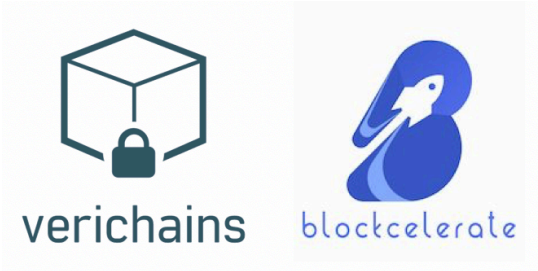

SECURITY AUDIT OF DREP SMART CONTRACTS

REPORT

JULY 06, 2018

✓erichains Lab

info@verichains.io

https://www.verichains.io

Driving Technology >> Forward

EXECUTIVE SUMMARY

This Security Audit Report prepared by Verichains Lab on July 06, 2018. We would like to thank DREP Foundation to trust Verichains Lab to audit smart contracts. Delivering high-quality audits is always our top priority.

This audit focused on identifying security flaws in code and the design of the smart contracts. The scope of the audit is limited to the source code files provided to Verichains Lab from DREP's github repositories. Verichains Lab completed the assessment using manual, static, and dynamic analysis techniques.

The assessment identified no security issues in DREP smart contracts code. However, we recommend DREP to use OpenZeppelin, the industry-standard library to make it more readable and secure for DREP's token and crowdsale contracts. Some extra modules and features from OpenZepplin are recommeded to be used such as IndividuallyCappedCrowdsale, CappedCrowdsale, TimedCrowdsale, Pausable, Ownable and SafeMath.

CONTENTS

ACRONYMS AND ABBREVIATIONS

Ethereum	An open source platform based on blockchain technology to create and distribute smart contracts and decentralized applications.
ETH (Ether)	A cryptocurrency whose blockchain is generated by the Ethereum platform. Ether is used for payment of transactions and computing services in the Ethereum network.
Smart contract	A computer protocol intended to digitally facilitate, verify or enforce the negotiation or performance of a contract.
Solidity	A contract-oriented, high-level language for implementing smart contracts for the Ethereum platform.
Solc	A compiler for Solidity.
EVM	Ethereum Virtual Machine.

AUDIT OVERVIEW

ABOUT DREP

DREP Foundation aims to provide a decentralised reputation ecosystem that empowers existing Internet platforms to unleash its reputation value and achieve the interconnectedness of reputation data in the Internet community. DREP Foundation provides a decentralized reputation ecosystem comprised of a public chain, a reputation-based protocol and the tools for Internet platforms (DRApps) to leverage it all.

DREP（Decentralized Reputation System）is a decentralized solution based on blockchain technology that aims to quantify and tokenize the value of reputation for a variety of online commerce, trading, investment and data sharing purposes. DREP aims to empower internet platforms to solve their pain points, restructure their value ecosystem and facilitate their transition and acceleration via reputation-centered tokenomics and blockchain technology

The website of DREP is at https://www.drep.org/
White paper (EN – version 0.3.4) is at https://s3-ap-southeast-1.amazonaws.com/drep-singapore/DREP-WhitePaperv1.0.5.pdf

About DREP Token

Token Name	DREP
Symbol	DREP
Decimals	18
Total Tokens	10 Billion
Token Distribution	
Team	1.5 Billion (15%)
Community Development, Branding and Treasury	1.5 Billion (15%)
Token Presale and Crowdsale	3 Billion (30%)
User Acquisition and Strategic Partnerships	4 Billion (40%)

SCOPE OF THE AUDIT

This audit focused on identifying security flaws in code and the design of the smart contracts for the upcoming Crowdsale and Token.

It was conducted on commit `9ab74be` of branch `master` from drep-project/crowdsale and commit *c0e331e* of branch `master` from drep-project/ ERC20Token GitHub repositories.

Repository URL:
https://github.com/drep-project/ERC20Token
https://github.com/drep-project/crowdsale

The scope of the audit is limited to the following 2 source code files:

Source File	SHA256 Hash
crowdsale.sol	`fc6d54f1767377618199c95036d0aa3968dbeb4649a8659d287a0267abd26cf9`
drep_token.sol	`7673907e1f53887f3f2a50d76dfd81c7a5ba2a2bd2c2ef8a7d7c770fe01eee28`

Note that parameters for crowdsale contracts including start and end time, hard cap, personal cap... will be passed to the contract by the contract owner any time so verifying the correctness of initialized parameters is not possible at the time of this audit.

AUDIT METHODOLOGY

Our security audit process for smart contract includes two steps:

- Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using public and in-house automated analysis tools.
- Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that was considered during the audit of the smart contract:

- Integer Overflow and Underflow
- TimeStamp Dependence
- Race Conditions
- Transaction-Ordering Dependence
- DoS with (Unexpected) revert
- Dos with Block Gas Limit
- Gas Usage, Gas Limit and Loops
- Redundant fallback function
- Unsafe type Inference
- Reentrancy
- Explicit visibility of functions state variables (external, internal, private and public)
- Logic Flaws

For vulnerabilities, we categorize the findings into categories, depending on their criticality:

LOW	An issue that does not have a significant impact, can be considered as less important
MEDIUM	A vulnerability that could affect the desired outcome of executing the contract with medium impact in a specific scenario; needs to be fixed.
HIGH	A vulnerability that could affect the desired outcome of executing the contract with high impact; needs to be fixed with high priority.
CRITICAL	A vulnerability that can disrupt the contract functioning; creates a critical risk to the contract; required to be fixed immediately.

AUDIT RESULT

VULNERABILITIES FINDINGS / RECOMMENDATIONS

- The assessment identified no security issues in DREP smart contracts code.
- We recommend DREP to use OpenZeppelin, the industry-standard library to make it more readable, modular and secure for DREP's token and crowdsale contracts. Some extra modules and features from OpenZepplin are recommeded to be used are IndividuallyCappedCrowdsale, CappedCrowdsale, TimedCrowdsale, Pausable, Ownable and SafeMath.
- Many important parameters of the crowdsale rounds, such start and stop times, rates, caps are modifiable after the sale has started. We recommend to set these parameters during the creation of crowdsale contract.
- We recommend DREP to use separated contract instances for each sale round instead of combining all the sale rounds into one single contract to allow flexibilites and avoid any potential issues due to the messed up of initilized sales parameters.
- Token burn (function burn in line #77) does not emit any event, all balance changes should be able to tracked by emitting an event.
- The "*emit* " keyword is only supported from compiler version 0.4.21, please change the contract code pragma version accordingly.
- DREP should add documents for all modifiers and functions to ensure its correct usage.

CONCLUSION

DREP crowdsale and token smart contracts have been audited by Verichains Lab using various public and in-house analysis tools and intensively manual code review. The assessment identified no issues in DREP smart contracts code. However, we recommend DREP to use OpenZeppelin, the industry-standard library to make it more readable, modular and secure for DREP's token and crowdsale contracts..

LIMITATIONS

Note that initialized parameters for crowdsale contracts including start and end time, hard cap, personal cap... will be passed to the contract when these contracts are created so verifying the correctness of initialized parameters is not possible at the time of this audit.

Security auditing cannot uncover all existing vulnerabilities, and even an audit in which no vulnerabilities are found is not a guarantee for a 100% secure smart contract. However, auditing allows discovering vulnerabilities that were unobserved, overlooked during development and areas where additional security measures are necessary.

APPENDIX I - CALL FLOWS

Figure 1 Call graph of crowdsale.sol