.sol	High	Aragon one solved the issue by only accepting this feature by explicitly specifying it with `DepositableStorage.setDepositable(true)`

Description

Related to the *V034-LW Can't call send or transfer on aragon app* issue, Aragon One solved it by creating a new `DepositableDelegateProxy` with this functionality

```
function () payable public {
     // send / transfer
     if (msg.gas < FWD_GAS_LIMIT) {
         require(msg.value > 0 && msg.data.length == 0);
         ProxyDeposit(msg.sender, msg.value);
     } else { // all calls except for send or transfer
         address target = implementation();
         delegatedFwd(target, msg.data);
     }
 }
```

The developer, by default, expects that if the AragonApp target function reverts, the `delegatedFwd` will also fail and the transaction will be reverted.

But, in the case that the client is doing an automatic gas estimation / manual gas input, if `(msg.gas < FWD_GAS_LIMIT)` branch is chosen, the transaction will not revert and the value will be transferred, accepting ether when they must not be accepted. This is something that could lead to some serious problems in some scenarios (like reaching an ICO's max cap or payments that come in too late).

Comment

Creating code that changes its behaviour depending on the gas left is a very advanced feature that should be explicitly activated on developers awareness of its subtleties.

[V075-HI] Explicitly authorize external call data

Location	Severity	Status
https://github.com/aragon/aragon-apps/blob/dbc099047077b96c5d300e2e1eeeb24bef03b685/apps/vault/contracts/Vault.sol#L89	High	Fixed. Aragon team removed this functionality entirely from the Vault, so that it only sends ETH with limit gas: https://github.com/aragon/aragon-apps/pull/492

Description

The authorization made in the `transfer` call

```
authP(TRANSFER_ROLE, arr(_token, _to, _value))
```

is not covering the called `_data`,

```
_to.call.value(_value)(_data)
```

So, the call data sent to the destination contract could be manipulated without being explicitly authorized, leading to security breaches.

[V076-LW] TRANSFER_ROLE naming is too weak

Location	Severity	Status
https://github.com/aragon/aragon-apps/blob/0caee4a32afd129307e7ae93b367117f5dea056b/apps/vault/contracts/Vault.sol#L118	Low	Fixed. Aragon team removed this functionality entirely from the Vault, so that it only sends ETH with limit gas: https://github.com/aragon/aragon-apps/pull/492

Description

`TRANSFER_ROLE` is able to do a `_to.call.value(_value)(_data)` from the vault storing all funds. This is unclear and could lead to unexpected problems if developers reuse this role for other purposes. Other names like `TRANSFER_CALL_PRIV_ROLE` or having two different roles, one for sending _data and another without, could be useful.

Comment

> The audit team suspects that this could potentially open future attacks (like ERC725/ERC820 decentralized registry poisonings) but at this moment is not able to build a threat model based on this.

[V077-LW] Unusable vault transfer parameter

Location	Severity	Status
https://github.com/aragon/aragon-apps/blob/0caee4a32afd129307e7ae93b367117f5dea056b/apps/vault/contracts/Vault.sol#L39	Low	Aragon team fixed it in https://github.com/aragon/aragon-apps/pull/491

Description

The `Vault.deposit(address _token, address _from, uint256 _value)` function is calling internally the `_deposit` function that requires `_from==msg.sender`, so it seems that the `_from` parameter is not usable.

[V078-CM] Use require messages to improve troubleshooting

Location	Severity	Status
*	Comment	Fixed. aragonOS: https://github.com/aragon/aragonOS/pull/441 aragon-apps: https://github.com/aragon/aragon-apps/pull/495 Aragon team opted not to include revert reasons in proxies after gas analysis: https://github.com/aragon/aragonOS/pull/440#issuecomment-430323016 Mostrar menos

Comment

`require/assert` messages are not secure at UI level (because the address of the generator is not known and could be easily generated by a called contract) but could improve and speed up user's troubleshooting specially when AragonOS will be deployed in the mainnet.

[V079-CM] Comment optimizations based on non-evident invariants

Location	Severity	Status
https://github.com/aragon/aragon-apps/blob/dbc099047077b96c5d300e2e1eeeb24bef03b685/apps/vault/contracts/Vault.sol#L39	Comment	Fixed due refactor. https://github.com/aragon/aragon-apps/blob/9a404c3bed8e1449ca819a6fa1764bf02ced1c19/apps/vault/contracts/Vault.sol#L47

Comment

The `isInitialized` check is bypassed by using the invariant that `isDepositable()` is `false` by default (the `DepositableStorage` constructor does not set the slot at the beginning).

Further AragonApps maintainers may be not aware about this optimization and can modify the code without being aware of it, causing unexpected bugs and security breaches.