MOBU

SMART CONTRACT AUDIT

Date: June 10th, 2019

0xblock

London, UK

E-mail: contact@0xblock.uk

Table of Contents

June 10, 2019

0xblock

Disclaimer

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only.

Contract Reviewed

https://github.com/mobuadmin/mobu/blob/master/MobuToken.sol

Filename	Git Hash
MobuToken.sol	1db3b83f497abd3981c3656b43e1bf5b9ed71345

Contract Deployment Address

https://etherscan.io/token/0x1cd2a911a28a034da2645fea802e280253c7f527

Audit Summary

Analysis	Description	Result
Design Patterns	Inspect the overall structure of the smart contract	Passed
Static Analysis	Performed using a series of automated tools	Passed
Manual Analysis	Performing a hands-on code review of the smart contract to identify common vulnerabilities	Passed
Network Behavior	GAS optimization and smart contract interaction checks	Passed

Token Allocation

- **Symbol**: MOBU
- **Total Supply**: 150,000,000
  - Token Sale (120,000,000)
    - Total 80% Tokens minted to 0xf637ba8fe861aaef1b9f8b45b9e0b040af15e018
  - Team (18,000,000)
    - Total 12% of Team Tokens locked up for 365 days
    - The Team address 0x1d3a6fc3f6ce8d6e6469c2bd4354759d50175220 will be able to claim these tokens after lock-up period expired
  - Advisors (6,000,000)
    - Total 6% of Advisors Tokens locked up for 90 days
    - The Advisors address 0xd5778cb3844b530eaf9f115af9f295e378a1b449 will be able to claim these tokens after lock-up period expired
  - Airdrop / Bounty (6,000,000)
    - Total 6% of Airdrop / Bounty Tokens locked up for 30 days
    - The Bounty address 0x148595db00a4aa94a1b05f8cd1aa6d9b4fdffc21 will be able to claim these tokens after lock-up period expired

Known Attacks

Known Attack	Result / Observation
Reentrancy on a Single Function	Passed
Cross-function Reentrancy	Passed
Timestamp Dependence	Passed
Integer Overflow and Underflow	Passed
Forcibly Sending Ether to a Contract	Passed
Parity Multisig Bug 2	Passed
Call Stack Depth Attack	Passed
Transaction-Ordering Dependence	Passed
Proxy and Buffer Overflow	Passed
Denial Of Service (DoS)	Passed.

June 10, 2019

0xblock

Audit Assessment Summary

Here is our assessment and recommendations, in order of importance.

Critical severity
No critical severity issues were found.

High severity
No high severity issues were found.

Medium severity
No medium severity issues were found.

Low severity
While performing static analysis on Remix, compiler complaints about low severity warnings in open-zeppelin contracts and unlimited GAS warnings due to loops but we have checked contract code thoroughly and confirmed there were no such loops as 95% of code derived from the open-zeppelin standard code.

Line by Line Code Review

Line 1: Solidity version: Specified stable Solidity Compiler 0.5.8 without ^ which is usually a good practice.
Status: Everything is OK

Line 3-65: Library SafeMath corresponds to OpenZeppelin implementation
https://github.com/OpenZeppelin/openzeppelin-solidity/blob/master/contracts/math/SafeMath.sol
Status: Everything is OK

Line 67-137: Library Ownable corresponds to OpenZeppelin implementation
https://github.com/OpenZeppelin/openzeppelin-solidity/blob/master/contracts/ownership/Ownable.sol
Status: Everything is OK

Line 139-171: Library Claimable corresponds to OpenZeppelin implementation
https://github.com/ConsenSys/real-estate-standards/blob/master/node_modules/openzeppelin-solidity/contracts/ownership/Claimable.sol
Status: Everything is OK

Line 174-194: ERC20 Interface corresponds to OpenZeppelin implementation
https://github.com/OpenZeppelin/openzeppelin-solidity/blob/master/contracts/token/ERC20/IERC20.sol
Status: Everything is OK

Line 196-380: Standard ERC20 token Implementation corresponds to OpenZeppelin implementation https://github.com/OpenZeppelin/openzeppelin-solidity/blob/master/contracts/token/ERC20/ERC20.sol
Status: Everything is OK

Line 382-403: Burnable Token corresponds to OpenZeppelin implementation https://github.com/OpenZeppelin/openzeppelin-solidity/blob/master/contracts/token/ERC20/ERC20Burnable.sol
Status: Everything is OK

Line 404-482: Mobu Token Implementation - See MOBU Token Class Code Review for details
Status: Everything is OK

