Mantle Token and Bridge Audit

OPENZEPPELIN SECURITY | JULY 18, 2023

Security Audits

July 18, 2023

This security assessment was prepared by **OpenZeppelin**.

Table of Contents

Summary

Type
: Token

Timeline
: From 2023-06-06
: To 2023-06-13

Languages
: Solidity

Total Issues
: 16 (10 resolved)

Critical Severity Issues
: 0 (0 resolved)

High Severity Issues
: 0 (0 resolved)

Medium Severity Issues
: 0 (0 resolved)

Low Severity Issues
: 3 (2 resolved)

Notes & Additional Information
: 13 (8 resolved)

Client Reported Issues
: 0 (0 resolved)

Scope

We audited the mantle-token-contracts repository at the `b2016dfb932d85b8b33a9294e8280aa04ca46975` commit and the mantle repository at the `d627d242fe19f50f344f1ff4b27532d1757303a6` commit.

```
contracts
├── L1
│   └── L1MantleToken.sol
└── Migration
    └── MantleTokenMigrator.sol
```

```
packages
└── contracts
    └── contracts
        ├── L1
        │   └── messaging
        │       └── L1StandardBridge.sol
        └── L2
            └── messaging
                └── L2StandardBridge.sol
```

When reviewing the bridge contracts, we assumed the cross-domain messengers work as documented.

System Overview

The Mantle token (MNT) is the gas token of the Mantle network. On the Ethereum network, it is an upgradeable ERC-20 token with mint, burn, permit, and vote extensions. In the default configuration, it allows minting to occur at most once a year and has a configurable mint cap that limits how many tokens can be minted at once, though both parameters can be changed by the owner.

The migrator contract facilitates migration from BIT to MNT by holding MNT tokens and exchanging them for BIT at a specified ratio. It also has the functionality to recover mistakenly sent tokens as well as send BIT and MNT tokens from the contract to a treasury address.

The bridge is a fork of the Optimism bridge contracts, extended to handle the Mantle token. It allows anyone to lock ETH or arbitrary ERC-20 tokens on the Ethereum mainnet in exchange for corresponding ERC-20 tokens on the Mantle network, which can later be destroyed to recover the

replicated on the layer 2 (L2) contract unless it is specifically designed that way.

Security Model and Trust Assumptions

The Mantle token is upgradeable, meaning that the mint cooldown and other constants can be changed as well as the behavior of the token.

The bridge attempts to identify valid token exchanges, but with the exception of ETH and MNT, it trusts the L2 tokens to be able to identify the corresponding L1 token address. Therefore, the onus is on users to validate that they only interact with trustworthy tokens.

Privileged Roles

The owner of the Mantle token can upgrade the contract, mint new tokens, change the mint cap, and transfer the ownership.

The owner of the migrator contract can pause and unpause the contract, change the treasury address, transfer BIT and MNT tokens to the treasury address, and transfer all the other ERC-20 tokens to an arbitrary address.

The Mantle team has claimed that both contracts will be owned by the governance.

Low Severity

Misleading Comments

The following misleading comments were identified:

- The comment describing the `setMintCapNumerator` references a `MintCapNumeratorSet` event, but it should be `MintCapNumeratorChanged`.
- The comment describing the `mint` function says the mint time interval "is initially set to 1 year", which suggests it could be updated. It is actually a constant and can only be changed if the whole contract is upgraded.

Consider updating the comments accordingly.

The `L1StandardBridge` contains a guard condition to prevent it from being reinitialized. However, it assumes the `messenger` will be non-zero after initialization, which is not guaranteed by the `initialize` function. This means it is possible to initialize the other variables and later overwrite them. In practice, the contract will be non-functional until the messenger is set.

Nevertheless, in the interest of predictability, consider ensuring the messenger is non-zero during initialization.

Update: *Resolved in pull request #1027 at commit e641f0e.*

Disable Implementation Contract

The `L1StandardBridge` implementation contract sets the messenger to the zero address, but this doesn't prevent it from being initialized.

