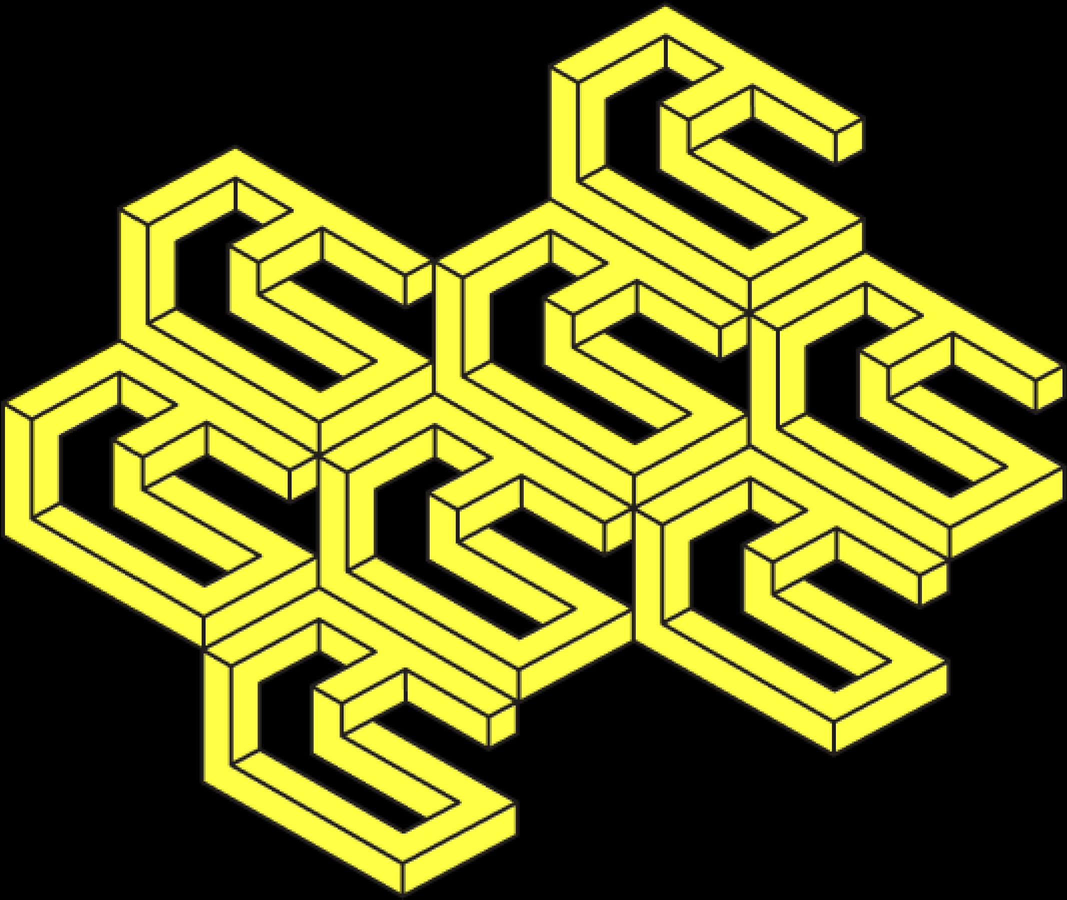

Shape Gasback Smart Contract Review | August 2024

by ChainSafe Systems | August 2024

Table of contents

1. Introduction

Date	Auditor(s)
August 2024	Oleksii Matiiasevych, Tanya Bushenyova

Shape requested **ChainSafe Systems** to perform a review of the gasback smart contracts. The contracts can be identified by the following git commit hash: `8a6ebb3c52ba3a05038ff563a2e92fd892c2467d`

There are 7 contracts in scope.

After the initial review, Shape team applied a number of updates which were reviewed. Several iterations of updates and reviews were performed. The final code can be identified by the following git commit hash: `50d98b1fc1133f09115b98b9622bdbcdcf0156b1`

Defining Severity

Each finding is assigned a severity level.

Note	Notes are informational 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.
Optimization	Optimizations are objective in nature but are not security vulnerabilities. These should be addressed unless there is a clear reason not to.
Minor	Minor issues are objective in nature but are not security vulnerabilities. These should be addressed unless there is a clear reason not to.
Major	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	Critical issues are directly exploitable security vulnerabilities that need to be fixed.

Referencing updated code

Resolved	The finding has been acknowledged and the team has since updated the code.
Rejected	The team dismissed this finding and no changes will be made.

Disclaimer

The review makes no statements or warranties about the utility of the code, safety of the code, suitability of the business model, regulatory regime for the business model, or any other statements about the fitness of the contracts for any specific purpose, or their bug free status.

2. Executive Summary

There are **no** known compiler bugs for the specified compiler version (0.8.20), that might affect the contracts' logic.

There were **0 critical**, **0 major**, **2 minor**, 5 informational/optimization issues identified during the audit process. 1 optimization issue was acknowledged by the team and remained in the final version of the contracts, other issues were not present in the final version. The Shape team were active and engaged in the discussion and improvement process throughout the review.

3. Critical Bugs and Vulnerabilities

No critical issues were identified.

4. Line-by-line review

Gasback.sol

commit 8a6ebb3c52ba3a05038ff563a2e92fd892c2467d

L291 Note Resolved

A function for batch gasback distribution (for an array of smart contract addresses) might be useful.

L294 Minor Resolved

The `distributeGasback()` function allows distribution up to `block.number` including, which might introduce inconsistency if in the same block, after the `distributeGasback()` is called, other transactions would credit gas to the specified smart contract.

commit 1fdb717296915e6ff0f3f1758a0430977a444a0f

L281 Note Resolved

The name of the local variable `totalEarned` might be misleading because it might be confused with `ContractData.totalEarned`.

commit 38c595fce02740b8e557bba3295cf028fb27a973

L253 Optimization Resolved

```
for (uint256 i = 0; i < delegators.length; i++) {
    DelegateData memory delegateData =
_delegatorToDelegateData[delegators[i].delegatorAddress];
    if (delegateData.confirmed) {
        confirmedCount++;
    }
}
```

The data are unnecessarily read from storage while they are already in memory.

L259 Minor Resolved

```
DelegatorData[] memory confirmedDelegators = new DelegatorData[]
(confirmedCount);
for (uint256 i = 0; i < delegators.length; i++) {
    DelegateData memory delegateData =
_delegatorToDelegateData[delegators[i].delegatorAddress];
    if (delegateData.confirmed) {
        confirmedDelegators[i] =
            DelegatorData({delegatorAddress:
delegators[i].delegatorAddress, delegateConfirmed: true});
    }
}
```

The `getConfirmedDelegators()` function will revert if there is an unconfirmed delegator and then confirmed after it, because assigning to `confirmedDelegators[i]` is performed and `i` could be greater than the length of the `confirmedDelegators`.

RecipientValidator.sol

L65-66 Optimization Resolved

```
_activeProviderHashes[providerHash] =
!_activeProviderHashes[providerHash];
emit ProviderStateToggled(providerHash,
_activeProviderHashes[providerHash]);
```

`_activeProviderHashes[providerHash]` is read twice from storage. It could be cached in a local variable.

IGasback.sol

L5-6 Optimization Rejected

```
struct NftData {
    uint256 balanceUpdatedBlock;
    uint256 tokenId;
}
```

The `NftData` struct' `tokenId` field and `balanceUpdatedBlock` field data types could be shrunk to a smallest safe size to make `NftData` fit into a single storage slot.