NaumovLab DEVELOPMENT

NAUMOVLAB.COM
SMART-CONTRACTS.RU

NaumovLab auditors

Telegram: @radnek
Skype: mezone

Smart Contract TRONFARM

https://tronscan.org/#/contract/TKTnsntt9vz9CzHYg7dwtspK2UJqXBotSD/code
https://github.com/whalesFinance/contracts/blob/master/contracts/TRONFARM.sol

The code was checked for the bytecode:

When compiling the code with the compiler version 0.4.25 with the optimization enabled, the resulting bytecode matches the bytecode of the deployed smart contract.

!IMPORTANT: The TRONFARM contract interacts with the WHALES token smart contract located at: 0x51920f822760DD05663dd0c52FD5F3581DF673C8 (THQWi4gRDrmtCPqZArE8abuz8yjeaBZDFJ). An audit of its code is also required.

During the audit, it was assumed that the token code matches the code located at: https://github.com/whalesFinance/contracts/blob/master/contracts/WHALES.sol (bytecode not matched)

During the audit, critical errors were discovered, one of which can be used as a backdoor to withdraw funds from users by the project admins.

Error description:

1. If the user wants to withdraw funds from the contract, he calls the "**leave()**" method

When this method is called, the funds previously blocked by him on the contract and the tokens accrued to him are transferred to the user's balance.

If the balance of tokens in the contract is less than necessary to transfer the amount of tokens, the transaction falls with the message "REVERT opcode executed. Message: SafeMath sub error". Thus, the user cannot withdraw his money from the contract until the contract is replenished with the required number of tokens.

Since the contract contains the “**finish()**” function, which allows admins to pick up the contract balance 14 days after the start of its work, the admins will be able to withdraw TRX that users cannot pick up to their wallet.

In addition, by manipulating the "**reward()**" function, admins can charge users large amounts in tokens in order to block their ability to withdraw funds from the contract.

2. If the user made a deposit into the system, then took it and re-entered it, then due to an error in the **join()** function, namely in the lines:

```
if (farmer.deposited == 0) {
  farmerAddresses.push (msg.sender);
}
```

   The user's address is duplicated in the **farmerAddresses** array. Because of this, when the **reward()** function is called, the user will receive a double reward in tokens. The user can repeat this operation many times - increasing his reward any number of times.

<u>Notes:</u>

1. Using a loop

```
for (uint i = 0; i <farmerAddresses.length; i ++)
```

with a large number of users may lead to blocking of execution of the "**reward()**" function.

2. Bonus tokens are credited by calling the "**reward()**" function by the admins, while the logic implies that this should be done once every 1 minute. This method of calculating remuneration is a bad practice, since the accrual of bonus tokens depends entirely on the actions of the project administrators, and the accuracy of the calculation depends on the settings of external software and delays in the network.

<u>Conclusions:</u>

The contract contains critical errors that require correction. The main logic of the contract is implemented incorrectly, we recommend using a different method of calculating bonus rewards, which does not depend on the actions of the administrator and the time of calling the "**reward ()**" function.

NaumovLab auditors