



AAVE

Aave Governance v3 Contract Review Round 2

Version: 1.0

September, 2023

Contents

Introduction	2
Disclaimer	2
Document Structure	2
Overview	2
Security Assessment Summary	3
Findings Summary	3
A Test Suite	4
B Vulnerability Severity Classification	6

Introduction

Sigma Prime was commercially engaged to perform a time-boxed security review of the AAVE smart contracts. The review focused solely on the security aspects of the Solidity implementation of the contract, though general recommendations and informational comments are also provided.

Disclaimer

Sigma Prime makes all effort but holds no responsibility for the findings of this security review. Sigma Prime does not provide any guarantees relating to the function of the smart contract. Sigma Prime makes no judgements on, or provides any security review, regarding the underlying business model or the individuals involved in the project.

Document Structure

The first section provides an overview of the functionality of the AAVE smart contracts contained within the scope of the security review. A summary followed by a detailed review of the discovered vulnerabilities is then given which assigns each vulnerability a severity rating (see [Vulnerability Severity Classification](#)), an *open/closed/resolved* status and a recommendation. Additionally, findings which do not have direct security implications (but are potentially of interest) are marked as *informational*.

Outputs of automated testing that were developed during this assessment are also included for reference (in the Appendix: [Test Suite](#)).

The appendix provides additional documentation, including the severity matrix used to classify vulnerabilities within the AAVE smart contracts.

Overview

Aave Governance V3 is the most recent version of Aave on-chain governance. It allows users to control the entire ecosystem in a decentralised way. The new version is cross-chain and aims to minimise voting cost.

Aave Governance V3 allows users to create proposals on the Governance contract deployed on Ethereum with different payloads that could belong to different networks. When creating the proposal, the proposer specifies the payloads to execute and the networks where these payloads are going to be executed, as well as the *voting network*, where the votes will be cast. Users can then vote on proposals using their voting power on Ethereum via the *voting machine* contract deployed on the voting network.

Security Assessment Summary

This review was conducted on the files hosted on the [Aave Governance v3 repository](#) and were assessed at commit [fef7af2](#).

A previous security review of the repository had been conducted at commit [a72fa0e](#), the findings of the previous review have been shared in a separate report.

Specifically, the files in scope are as follows:

- `Governance.sol`
- `GovernanceCore.sol`
- `GovernancePowerStrategy.sol`
- `BaseGovernancePowerStrategy.sol`
- `BaseVotingStrategy.sol`
- `VotingPortal.sol`
- `SlotUtils.sol`
- `Executor.sol`
- `PayloadsController.sol`
- `PayloadsControllerCore.sol`
- `DataWarehouse.sol`
- `VotingMachine.sol`
- `VotingMachineWithProofs.sol`
- `VotingStrategy.sol`
- `Errors.sol`

Note: the OpenZeppelin libraries and dependencies were excluded from the scope of this assessment.

The manual code review section of the report is focused on identifying any and all issues/vulnerabilities associated with the business logic implementation of the contracts. This includes their internal interactions, intended functionality and correct implementation with respect to the underlying functionality of the Ethereum Virtual Machine (for example, verifying correct storage/memory layout). Additionally, the manual review process focused on all known Solidity anti-patterns and attack vectors. These include, but are not limited to, the following vectors: re-entrancy, front-running, integer overflow/underflow and correct visibility specifiers. For a more thorough, but non-exhaustive list of examined vectors, see [1, 2].

To support this review, the testing team used the following automated testing tools:

- Mythril: <https://github.com/ConsenSys/mythril>
- Slither: <https://github.com/trailofbits/slither>
- Surya: <https://github.com/ConsenSys/surya>

Output for these automated tools is available upon request.

Findings Summary

The testing team identified no issues during this assessment.

Appendix A Test Suite

A non-exhaustive list of tests were constructed to aid this security review and are provided alongside this document. The `brownie` framework was used to perform these tests and the output is given below.

