Quantstamp

Beam Token

Executive Summary

This audit report was prepared by Quantstamp, the leader in blockchain security.

Type	ERC-20
Timeline	2023-10-04 through 2023-10-04
Language	Solidity
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	Beam Token
Source Code	• https://github.com/Merit-Circle/beam-token #f55fa7a
Auditors	• Ibrahim Abouzied Auditing Engineer • Zeeshan Meghji Auditing Engineer

Documentation quality	Low
Test quality	High
Total Findings	5 Fixed: 3 Acknowledged: 2
High severity findings	0
Medium severity findings	0
Low severity findings	3 Fixed: 2 Acknowledged: 1
Undetermined severity findings	1 Acknowledged: 1
Informational findings	1 Fixed: 1

Summary of Findings

`BeamToken` is an extension of the `ERC20Votes` contract that allows an admin to designate Minters and Burners, who can then mint and burn tokens to and from any address. This is coupled with a `Migrator` contract, which will be assigned as a Minter and Burner to a destination contract and a source contract respectively, and facilitate token conversions on the behalf of users. An unrelated `TokenBurner` contract is included, which acts as a token sink by burning its own token balance.

All together, these contracts will allow token holders of MC to convert their tokens to Beam tokens at a rate of 1 MC : 100 Beam Tokens.

Over the course of the audit, we were not able to find any high-severity issues. The contracts and their designs were simple and straightforward. We have left our recommendations for how the contract's security could be tightened even further.

ID	DESCRIPTION	SEVERITY	STATUS
BEAM-1	Tokens Can Be Minted to the Beam Token Contract	Low	Fixed
BEAM-2	Missing Input Validation	Low	Fixed
BEAM-3	Contract Can Be Left without an Admin	Low	Acknowledged
BEAM-4	The `migrationRate` Cannot Be Fractional	Informational	Fixed
BEAM-5	`BURNER_ROLE` Has Unrestricted Burn Allowance	Undetermined	Acknowledged

Assessment Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.

Disclaimer

Only features that are contained within the repositories at the commit hashes specified on the front page of the report are within the scope of the audit and fix review. All features added in future revisions of the code are excluded from consideration in this report.

Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow / underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

Methodology

1. Code review that includes the following
    1. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
    2. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
    3. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Quantstamp describe.
2. Testing and automated analysis that includes the following:
    1. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
    2. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarity, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, and actionable recommendations to help you take steps to secure your smart contracts.

Scope

Files Included

- `contracts/BeamToken.sol`
- `contracts/Migrator.sol`
- `contracts/TokenBurner.sol`

Files Excluded

- `contracts/BeamDAO.sol`

Findings

BEAM-1 Tokens Can Be Minted to the Beam Token Contract

Low Fixed

Update
Marked as "Fixed" by the client. Addressed in: `f92b03a`.

File(s) affected: `BeamToken.sol`

Description: The Beam Token's `_transfer()` function blocks token transfers to `address(this)` such that the contract always has a balance of zero. However, it is possible for the `MINTER_ROLE` to mint tokens directly to the contract.

Recommendation: Validate that `mint()` is not called on `address(this)`.

BEAM-2 Missing Input Validation

Low Fixed

Update
Marked as "Fixed" by the client. Addressed in: `ffae81b`.

File(s) affected: `BeamToken.sol`, `Migrator.sol`, `TokenBurner.sol`

Related Issue(s): SWC-123

Description: It is important to validate inputs, even if they only come from trusted addresses, to avoid human error. A few issues were noticed:

1. The provided documentation states, "The token will have an initial supply of 0 (zero) tokens." However, it is currently possible for the token to be initialized with a non-zero total supply.
2. The `Migrator` is designed to multiply the amount being migrated by a `migrationRate`, with the intended use case being "to migrate at a rate of 1 MC : 100 Beam tokens." However, if the `migrationRate` is mis-initialized to zero, the contract will burn all received tokens without minting any further tokens to the destination contract.
3. Many of the input strings and addresses were unvalidated.

