GVISP 1
your space

2018.11.20.

TokensMarketplace.COM

#CryptoChristmasGiveaway

support@tokensmarketplace.com

GVISP1 LTD

office@gvisp.com

CHRISTMAS_ PRESENTS.SOL

SMART CONTRACT AUDIT

GVISP1 LTD.

GVISP1 Ltd. **** Your space**** Bul.Vojvode Misica 17, 11000 Belgrade, Serbia, EU

1. Audit description

One contract was checked: **ChristmasPresents.sol.**
The purpose of this audit is to check all functionalities of ChristmasPresents.sol contract and to determine the level of security and probability of adverse outcomes.

ChristmasPresents.sol is a smart-contract that enables giveaway competition on Ethereum blockhain. Competitors send their ETH in order to compete and contract sends them a package of random amount of ERC20 tokens as a reward. They also get their unique coupon ID which makes them eligible for one of the three main prizes.

After compentition ends, on 24th of December 2018 at 00:00 (CET), the contract generates three random IDs that correspond to competitors' addresses and chooses the winners.
Total amount of fee equals to 10% of collected ETH, third winner gets 10% of the rest, second gets 30% and the main winner gets 60% of ETH.

2. Quick review

- ✓ All functions and state variables are well commented which is good in order to understand quickly how everything is supposed to work. Other than the constructor, no function has any loop, which is cost-effective.
- ✓ The contract was written in accordance with solidity's aesthetic standards (names of state variables and functions start with lowercase letter, names of structures start with capital letter, etc)
- ✓ During deployment, an array of addresses that will have admin authority must be provided together with the timestamp of giveaway end and the list of ERC20 token addresses and their maximum amount available for automatic gifts.
- ✓ All orders are sorted within specific mappings and arrays are not used. It would have been more simple to code coupons into an array, but if arrays had been used, each time someone calls some functions a certain loop would have been activated which would be highly cost ineffective.

3. A brief review of contract's functionalities

Contract's functions are:

- ✓ `changeAvailableTokensAndAmounts` - can be called only by admins and is used to allow specified token to be an award. Parameters are: 1. list of tokens' addresses, 2. list of amount for each token.
- ✓ `collectFee` - can be called only by admins and is used to transfer fees from the contract to caller's address. There are no parameters for this function.

- ✓ `executeLoterry` - can be called only by admins. Draws the winners of the loterry and sends them awards in ETH. There are no parameters for this function.
- ✓ `fallback` - triggered when someone sends ETH to contract's address. Can be called by anyone and registers the sender for the loterry, and sends him automatically the gift of random amount of ERC20 tokens.

Functions only authorized addresses can call:

- ✓ `executeLoterry`
- ✓ `collectFee`
- ✓ `changeAvailableTokensAndAmounts`

4. Functionalities test

After deployment

- ➢ `fallback` : ✔
- ➢ `random` : ✔
- ➢ `changeAvailableTokensAndAmounts` : ✔
- ➢ `collectFee` : ✔
- ➢ `executeLottery` : ✔

5. Detailed code check (line-by-line)

- ✓ There are no state variables (other than mappings).

Mappings:

(address => bool) admins – remembers if an address has admin authorization
(address => uint) maxAmountsOfTokens – remembers the maximum amount of each token to be given as an award.
(address => uint[]) tickets – remembers the unique IDs for each ticket that belong to a specific address.
(uint => address) contestants – maps each ticket to a specific contestant's address.

There are no modifiers.

During deployment, one must specify an array of addresses that will be authorized to call `collectFee`, `changeAvailableTokensAndAmounts` and `executeLoterry` functions. These addresses are immutable which is good for safety reasons. Also, the timestamp of giveaway end must be specified together with the list of available tokens and maximum amount for each of them.

Functions:

- `**changeAvailableTokensAndAmounts**` - can be called only by admins and is used to allow specified token to be an award. Parameters are: 1. list of tokens' addresses, 2. list of amounts for each token.
- `**collectFee**` - can be called only by admins and is used to transfer fees from the contract to caller's address. There are no parameters for this function.
- `**executeLoterry**` - can be called only by admins. Draws the winners of the loterry and sends them awards in ETH. There are no parameters for this function.
- `**fallback**` - triggered when someone sends ETH to contract's address. Can be called by anyone and registers the sender for the loterry, and sends him automatically the gift of random amount of ERC20 tokens.
- `**random**` - generates a random integer within a given interval (internal function).

6. Static analysis test, vulnerabilities and outcomes

✓ Static analysis of the code was conducted and no security flaws were found.

✓ Over and underflows are not possible.

https://oyente.melon.fund

- ✓ *browser/ChristmasPresents.sol:ChristmasPresents*
  EVM Code Coverage : 54%
- ✓ *Callstack Depth Attack Vulnerability : False*
- ✓ *Re-Entrancy Vulnerability : False*
- ✓ *Assertion Failure : False*
- ✓ *Parity Multisig Bug 2 : False*
- ✓ *Transaction-Ordering Dependence (TOD) : False*

7. Final comments

The code is well organized and commented. Structure is cost-effective and there is no evident possibility of adverse outcomes.