ParagonCoin Token Contract: Review

Mikhail Vladimirov and Dmitry Khovratovich

8th September, 2017

This document describes issues found in ParagonCoin during code review performed by ABDK Consulting.

1. Introduction

We were asked to audit the ParagonCoin Token Contract. The contract is contained in gistfile1.txt.

We got no additional documentation on the contract except for the whitepaper.

2. Paragon Token

In this section we describe issues related to the token contract defined in gistfile1.txt.

2.1 Documentation Issues

This section lists documentation issues found in the token smart contract.

1. It actually reverts, not throws in case of overflow (line 11, line 39).
2. It actually reverts, not throws in case of underflow (line 25).
3. Number of tokens hardcoded into smart contract (line 274).

2.2 Suboptimal Code

This section lists suboptimal code patterns found in token smart contract.

1. Code in lines 334-341 could be simplified by using `` `AbstractToken.transfer` `` method and `` `burnTokens` `` method internally.
2. Code in lines 353-357 could be simplified. Calculation of x/2 rounded up may be done by:
   - (x + 1) / 2;
   - x - x / 2;
   - (x - 1) / 2 + 1;
   - x / 2 + x % 2 & etc.

   Besides, there is no need to actually round up. Instead:

```
uint256 fundFee = halfFundFee(feeTotal);
uint256 burnFee = safeSub(feeTotal, fundFee)
```

will be effective:

```
uint256 burnFee = feeTotal / 2;
uint256 fundFee = safeSub (feeTotal, burnFee);
```

It gives the same result.

3. There is no reason to override method and just call overridden method inside (line 371). This is equivalent to not override method at all.

2.3 Suspicious Behaviour

This section lists issues of token smart contract related to suspicious behaviour.

1. The `transferFrom` method does not charge fee (line 371).

2.4 Major Flaws

This section lists major flaws found in the token smart contract.

1. In line 20 should be `<=`. Otherwise it does not work for `x+y=MAX_UINT256`.
2. In line 34 should be `>=`. Otherwise it does not work for `x=y`.
3. In line 49 should be `<=`. Otherwise it does not work for `x*y=MAX_UINT256`.
4. Fee depends on the owner's balance, but it does not depend on who actually transfers the tokens, who the tokens are transferred to and how many tokens are being transferred (line 349). The whitepaper seems to imply that the transaction fee depends on the total supply[1], but it is not fully clear.

2.5 Other Issues

This section lists stylistic and other minor issues found in the token smart contract.

1. Instead `Transfer (msg.sender, _to, _value)` should be logged `Transfer (msg.sender, fund, fundFee)` (line 177).
2. Instead `_value` should be `valueWithFee` at line 174.

3. Security provisions

We recommend the smart contract designers to claim explicitly security provisions in addition to the intended functionality of the contract. Such provisions should be non-trivial falsifiable claims, i.e. they should declare certain property of the contract for which an attack can be demonstrated if implemented incorrectly. The security provisions serve not only for the purpose of confidence of future users in the contract's behaviour, but also as a platform for a potential bug bounty program. The contract authors are encouraged to offer various bounties binded to some provision being violated.

[1] In this case the owner might not be needed.

Here we provide a list of security provisions which are appropriate for these contracts and would be expected by ordinary users.

Name	Description	Current status
ParagonCoin		
Permanent supply decrease	The total number of tokens decreases with every transaction	FALSE
Safe multiple approvals	A token holder can approve another address with some tokens only if the previous allowance has been fully spent.	TRUE

4. Our Recommendations

Based on our findings, we recommend the following:

1. Fix overflow issues.
2. Fix underflow issues.
3. Change the code comment (line 274).
4. Refactor the code to remove suboptimal parts.
5. Fix issues related to suspicious behaviour.
6. Fix the `safeAdd` function in line 20.
7. Fix the `safeSub` function in line 34.
8. Fix the `safeMul` function in line 49.
9. Fix the `fee` function in line 349.