WALLIE Smart Contract

Mikhail Vladimirov[1] and Dmitry Khovratovich[2]

20th October 2018

This document describes the audit of the Wallie smart contract performed by ABDK Consulting.

1. Introduction

We've been asked to review the Wallie smart contract as deployed to the Ethereum mainnet. We were explicitly asked not to consider optimization issues and code redundancy. We considered comments in the code and website as a source of business requirements to the contract.

The Wallie contract is a simple investment-return scheme, where everyone can make an investment to the smart contract and get daily rewards from this and everyone else' money.

We were asked to search for backdoors and the ability of administrator or any other user to wipe out the contract balance or affect the contract execution in an adversarial way. We have not found any such opportunity.

2. Wallie Contract

The main issue we have found (Line 112) is that the user is unable to withdraw his rewards partially, so if the accumulated reward exceeds the contract balance, the investor can not get even a portion back. To mitigate that, he would have to send his own money to the contract and get the rewards back in the same transaction. However, this would not work for the administrator and referral fees, so the users who claim these funds can be frontrunned by other users if they use this mitigation.

Other issues are listed below.

---

[1] Smart Contract Lead
[2] Security and Cryptography Lead, CEO.

2.1 Documentation Issues

This section lists the inconsistency between the program code and comments that do not affect the money distribution

1. Line 34: To get a profit it is not required to sent a zero value; any transaction with value greater than 0.01 ETH would also trigger reward distribution.
2. Line 42: `charged` should be read as `credited`.

2.2 Distribution Issues

This section lists issues with reward distribution, which were found in the smart contract.

1. Lines 25-29: The distribution chart may not be consistent with actual contract behavior: the contract indeed creates liabilities to the administrator of 15% of the total investment, but the dividends to users do not form the remaining 85% (as the website mentions). Instead, the total liabilities can take 100% of the investment or even more as long as the contract has sufficient balance.
2. Lines 26, 29: the smart contract does not make a distinction between advertising and developer support distribution. Instead, both fractions go to the administrator.
3. Line 43: the RefBack bonus applies to the first investment from this address only.
4. Administrator can invest with a referral and return the investment faster than a regular investor due to administrator fees. Based on 13% referral+cashback return rate for normal users and 28%=13%+15% for administrator, we estimate that without a referral an investment is returned in full after 69 days, with a referral in 60 days, and for an administrator with a referral - in 50 days. Therefore an administrator with a referral gets rewards from the money invested before him if he invests within 19 days after the user without an referral, and within 10 days - if with a referral. To ensure fairness, an administrator should be unable to invest within 19 days of the last investment.

2.3 Statistics Issues

This section lists issues related to the statistics gathered by the smart contract.

1. Line 122: the variable `all_adm_payments` does not include cash backs to the administrator and referrals.
2. Line 129: the variable `all_ref_payments` does not include cash backs to the administrator but includes investment by the administrator.
3. Line 136: the variable `all_cash_back_payments` includes administrator's investments.

2.4 Arithmetic Overflow Issues

Despite including SafeMath library, there is a lot of arithmetic statements in the code not protected from overflows: lines 146, 156, 163, 177, 182. We have not found any way to exploit them though because they all are derived from msg.value with not so high multiplicators.

2.5 Referral Issues

This section lists issues related to the referral and cashback functionality.

1. Line 122: the condition `msg.sender != ref_addr` can be bypassed by investing from different addresses and cross-referencing them.

2.6 Other Issues

Line 301: the output of the address conversion method is not well defined when the `source` is not a 20-byte array, and may change over time.