Here is a non-exhaustive list of inputs that should be validated:

- `BeamToken.constructor()`:
  - Validate that `_name` and `_symbol` are non-empty strings.
  - Remove the line `_mint(_msgSender(), _initialSupply);` from the constructor.
- `Migrator.constructor()`:
  - Validate that `_source` and `_destination` are non-zero addresses.
  - Validate that `_migrationRate` is non-zero.
- `TokenBurner.constructor()`: Validate that `_token` is a non-zero address.

Recommendation: We recommend adding the relevant checks.

BEAM-3 Contract Can Be Left without an Admin

Low | Acknowledged

Update
Marked as "Acknowledged" by the client. The client provided the following explanation:

> This is an intended behavior that the team is aware of.

File(s) affected: `BeamToken.sol`

Description: If the admin renounces their role, role management for the `ONLY_MINTER_ROLE` and `ONLY_BURNER_ROLE` will not be possible.

Recommendation: Confirm that this is the intended behavior. If not, override and disable the `renounceRole()` function if the contract only has one admin.

BEAM-4 The `migrationRate` Cannot Be Fractional

Informational | Fixed

Update
Marked as "Fixed" by the client. Addressed in: `6b55ee9`.

File(s) affected: `Migrator.sol`

Description: The `Migrator` supports the ability to migrate tokens at a specified ratio with the `migrationRate`. When the migration is performed, the following destination amount is calculdated as:

```
uint256 destinationAmount = _sourceAmount * migrationRate;
```

This method of calculation enforces that the `destinationAmount >= _sourceAmount`. If it is ever desired that the `destinationAmount` be a fraction of the `_sourceAmount`, it will not be possible.

Recommendation: Add decimals to the `migrationRate` to allow for rates below 1. For example:

```
uint256 destinationAmount = _sourceAmount * migrationRate / DECIMAL_PRECISION; // DECIMAL_PRECISION = 1e18
```

BEAM-5 `BURNER_ROLE` Has Unrestricted Burn Allowance

Undetermined | Acknowledged

Update
Marked as "Acknowledged" by the client. The client provided the following explanation:

This is an intended functionality that the team is aware of.

File(s) affected: `Migrator.sol`

Description: The `BURNER_ROLE` is intended to be used to facilitate the migration between tokens by burning tokens supplied by the user from the source contract. We did not identify any vulnerabilities with the contracts in scope. However, should the `BURNER_ROLE` ever be assigned to an address that is not the `Migrator` contract, it will have unrestricted access to burning any user's tokens.

Recommendation: Confirm if this is the intended functionality. If not, consider updating the burn mechanism to draw from the allowance that the user has approved the contract for.

Definitions

- **High severity** – High-severity issues usually put a large number of users' sensitive information at risk, or are reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
- **Medium severity** – Medium-severity issues tend to put a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or are reasonably likely to lead to moderate financial impact.
- **Low severity** – The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
- **Informational** – The issue does not post an immediate risk, but is relevant to security best practices or Defence in Depth.
- **Undetermined** – The impact of the issue is uncertain.
- **Fixed** – Adjusted program implementation, requirements or constraints to eliminate the risk.
- **Mitigated** – Implemented actions to minimize the impact or likelihood of the risk.
- **Acknowledged** – The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).

Adherence to Best Practices

1. The interface name `IBeamToken` can make it seem as if the `Migrator` will be migrating between different Beam Token contracts. Consider renaming the file to indicate that it can migrate any contracts supporting minting/burning.

Toolset

The notes below outline the setup and steps performed in the process of this audit.

Setup

Tool Setup:

- Slither v0.8.3

Steps taken to run the tools:

1. Install the Slither tool: `pip3 install slither-analyzer`
2. Run Slither from the project directory: `slither .`

Automated Analysis

Slither

All relevant results have been included in the Findings section of the report.