MOBU Token Class Code Review

```
/*
// 0xblock comments
// Public Boolean flags holds the information whether team, advisor, or bounty tokens has been claimed or not
// This make sure that none of them will claim tokens more than once.
*/
    // Flags to check if locked tokens have been already transferred
    bool teamTokensClaimed = false;
    bool advisorsTokensClaimed = false;
    bool bountyTokensClaimed = false;

/*
// 0xblock comments
// Modifier to check unlocking period has been started or not
// This is a good example of Solidity modifier usage for common checks which required in multiple functions
*/
    modifier unlockingPeriodStarted() {
        require (beginUnlockDate != 0);
        _;
    }

/*
// 0xblock comments
// A function to claim Advisors tokens
// This function can be executed only if following conditions met
// 1. Owner has started the unlocking period
// 2. Unlocking period has been expired which is after 90 days since unlocking period was begun in this case
// 3. Advisors tokens have not been claimed already
// If all conditions met then this function transfers 4% (6m) MOBU tokens held by Smart contract
// to Advisor address which is 0xd5778cb3844b530eaf9f115af9f295e378a1b449 in this case
*/
    // 4% of Advisors tokens can be claimed after 3 months
    function claimAdvisorTokens() public unlockingPeriodStarted {
        require (now > beginUnlockDate + 90 days);
        require (!advisorsTokensClaimed);
        advisorsTokensClaimed = true;
        _transfer(address(this), advisorsAddress, 6000000e18);
    }

/*
// 0xblock comments
// This function allows owner to recover any ERC20 tokens sent to contract address by mistake
// This is a good example not to lose Ethers and other ERC20 tokens sent to the smart contract
*/
    // Owner can recover any ERC-20 tokens sent to contract address.
    function recover(ERC20 _token) public onlyOwner {
        _token.transfer(msg.sender, _token.balanceOf(address(this)));
    }
}
```

```
/*
// 0xblock comments
// Mobu token is Standard and Claimable
// This is good example of using Burnable and Ownership transfer support
*/
contract Mobu is ERC20Burnable, Claimable {

/*
// 0xblock comments
// Mobu token and symbol name is "MOBU"
// Mobu token decimals are 18
*/
    string public constant name     = "MOBU";
    string public constant symbol   = "MOBU";
    uint8 public constant decimals = 18;

/*
// 0xblock comments
// Mobu token initial supply is 150m
*/
    uint256 constant initialSupply = 150000000e18;

/*
// 0xblock comments
// Public address variables set during contract deployment
// These addresses specify where to mint tokens during contract deployment
*/
    // Token holders
    address public mainHolderAddress;
    address public teamAddress;
    address public advisorsAddress;
    address public bountyAddress;

/*
// 0xblock comments
// Public Boolean flag to hold information about whether Token unlocking period started or not
// This flag has been used so unlocking period will be started counting only after Owner executed startUnlockingPeriod function
*/
    // Start unlocking period for locked tokens
    uint public beginUnlockDate = 0;



/*
// 0xblock comments
// Constructor was called with below addresses
// _mainHolderAddress: 0xf637ba8fe861aaef1b9f8b45b9e0b040af15e018
// _teamAddress: 0x1d3a6fc3f6ce8d6e6469c2bd4354759d50175220
// _advisorsAddress: 0xd5778cb3844b530eaf9f115af9f295e378a1b449
// _bountyAddress: 0x148595db00a4aa94a1b05f8cd1aa6d9b4fdffc21
// 80% (120m) MOBU tokens were minted to Main holder address immediately
// 20% (30m) MOBU tokens were minted to Smart contract address and these tokens hold by smart contract until unlock period expires
*/
    constructor(address _mainHolderAddress, address _teamAddress, address _advisorsAddress, address _bountyAddress) public {
        mainHolderAddress = _mainHolderAddress;
        teamAddress = _teamAddress;
        advisorsAddress = _advisorsAddress;
        bountyAddress = _bountyAddress;

        // Main MOBU token holder address has 80% of MOBU Tokens
        _mint(_mainHolderAddress, 120000000e18);

        // Total 12% Tokens Lock-up schedule
        // Airdrop/bounty: 4% locked-up for 1 month
        // Team: 12% locked-up for 1 year.
        // Advisors: 4% locked-up for 3 months
        // These tokens will be held by the smart contract until the locked-up period
        _mint(address(this), 30000000e18);
    }

/*
// 0xblock comments
// A function to start unlocking period can be invoked only by the owner of the contract
// This functions sets unlocking boolean flag if not set already which is good way to implement function.
*/
    // Owner can start unlocking period
    function startUnlockingPeriod() public onlyOwner {
        require (beginUnlockDate == 0);
        beginUnlockDate = now;
    }
```

```
/*
// 0xblock comments
// A function to claim Bounty tokens
// This function can be executed only if following conditions met
// 1. Owner has started the unlocking period
// 2. Unlocking period has been expired which is after 30 days since unlocking period was begun in this case
// 3. Bounty tokens have not been claimed already
// If all conditions met then this function transfers 4% (6m) MOBU tokens held by Smart contract
// to Bounty address which is 0x148595db00a4aa94a1b05f8cd1aa6d9b4fdffc21 in this case
*/
    // 4% Airdrop/bounty tokens can be claimed after 1 month
    function claimBountyTokens() public unlockingPeriodStarted {
        require (now > beginUnlockDate + 30 days);
        require (!bountyTokensClaimed);
        bountyTokensClaimed = true;
        _transfer(address(this), bountyAddress, 6000000e18);
    }

/*
// 0xblock comments
// A function to claim Team tokens
// This function can be executed only if following conditions met
// 1. Owner has started the unlocking period
// 2. Unlocking period has been expired which is after 365 days since unlocking period was begun in this case
// 3. Team tokens have not been claimed already
// If all conditions met then this function transfers 12% (18m) MOBU tokens held by Smart contract
// to Team address which is 0x1d3a6fc3f6ce8d6e6469c2bd4354759d50175220 in this case
*/
    // 12% of Team tokens can be claimed after 12 months
    function claimTeamTokens() public unlockingPeriodStarted {
        require (now > beginUnlockDate + 365 days);
        require (!teamTokensClaimed);
        teamTokensClaimed = true;
        _transfer(address(this), teamAddress, 18000000e18);
    }
```

Conclusion

The reviewed MOBU smart contract is free of security issues and well crafted. Most of base contracts are derived from the Open Zeppelin standard contracts.