In the interest of limiting the attack surface, consider ensuring the implementation contract cannot be initialized. This could be achieved by setting the messenger to an unused non-zero address.

Update: *Acknowledged, not resolved. The Mantle team stated:*

> *We will use our proxy contract to initialize right after deploying the bridge contract, and the proxy contract will be required to be initialized only once. Even if someone tries to initialize the implementation contract afterwards, it will have no impact on our proxy contract.*

Notes & Additional Information

Typographical Errors

In `MantleTokenMigrator.sol` the `received` word is misspelled as `recieved` in several places. Consider resolving this typographical error.

Update: *Resolved in pull request #53 at commit 6b78f54. There have been unrelated changes that removed several instances.*

ETH Handling Can Be Simplified

by default. Consider removing the `receive` and `fallback` functions to simplify the codebase.

Update: *Resolved in pull request #45 at commit c662f74.*

Gas Savings

The `setMintCapNumerator` and `setTreasury` functions can consume less gas by emitting an event first and then changing the storage variable.

For example, the following code snippet

```
uint256 previousMintCapNumerator = mintCapNumerator;
mintCapNumerator = _mintCapNumerator;
emit MintCapNumeratorChanged(msg.sender, previousMintCapNumerator,
```

may be rewritten as:

```
emit MintCapNumeratorChanged(msg.sender, mintCapNumerator, _mintC
mintCapNumerator = _mintCapNumerator;
```

Consider rewriting the `setMintCapNumerator` and `setTreasury` functions to save gas.

Update: *Resolved in pull request #48 at commit 4159ef3.*

Imprecise Docstrings

Some imprecise docstrings have been identified:

- The comment describing the parameter for the `setMintCapNumerator` function does not follow the Ethereum Natural Specification Format (NatSpec) format.
- For consistency with the `migrateAllBIT` description, the `migrateBIT` description should note that the `_amount` must be non-zero.

commit d515706.

Redundant Cast

The `sweepTokens` function of the `MantleTokenMigrator` contract redundantly casts both known token addresses to the `address` type. Consider removing the unnecessary cast operations.

Update: *Resolved in pull request #44 at commit bb921bf.*

Unnecessary Multiple Inheritance

The `L1MantleToken` contract inherits from several contracts with redundant dependencies. This means that some of the contracts are inherited both directly and indirectly. For example, inheriting `ERC20VotesUpgradeable` makes inheriting `ERC20PermitUpgradeable` and `ERC20Upgradeable` redundant.

This is still a reasonable pattern because it improves explicitness and makes the sequence of initializations more intuitive. However, it also forces the `L1MantleToken` to introduce boilerplate functions that are unrelated to the token's new logic. Our opinion is that removing redundancy from the inheritance chain would make the contract simpler and easier to reason about. Consider limiting the inheritance chain to the necessary contracts (i.e., `ERC20BurnableUpgradeable`, `OwnableUpgradeable` and `ERC20VotesUpgradeable`).

Update: *Acknowledged, not resolved. The Mantle team stated:*

> *We prefer not to make modifications to improve explicitness and make the initialization sequence more intuitive.*

BIT Token Address Is Used Instead of MNT Token Address

The bridge contracts have specific logic to handle MNT tokens but the `L1StandardBridge` contract currently associates the BIT token with the L2 Mantle token. Consider reusing the existing variable to identify the MNT token address.

hardcoded instead of reusing the variable.

Unused Function

The `L1StandardBridge` has a function to donate ETH to the contract. However, this function is a holdover from the Optimism code base and is not required for a fresh deployment. Consider removing it.

Update: *Resolved in pull request #1029 at commit a47a661.*

Unusable Data Parameter

All deposits and withdrawals pass an arbitrary data parameter over the bridge. This parameter is emitted in the events on both sides, but is otherwise unused. The documentation claims it is a convenience for external contracts, but it is not passed to the destination and other contracts cannot read the events. Consider clarifying how the parameter could be used, or remove it from the transfer.

