Wrapped Elon Security Review

Pashov Audit Group

Conducted by: pashov

December 10th, 2023

Contents

1. About pashov

Krum Pashov, or **pashov**, is an independent smart contract security researcher. Having found numerous security vulnerabilities in various protocols, he does his best to contribute to the blockchain ecosystem and its protocols by putting time and effort into security research & reviews. Check his previous work here or reach out on Twitter @pashovkrum.

2. Disclaimer

A smart contract security review can never verify the complete absence of vulnerabilities. This is a time, resource and expertise bound effort where I try to find as many vulnerabilities as possible. I can not guarantee 100% security after the review or even if the review will find any problems with your smart contracts. Subsequent security reviews, bug bounty programs and on-chain monitoring are strongly recommended.

3. Introduction

A time-boxed security review of the **Wrapped Elon** was done by **pashov**, with a focus on the security aspects of the application's smart contracts implementation.

4. About Wrapped Elon

The protocols allows holders of Dogelon Mars ERC20 token to wrap them so that they are transformed to a wrapped version of the token that works better with bridging to other chains, because for example Solana doesn't work well with high decimals tokens. Bridging the wrapped version with smaller decimals resolves the integration problem.

5. Risk Classification

Severity	Impact: High	Impact: Medium	Impact: Low
Likelihood: High	Critical	High	Medium
Likelihood: Medium	High	Medium	Low
Likelihood: Low	Medium	Low	Low

5.1. Impact

- High - leads to a significant material loss of assets in the protocol or significantly harms a group of users.
- Medium - only a small amount of funds can be lost (such as leakage of value) or a core functionality of the protocol is affected.
- Low - can lead to any kind of unexpected behavior with some of the protocol's functionalities that's not so critical.

5.2. Likelihood

- High - attack path is possible with reasonable assumptions that mimic on-chain conditions, and the cost of the attack is relatively low compared to the amount of funds that can be stolen or lost.
- Medium - only a conditionally incentivized attack vector, but still relatively likely.
- Low - has too many or too unlikely assumptions or requires a significant stake by the attacker with little or no incentive.

5.3. Action required for severity levels

- Critical - Must fix as soon as possible (if already deployed)
- High - Must fix (before deployment if not already deployed)
- Medium - Should fix
- Low - Could fix

6. Security Assessment Summary

***review commit hash* - c4fff0af9d6a2eec5258ec870f5314001c48c026**

***fixes review commit hash* - a04916b3cbf1719438857701dcace126c67bb9bd**

7. Executive Summary

Over the course of the security review, pashov engaged with Wrapped Elon to review Wrapped Elon. In this period of time a total of **4** issues were uncovered.

Protocol Summary

Protocol Name	Wrapped Elon
Date	December 10th, 2023

Findings Count

Severity	**Amount**
Medium	1
Low	3
Total Findings	**4**

Summary of Findings

ID	Title	Severity	Status
[M-01]	Centralization attack vector is present in setEnabledState	Medium	Acknowledged
[L-01]	Disabling unwrapping will block all bridges at the same time	Low	Acknowledged
[L-02]	Wrapped token name is the unwrapped token name	Low	Acknowledged
[L-03]	Protocol is using a vulnerable library version	Low	Resolved

8. Findings

8.1. Medium Findings

[M-01] Centralization attack vector is present in `setEnabledState`

Severity

Impact: High, as an owner can block unwrapping of wrapped assets

Likelihood: Low, as it requires a malicious or a compromised owner

Description

The `setEnabledState` method of `WrappedElon` allows the owner of the contract to disable (or enable) wrapping and unwrapping of tokens. The issue is that a malicious or a compromised owner can decide to act in a bad way towards users and block unwrapping of the tokens, essentially locking them out of their funds. If the ownership is burned then (or private keys are lost) it will be irreversible.

Recommendations

Potential mitigations here are to use governance or a multi-sig as the contract owner. Even better is to use a Timelock contract that allows users to be notified prior to enabling/disabling wrapping/unwrapping so that they can take action, although this removes the benefit of using the method as a risk mitigation for bridge attacks.

8.2. Low Findings

[L-01] Disabling unwrapping will block all bridges at the same time

The `wrapEnabled` and `unwrapEnabled` variables are added as a mitigation mechanism against flawed bridges (that have potentially infinite mint vulnerability). The problem is that if this wrapped token is used by or integrated with multiple bridges, setting `unwrapEnabled` to `false` will block all bridges at the same time, even if just one of them is faulty. Consider switching to a mechanism that can handle multiple bridges integrations in a fault-tolerant way.

[L-02] Wrapped token name is the unwrapped token name

The `WrappedElon` contract serves as a wrapper for `$ELON` tokens. Here is how its constructor looks like:

```
constructor() ERC20("Dogelon", "ELON") {}
```

The problem is that instead of naming the token with the same name, prepended with the "Wrapped" word, it is using the same name as the unwrapped version. This can lead to confusions, especially if a liquidity pool is created with the wrapped token for some reason. Change the constructor in the following way:

```diff
-constructor() ERC20("Dogelon", "ELON") {}
+constructor() ERC20("WrappedDogelon", "WELON") {}
```

[L-03] Protocol is using a vulnerable library version

In `package.json` file in the repository we can see this:

```
"@openzeppelin/contracts": "^4.7.3",
```

This version contains multiple vulnerabilities as you can see here. While the problems are not present in the current codebase, it is strongly advised to upgrade the version to v4.9.5 which has fixes for all of the vulnerabilities found so far after v4.7.3.