HackSafe

Smart Contract Security Audit Report

Krill

August 2022

Security Status

★★★★½

www.hacksafe.io

Audit Details

Audited project
Krill

Deployer address
0x14026129cee6321810d15e1b277313104da5c309

Client contacts
Krill

Blockchain
Polygon

Website
Not provided by the owner

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and TechRate and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (HackSafe) owe no duty of care towards you or any other person, nor does HackSafe make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and HackSafe hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, HackSafe hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against HackSafe, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

Procedure

Step 1 - In-Depth Manual Review

Manual line-by-line code reviews to ensure the logic behind each function is sound and safe from various attack vectors. This is the most important and lengthy portion of the audit process (as automated tools often cannot find the nuances that lead to exploits such as flash loan attacks).

Step 2 - Automated Testing

Simulation of a variety of interactions with your Smart Contract on a test blockchain leveraging a combination of automated test tools and manual testing to determine if any security vulnerabilities exist.

Step 3 – Leadership Review

The engineers assigned to the audit will schedule meetings with our leadership team to review the contracts, any comments or findings, and ask questions to further apply adversarial thinking to discuss less common attack vectors.

Step 4 - Resolution of Issues

Consulting with the team to provide our recommendations to ensure the code's security and optimize its gas efficiency, if possible. We assist project team's in resolving any outstanding issues or implementing our recommendations.

Step 5 - Published Audit Report

Boiling down results and findings into an easy-to-read report tailored to the project. Our audit reports highlight resolved issues and any risks that exist to the project or its users, along with any remaining suggested remediation measures. Diagrams are included at the end of each report to help users understand the interactions which occur within the project.

Background

HackSafe was commissioned by Krill to perform an audit of smart contract:

- https://polygonscan.com/address/0x05089c9ebffa4f0aca269e32056b1b36b37ed71b#code

The purpose of the audit was to achieve the

- Ensutre that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Contract Details

Token contract details for 02.08.2022

Token Type	: ERC20
Contract name	: Krill
Contract address	: 0x05089C9EBFFa4F0AcA269e32056b1b36B37ED71b
Compiler version	: v0.7.4+commit.3f05b770
Total supply	: 13,045,790.464168
Token Ticker	: Krill
Decimals	: 18
Token Holders	: 6,961
Top 100 token holder's	: 96.96%
Transactions count	: 4,623,171
Contract deployer address	: 0x14026129cee6321810d15e1b277313104da5c309
Owner address	: 0x34bc3d36845d8a7ca6964261fbd28737d0d6510f

Social profiles

Twitter Profile	: https://twitter.com/polywhalefi
Github Profile	: https://github.com/polywhalefarm/contracts
Telegram Profile	: https://t.me/polywhalefi
Coingecko profile	: https://www.coingecko.com/en/coins/polywhale/

Audit Summary

According to the standard audit assessment, Customer`s solidity smart contracts are “Secure”. This token contract does contain owner control, which do not make it fully decentralized as owner does have control over smart contract.

Insecure	Poor	Secure	Well-secured

You are here 

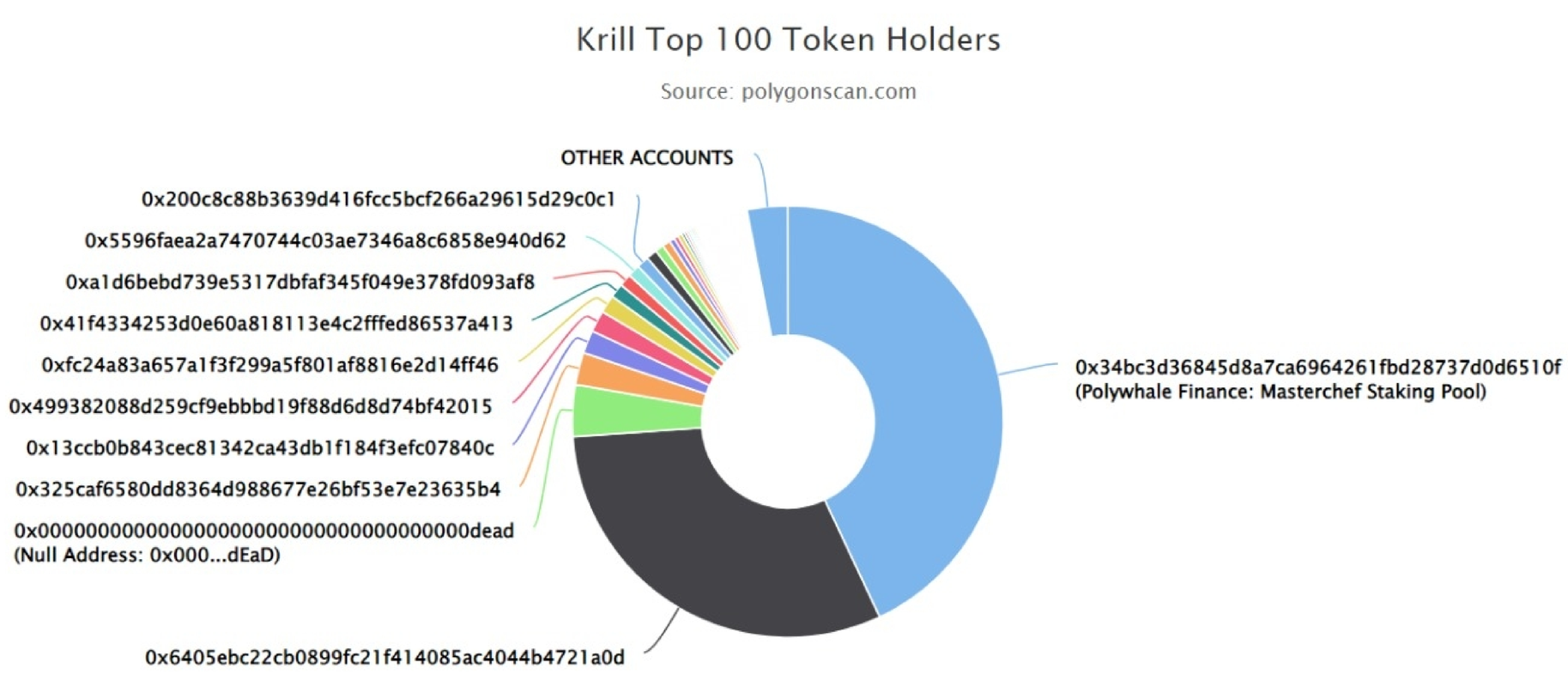
We used various tools like Slither, Mythril and Remix IDE. At the same time this finding is based on critical analysis of the manual audit. All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the issues checking status.

