**Executive Summary**

Quantstamp conducted a formal verification audit of the Lockdrop contract, a time-lock smart contract for locking Ether. The audit process involved the execution of static analysis tools such as Slither and Mythril, as well as the coverage tool solidity-coverage. The audit report highlights the adequacy and coverage of the provided contracts, along with the severities of issues highlighted by the tools.

**Conclusion**

Quantstamp endorses the content on that web site or the operator or operations of that site. You are solely responsible for determining the extent by such software.

**Timeliness of content**

The code is reasonable commented.

**Steps taken to run the tools:**

1. Installed the solidity-coverage tool (within the project's root directory):
   - `npm install solidity-coverage`
2. Compiled the code:
   - `solc -o artifacts --standard --combined-abi Contract.sol`
3. Created a project root:
   - `mkdir Lockdrop`
4. Added solidity-coverage dependencies:
   - `npm install solidity-coverage`
5. Export the project root:
   - `cd Lockdrop`
6. Added ganache-chainer dependencies:
   - `npm install ganache-chainer`
7. Installed ganache-chainer:
   - `npm install ganache-chainer`
8. Installed truffle:
   - `npm install truffle`
9. Installed hardhat:
   - `npm install hardhat`
10. Compiled the project:
   - `truffle compile`
11. Deployed the contract:
   - `hardhat run scripts/01_deploy.js`
12. Exported the project root:
   - `cd Lockdrop`
13. Deployed the contract:
   - `hardhat run scripts/02_deploy.js`
14. Created a new instance of
   - `hardhat run scripts/03_deploy.js`

**Audit**

A Denial-of-Service (DoS) attack is a situation which an attacker renders a smart contract unusable. The factory contract time-lock smart contracts for locking Ether. Quantstamp, however, it is clear that the implementation is minimalistic and easy time-lock smart contracts for locking Ether. smart contracts aim to implement factory of contracts.

The purpose of the factory contract is to lock Ether for a set duration. It is important to note that the contract only allows calls from the owner or a user who has been whitelisted. The contract also ensures that the duration is greater than zero and that the user has enough balance to lock the Ether. The contract checks these conditions before proceeding with the locking operation.

**Recommendation**

The test coverage appears good, however, it misses the branch where equality is reached.

---

**Denial of Service, unsuccessful transfer of Ether, infinite lock of funds.**

- **Possible issues we looked for included (but are not limited to):**
  - Incorrect values in the call function.
  - Missing checks for the duration of the lock.
  - Lack of a mechanism to unlock the contract.

---

**Severity:**

- **High:**
  - The implementation is minimalistic and easy to exploit.
  - The contract lacks robust checks for the duration of the lock.
  - The contract does not have a mechanism to unlock the contract.

---

**Recommended Changes:**

- **Add missing checks for the duration of the lock.**
- **Implement a mechanism to unlock the contract.**

---

**Recommendations**

1. **Add missing checks for the duration of the lock.**
2. **Implement a mechanism to unlock the contract.**
3. **Add additional security features.**
4. **Investigate the use of additional security practices.**
5. **Consider using additional security tools.**
6. **Review the implementation of the contract.**

---

**Executive Summary**

Quantstamp's dedication to research and development in the form of collaborations with leading academic institutions such as National University of Singapore and National University of Ireland, Galway, has led to the development of a robust and secure smart contract solution. Quantstamp's team boasts decades of combined experience in formal verification, static analysis, and software verification. Collectively, our individuals help boost adoption of this exponentially growing technology.

Quantstamp is a Y Combinator-backed company that helps to secure smart contracts at scale using computer-aided reasoning tools, with a mission to help companies unlock the power of smart contracts. Our team uses the latest advances in formal verification and static analysis to identify and mitigate risks in smart contracts. We provide a comprehensive audit report that includes a detailed analysis of the smart contract, as well as recommendations for improving security and compliance.

**Changelog**

- **March 17th 2020**
  - Added a new section on the importance of security in smart contracts.
  - Updated the list of recommended changes to reflect recent developments.

---

**Source Code**

- **Lock.sol**
- **Lockdrop.sol**
- **Contract.sol**

---

**References**

1. Martin Derka
2. Stake Technologies Lockdrop

---

**License**

This document is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License. You may share and adapt this document, as long as you give appropriate credit, provide a link to the license, and indicate if changes were made. You must share your adaptation under the same license as the original.

---

**Contact Us**

For more information or to schedule a meeting about our services, please contact us at contact@quantstamp.com. We look forward to hearing from you.