Update: *Acknowledged, not resolved. The Mantle team stated:*

> *We see this as a data interface reserved for subsequent cross-chain interoperability.*

Unnecessary Specialization

The bridge contracts contain custom logic to support depositing Mantle tokens. However, only the token mapping validation differs from the generic ERC-20 case, so the `finalizeDeposit` calldata specification can be unified between both branches. Similarly, as long as the `BVM_MANTLE` token is configured correctly Mantle tokens can be withdrawn using the generic ERC-20 logic. This would remove unnecessary withdrawal logic and make the `finalizeMantleWithdrawal` function obsolete. Consider simplifying the code accordingly.

Update: *Acknowledged, not resolved. The Mantle team stated:*

> *We divided this method mainly to facilitate the addition of necessary restrictions for the L2 native token, and also to facilitate subsequent targeted maintenance.*

this check is repeated on the `finalizeERC20Withdrawal` function. Consider removing the redundant validation.

Update: *Resolved in pull request #1031 at commit 5fbb898.*

Incorrect Encapsulation

The `_initiateWithdrawal` function claims to retrieve tokens from the `_from` address but it actually takes them from the caller. Similarly, the event references the caller instead of the `_from` address. This behavior is equivalent in the current code base because both calling functions pass the message sender as the `_from` address. Nevertheless, in the interest of predictability and encapsulation, consider using the `_from` address inside the function.

Update: *Acknowledged, not resolved. The Mantle team stated:*

> *We will consider this in future Mantle Network upgrades.*

Use of Hardcoded Values

The `L1StandardBridge` contract has the address of the MNT token hardcoded. Consider creating a constant variable and using it instead of the hardcoded address for clarity and readability.

Similarly, the `L2StandardBridge` contract hardcodes the `IL2StandardERC20` identifier. Consider using the more expressive `type(IL2StandardERC20).interfaceId` statement instead.

Update: *Acknowledged, not resolved. The Mantle team stated:*

> *We will consider this in future Mantle Network upgrades. For now, we have decided to hardcode this address as we do not foresee the MNT token address changing in the future as the MNT token itself is upgradable.*

Conclusions

Below we provide our recommendations for monitoring important activities in order to detect and prevent potential bugs.

Monitoring Recommendations

While audits help in identifying code-level issues in the current implementation and potentially the code deployed in production, the Mantle team is encouraged to consider incorporating monitoring activities in the production environment. Ongoing monitoring of deployed contracts helps identify potential threats and issues affecting production environments. With the goal of providing a complete security assessment, the monitoring recommendations section raises several actions addressing trust assumptions and out-of-scope components that can benefit from on-chain monitoring.

Governance

There are multiple privileged actions with serious security implications:

- The `L1MantleToken` contract owner can :
  - Change the mint cap, within some bounds.
  - Mint tokens within the mint cap.
  - Upgrade the contract arbitrarily, potentially removing the mint cap restrictions.
- The `MantleTokenMigrator` contract owner can:
  - Halt or unhalt the contract.
  - Change the treasury address that can receive the BIT and MNT tokens.
  - Send the BIT or MNT tokens to the treasury address.
  - Retrieve any other tokens sent to the contract.
- The `L1StandardBridge` contract owner can upgrade the contract, which gives the owner access to all funds secured by the bridge.

Consider monitoring these administrator functions to ensure all changes are expected.

Financial activity

Consider monitoring the token transfers over the bridge to identify:

- Transfers that take unusually long to complete
- Transactions that revert
- Any deposits that invoke the refund mechanism
- Any unusually large, small or frequent transactions
- Transfers with mismatched L1 and L2 tokens

These may indicate a user interface bug, an ongoing attack or other unexpected edge cases.

Related Posts

Beefy Zap Audit

BeefyZapRouter serves as a versatile intermediary designed to execute users' orders through routes...

Security Audits

OpenBrush Contracts Library Security Review

OpenBrush is an open-source smart contract library written in the Rust programming language and the...

Security Audits

Linea Bridge Audit

Linea is a ZK-rollup deployed on top of Ethereum. It is designed to be EVM-compatible and aims to...

Security Audits