Test Suite Results

The test-suite was executed by running the command `yarn test`. All `15` tests passed.

```
BeamToken
  contructor
```

```
    ✓ Constructor args should be used
    ✓ Should assign DEFAULT_ADMIN_ROLE to deployer
  mint
    ✓ Should work when calling from address which has MINTER_ROLE (43ms)
    ✓ Should revert when called from address without MINTER_ROLE
  burn
    ✓ Should work when calling from address which has BURNER_ROLE (73ms)
    ✓ Should revert when called from address without BURNER_ROLE
  transfer
    ✓ transfer to token contract should fail
    ✓ transfer should work normally (75ms)

Migrator
  contructor
    ✓ Constructor args should be used
  migrate
    ✓ Should work when called by a token owner and Migrator has Minter and Burner role (93ms)
    ✓ Migrating should emit the correct event (128ms)
    ✓ Should revert when called by a token owner without the amount (41ms)
    ✓ Should revert when Migrator contract does not have burner role in source token (99ms)
    ✓ Should revert when Migrator contract does not have minter role in destination token (92ms)

TokenBurner
  ✓ Burn should work (103ms)


  15 passing (2s)
```

Code Coverage

The code coverage results were obtained by running the command `yarn coverage`. The current coverage is `100%` for the audited contracts.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	100	100	100	100	
BeamToken.sol	100	100	100	100	
Migrator.sol	100	100	100	100	
TokenBurner.sol	100	100	100	100	
All files	100	100	100	100	

Changelog

- 2023-10-04 - Initial report
- 2023-10-09 - Fix Review

About Quantstamp

Quantstamp is a global leader in blockchain security. Founded in 2017, Quantstamp's mission is to securely onboard the next billion users to Web3 through its best-in-class Web3 security products and services.

Quantstamp's team consists of cybersecurity experts hailing from globally recognized organizations including Microsoft, AWS, BMW, Meta, and the Ethereum Foundation. Quantstamp engineers hold PhDs or advanced computer science degrees, with decades of combined experience in formal verification, static analysis, blockchain audits, penetration testing, and original leading-edge research.

To date, Quantstamp has performed more than 500 audits and secured over $200 billion in digital asset risk from hackers. Quantstamp has worked with a diverse range of customers, including startups, category leaders and financial institutions. Brands that Quantstamp has worked with include Ethereum 2.0, Binance, Visa, PayPal, Polygon, Avalanche, Curve, Solana, Compound, Lido, MakerDAO, Arbitrum, OpenSea and the World Economic Forum.

Quantstamp's collaborations and partnerships showcase our commitment to world-class research, development and security. We're honored to work with some of the top names in the industry and proud to secure the future of web3.

Notable Collaborations & Customers:

- Blockchains: Ethereum 2.0, Near, Flow, Avalanche, Solana, Cardano, Binance Smart Chain, Hedera Hashgraph, Tezos
- DeFi: Curve, Compound, Maker, Lido, Polygon, Arbitrum, SushiSwap
- NFT: OpenSea, Parallel, Dapper Labs, Decentraland, Sandbox, Axie Infinity, Illuvium, NBA Top Shot, Zora
- Academic institutions: National University of Singapore, MIT

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by Quantstamp; however, Quantstamp does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication or other making available of the report to you by Quantstamp.

Notice of confidentiality

This report, including the content, data, and underlying methodologies, are subject to the confidentiality and feedback provisions in your agreement with Quantstamp. These materials are not to be disclosed, extracted, copied, or distributed except to the extent expressly authorized by Quantstamp.

Links to other websites

You may, through hypertext or other computer links, gain access to web sites operated by persons other than Quantstamp. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites&aspo; owners. You agree that Quantstamp are not responsible for the content or operation of such web sites, and that Quantstamp 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 Quantstamp 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 report. Quantstamp assumes no responsibility for the use of third-party software on any website and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any output generated by such software.

Disclaimer

The review and this report are provided on an as-is, where-is, and as-available basis. To the fullest extent permitted by law, Quantstamp disclaims all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. You agree that your access and/or use of the report and other results of the review, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE. This report is based on the scope of materials and documentation provided for a limited review at the time provided. You acknowledge that Blockchain technology remains under development and is subject to unknown risks and flaws and, as such, the report may not be complete or inclusive of all vulnerabilities. The review is limited to the materials identified in the report and does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. The report does not indicate the endorsement by Quantstamp of any particular project or team, nor guarantee its security, and may not be represented as such. No third party is entitled to rely on the report in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. Quantstamp does not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party, or any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, or any related services and products, any hyperlinked websites, or any other websites or mobile applications, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third party. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.

Quantstamp