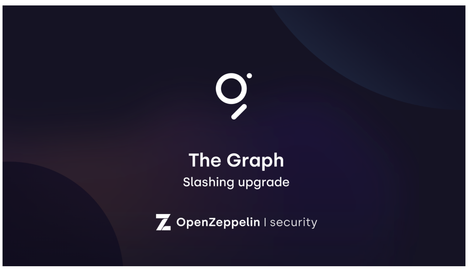
The Graph – Slashing Upgrade Audit

APRIL 27, 2021 | IN SECURITY AUDITS | BY OPENZEPPELIN SECURITY



Introduction

The Graph team asked us to audit an upgrade of the `DisputeManager` contract that adds separated slashing percentages to indexing and query disputes.

The pull request that we have audited is the PR#458 at commit `b61a9b27eb18eab5ec6bb612244d58c33f2321f2` and the audited files are the following:

```
contracts/disputes/DisputeManager.sol
contracts/disputes/DisputeManagerStorage.sol
contracts/disputes/IDisputeManager.sol
```

Overview

Initially the protocol had a single slashing percentage both for query and indexing disputes, but now it gives the possibility to set two independent state variables, namely `qrySlashingPercentage` and `idxSlashingPercentage`, together with a modification to the governance restricted setter function that now accepts two parameters, and a helper function to retrieve the corresponding slashing percentage according to the `DisputeType` enum value passed as input parameter. Moreover, to better reflect the type of the dispute, the `Dispute` struct has been modified to include the type among its parameters.

Apart from this the `getTokensToReward` and `getTokensToSlash` functions have been removed and their logic implemented in the `_slashIndexer` function.

Summary

We understand the purpose of this enhancement and are happy with the small and modular changes. We must also notice that the PR in question is still not merged; we assume that The Graph team will merge it as it is or that no other bugs are introduced in eventual changes. The code has been audited by two auditors over two days, with our findings presented below.

Update: *All of the following issues have been either fixed or acknowledged by the graph team. Our analysis of the mitigations is limited to the specific changes made to cover the issues, and disregards all other unrelated changes in the codebase.*

Critical Severity

None.

High Severity

None.

Medium Severity

None.

Low Severity

[L01] Duplicated code

The helper functions `_pushTokens` and `_pullTokens` are introduced as member functions to the `DisputeManager` contract in this PR. At the same time, the `_pushTokens` and `_pullTokens` are introduced as member functions to the `Staking` contract in PR#457. Aside from minor differences in revert messages, these functions sharing the same name have identical logic.

Assuming both of these PR's are merged without additional changes made, there will be duplicate functions across modules in this codebase. This can cause many problems in terms of the maintainability of the codebase.

Consider consolidating duplicate functions within a common library.

Update: *Fixed in commit `3c684f4079b118d6aa42cbb4ce944a885f50707e`.*

[L02] Constant is not declared explicitly

The `ATTESTATION_SIZE_BYTES` constant is defined in the `DisputeManager` contract to be `161`. Literal values in the codebase without an explained meaning make the code harder to read, understand and maintain, thus hindering the experience of developers, auditors, and external contributors alike.

Developers should define a constant variable for every magic value used (including booleans), giving it a clear and self-explanatory name. Additionally, for complex values, inline comments explaining how they were calculated or why they were chosen are highly recommended.

Consider either defining `ATTESTATION_SIZE_BYTES` to be a sum of intermediate constants or documenting in comments from where it is derived.

Update: *Fixed in commit `d5487d80f9dd099f2970ea98e34ff30813c7693e`.*

[L03] Same event used for multiple variables

The `ParameterUpdated` event of the `Managed` contracts (including the `DisputeManager` contract) is emitted whenever a new value is set for a contract parameter.

In the case of the `_setSlashingPercentage` function, the event is emitted only once two variables are set. Moreover, neither old nor new values are emitted together with the event.

Consider being more explicit in the event string input parameter to reflect that two variables have been set.

Update: *Fixed in commit `7709b944fdbef185a06d48937a099f0b4b19339b`.*

[L04] Unclear lack of input validation

The `initialize` and `setSlashingPercentage` functions of the `DisputeManager` contract are lacking of any input validation for the `_qrySlashingPercentage` and `_idxSlashingPercentage` variables.

At the same time, the comment in line 264 suggests that those slashing percentages are allowed to be zero.

If this is true and there's a dispute that has been created, any call to `acceptDispute` will internally call the `_slashIndexer` and the call will ultimately fail and revert here because of the zero values of the slashing percentages.

This is not a security issue as in this case, to finalize the dispute, the `drawDispute` function should be called instead. However, unexpected failures may confuse the users.

Consider adding a check in the `acceptDispute` function that will revert early and loudly whenever a dispute is trying to be accepted with zero slashing percentages. Moreover, make sure to set these parameters to positive values if disputes are intended to be accepted.

Update: *Fixed in commit `3b38f9e6d82b7cf7169cad111d9c9cf4cdea3326` where documentation was included to clarify that `acceptDispute` will fail under certain conditions and that best course of action is to resolve using `drawDispute` or `rejectDispute`.*

Notes & Additional Information

[N01] Default value used

The `IDisputeManager` interface is using the zero value `DisputeType.IndexingDispute` as meaningful value for the `enum`.

Zero values should never be used to represent meaningful states or values for contract variables, as this can lead to confusion.

Consider adding an initial `Null` state to the `enum` to explicitly set non-zero values to proper `DisputeType`s.

Update: *Fixed in commit `59157c8705c7cda991b6609ede463c6e542d2201`.*

Conclusions

4 Low severity issues and other notes have been reported with recommended changes to improve the codebase.