SmartDec

BCShop Functionality Verification

This report is public.

Published: February 10, 2018

Abstract

In this report we consider the BCShop project. Our task is to verify functionality of the smart contracts.

Procedure

We perform our analysis according to the following procedure:

- manual verification
  - we check smart contracts logic and compare it with the declared one
  - we verify that smart contracts deployed to the Ethereum mainnet have same parameters as declared
- report
  - we reflect all the gathered information in the report

Disclaimer

This report does not give any warranties on the security of the code. We always recommend proceeding to several independent audits and a public bug bounty program to ensure the security of the smart contracts. Besides, this report is not an investment advice.

Project Overview

In our analysis we consider:

- BCShop smart contracts code
  - CustomTrancheWallet (commit 0473cef)
  - TokenTrancheWallet (commit 07194fa)
  - TrancheWallet (commit 07194fa)
- tests
  - customTrancheWallet.js (commit 0473cef)
  - trancheWallet.js (commit 20802db)
- description

Project Architecture

The project is a Truffle project.
The files under consideration are two token wallets: TokenTrancheWallet.sol (32 LoC), CustomTrancheWallet.sol (118 LoC).

- CustomTranchWallets
  - Owned.sol (24 LoC)
  - IERC20Token.sol (25 LoC).
- TokenTrancheWallet is inherited from
  - IOwned.sol (7 LoC)
  - TrancheWallet.sol (97 LoC).

The project includes tests.
Compilation of the project fails (however, the contracts that are being audited can be compiled).

Verified Functionality

The wallets store a standard ERC20 token with the following functionality: the owner can define the locking parameters and then lock funds in the contract. After that, in the defined periods of time the defined amount of token will become available for the withdraw by the beneficiary. The owner can change the beneficiary after the funds have been locked, but can not change the parameters of the token locking.

Hereby we verify the following description of the contracts functionality:

“**`CustomTrancheWallet`** contract is intended for token storage and unlocking in accordance with the schedule. The schedule is defined in the constructor and can be changed before the locking function is called with the `setParams` function.
The schedule is set by the two arrays:

- `unlockDates` is an ordered array of unlock dates in the standard Ethereum `timestamp` format (for example 1517443200 is 01 Feb 2018 00:00:00 GMT)
- `unlockAmounts` is an ordered array; each element of `unlockAmounts` is equal to the total amount of tokens available for the withdraw after the corresponding unlock date

For example, if total amount of locked tokens is 100 and we want them to be unlocked in four stages:

- the first stage: 10 tokens unlocked
- the second stage: the rest of the tokens unlocked; 25 tokens available
- the third stage: the rest of the tokens unlocked; 55 tokens available

- the fourth stage: the rest of the tokens (i.e. 45 tokens) unlocked; 100 tokens available

then `unlockAmounts` must be set as [10,25,55,100]. The last element of the array must be equal to the total amount of locked tokens.
Workflow:

1. Create the contract with the desired schedule, stored token and beneficiary.
2. Transfer tokens to the contract.
3. Call `lock` function to lock the tokens.
4. After the tokens are locked, one can check the number of unlocked tokens with `getAvailableAmount` and get all available tokens with `sendToBeneficiary`.

The contract keeps count of unlocked and transferred tokens using `alreadyWithdrawn` field.
After the tokens are transferred to the contract but before the `lock` call it is possible to withdraw all the tokens from the contract.
After the tokens are locked it is possible to change the beneficiary but it is impossible to change lock parameters.

This contract is already deployed to the real network and verified on etherscan.io:

1. Airdrop wallet
ETH address: 0x94Fa60ecD8071597672e937698575aC84bD52b79
Parameters:
On 10 Apr 2018 00:00 GMT 250,000 BCS tokens are unlocked
On 01 Jun 2018 00:00 GMT 1,750,000 BCS tokens are unlocked
The total token number is: 2,000,000

2. Bancor wallet
ETH address: 0xb7804329cab71b5d96cA608A975C7436b23d4DE5
On 10 Apr 2018 00:00 GMT 20,000 BCS tokens are unlocked
On 01 May 2018 00:00 GMT 170,000 BCS tokens are unlocked
The total token number is: 190,000

3. Advisors wallet
ETH address: 0x7F920F9e34573C09B257Ae6e2620aF164158c04e
On 01 May 2018 00:00 GMT 325,000 BCS tokens are unlocked
The total token number is: 325,000

4. Partners wallet
ETH address: 0x2B1Be4591238E0E46401F432a5C6500D804041Bb
On 01 May 2018 00:00 GMT 1,200,000 BCS tokens are unlocked
The total token number is: 1,200,000

5. Long wallet
ETH address: 0xe72017D34F72547793dF611418B228930dA7A7FC
On 01 Jan 2019 00:00 GMT 470,000 BCS tokens are unlocked
On 01 Jan 2020 00:00 GMT 940,000 BCS tokens are unlocked (1,410,000 in total)
On 01 Jan 2021 00:00 GMT 1,410,000 BCS tokens are unlocked (2,820,000 in total)
On 01 Jan 2022 00:00 GMT 1,622,000 BCS tokens are unlocked
The total token number is: 4,442,000

`TokenTrancheWallet` contract is intended for token storage and unlocking in equal portions-tranches. Unlocking periods also have same durations. The initial schedule is set

by the two parameters:

- `tranchePeriodInDays` — time between tranches in days
- `trancheAmountPct` — the size of one tranche in percents of the initial amount of locked tokens

Workflow:

1. Create the contract with the desired tranches parameters, stored token and beneficiary.
2. Transfer tokens to the contract.
3. Call `lock` function to lock the tokens, the number of days for all the tokens to be unlocked should be passed as a parameter. The number of locked tokens is stored in `initialFunds` field.
4. After the tokens are locked, one can check the number of unlocked tokens with `amountAvailableToWithdraw` (which returns the number of available tokens as the first parameter and how many tranches is it as a second parameter) and get all the available tokens with `sendToBeneficiary`.

The contract keeps count of unlocked and transferred tokens using `tranchesSent` field.
After the tokens are transferred to the contract but before the `lock` call it is possible to withdraw all the tokens from the contract.
After the tokens are locked it is possible to change the beneficiary but it is impossible to change lock parameters.

This contract is already deployed to the real network and verified on etherscan.io at the address https://etherscan.io/address/0x700620683311f2d150974b4F262BACFcf7Ff5d8D.
Parameters: 1,500,000 BCS tokens were locked on 11 Jan 2018 11:14:41 GMT for 8 years, every 30 days starting from the first day 1% of the total amount is unlocked (i.e. 1 tranche is available at the time of the audit)”

Conclusion

In this report we have considered the BCShop smart contracts. We performed our analysis according to the procedure described above.
We verify that the functionality of the contracts corresponds to the one described in Verified Functionality section.

This analysis was performed by SmartDec.

COO Sergey Pavlin 

February 10, 2018