test_processStorageRoot	PASSED	[1%]
test_processStorageSlot	PASSED	[2%]
test_getStorage	PASSED	[3%]
test_executeTransaction	PASSED	[4%]
test_executeTransaction_selfdestruction	PASSED	[6%]
test_basic	PASSED	[7%]
test_constructor	PASSED	[8%]
test_initialize	PASSED	[9%]
test_create_proposal	PASSED	[11%]
test_create_proposal_without_payload	PASSED	[12%]
test_create_proposal_not_approved_voting_portal	PASSED	[13%]
test_create_proposal_proposition_power_low	PASSED	[14%]
test_activate_voting	PASSED	[16%]
test_activate_voting_proposal_not_created_state	PASSED	[17%]
test_activate_voting_cooldown_period_not_passed	PASSED	[18%]
test_activate_voting_proposition_power_low	PASSED	[19%]
test_vote_via_portal	PASSED	[20%]
test_vote_via_portal_proposal_not_active	PASSED	[22%]
test_vote_via_portal_many_voting_tokens	PASSED	[23%]
test_queue_proposal_passed_proposal	PASSED	[24%]
test_queue_proposal_failed_proposal_case_1	PASSED	[25%]
test_queue_proposal_failed_proposal_case_2	PASSED	[27%]
test_queue_proposal_invalid_caller	PASSED	[28%]
test_queue_proposal_not_active	PASSED	[29%]
test_execute_proposal	PASSED	[30%]
test_execute_proposal_not_in_queued_state	PASSED	[32%]
test_execute_proposal_cooldown_period_not_passed	PASSED	[33%]
test_execute_proposal_proposition_power_low	PASSED	[34%]
test_cancel_proposal_case_1	PASSED	[35%]
test_cancel_proposal_case_2	PASSED	[37%]
test_cancel_proposal_wrong_proposal_state_case_1	PASSED	[38%]
test_cancel_proposal_wrong_proposal_state_case_2	PASSED	[39%]
test_add_voting_portals	PASSED	[40%]
test_remove_voting_portals	PASSED	[41%]
test_rescue_voting_portal	PASSED	[43%]
test_rescue_voting_portal_voting_count_not_zero	PASSED	[44%]
test_set_power_strategy	PASSED	[45%]
test_set_voting_configs	PASSED	[46%]
test_constants_variable	PASSED	[48%]
test_get_voting_asset_list	PASSED	[49%]
test_get_voting_asset_config	PASSED	[50%]
test_basic	PASSED	[51%]
test_constructor	PASSED	[53%]
test_initialize	PASSED	[54%]
test_updateExecutors	PASSED	[55%]
test_createPayload	PASSED	[56%]
test_createPayload_reverts	PASSED	[58%]
test_receiveCrossChainMessage	PASSED	[59%]
test_receiveCrossChainMessage_expired	PASSED	[60%]
test_receiveCrossChainMessage_payload_created_after_proposal	PASSED	[61%]
test_cancelPayload	PASSED	[62%]
test_executePayload	PASSED	[64%]
test_submitVote_separate	PASSED	[65%]
test_getAccountSlotHash	PASSED	[66%]
test_constructor	PASSED	[67%]
test_receiveCrossChainMessage_create_proposal	PASSED	[69%]
test_receiveCrossChainMessage_bridge_vote	PASSED	[70%]
test_receiveCrossChainMessage_encode_errors	PASSED	[71%]
test_decodeVoteMessage	PASSED	[72%]
test_decodeProposalMessage	PASSED	[74%]
test_decodeMessage	PASSED	[75%]
test_submitVote	PASSED	[76%]

test_submitVote_reverts	PASSED	[77%]
test_submitVoteBySignature	PASSED	[79%]
test_settleVoteFromPortal	PASSED	[80%]
test_closeAndSendVote	PASSED	[81%]
test_constructor	PASSED	[82%]
test_receive_cross_chain_message_delivered_vote_case_1	PASSED	[83%]
test_receive_cross_chain_message_delivered_vote_case_2	PASSED	[85%]
test_receive_cross_chain_message_not_delivered	PASSED	[86%]
test_forward_start_voting_message	PASSED	[87%]
test_forward_start_voting_message_wrong_caller	PASSED	[88%]
test_forward_vote_message	PASSED	[90%]
test_forward_vote_message_voter_already_voted	PASSED	[91%]
test_constructor	PASSED	[92%]
test_getVotingAssetList	PASSED	[93%]
test_getVotingAssetConfig	PASSED	[95%]
test_hasRequiredRoots	PASSED	[96%]
test_getVotingPower_aave	PASSED	[97%]
test_getVotingPower_stkaave	PASSED	[98%]
test_getVotingPower_a_aave	PASSED	[100%]

Appendix B Vulnerability Severity Classification

This security review classifies vulnerabilities based on their potential impact and likelihood of occurrence. The total severity of a vulnerability is derived from these two metrics based on the following matrix.

Impact	High	Medium	High	Critical
	Medium	Low	Medium	High
	Low	Low	Low	Medium
		Low	Medium	High
		Likelihood		

Table 1: Severity Matrix - How the severity of a vulnerability is given based on the *impact* and the *likelihood* of a vulnerability.

References

- [1] Sigma Prime. Solidity Security. Blog, 2018, Available: <https://blog.sigmaprime.io/solidity-security.html>. [Accessed 2018].
- [2] NCC Group. DASP - Top 10. Website, 2018, Available: <http://www.dasp.co/>. [Accessed 2018].

σ'