We found 0 critical, 0 high, 0 medium and 2 low and some very low-level issues. These issues are not critical ones.

Krill Token Distribution

The top 100 holders collectively own 96.96% (12,649,322.26 Tokens) of Krill

Token Total Supply: 13,045,790.46 Token | Total Token Holders: 6,961



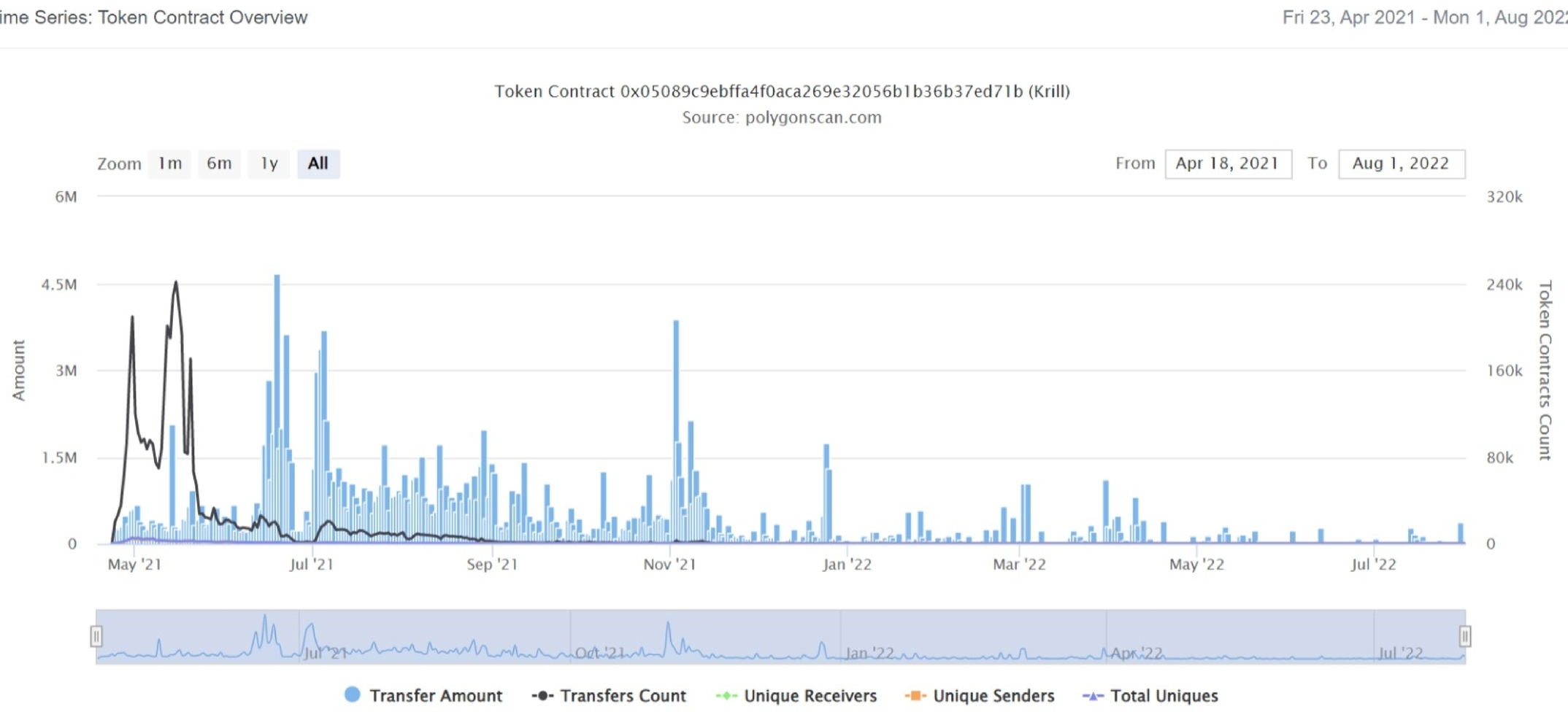
Krill Token Top 20 Token Holders

(A total of 12,649,322.26 tokens held by the top 100 accounts from the total supply of 13,045,790.46 token)

Rank	Address	Quantity (Token)	Percentage
1	Polywhale Finance: Masterchef Staking Pool	5,611,605.814394598878980986	43.0147%
2	0x6405ebc22cb0899fc21f414085ac4044b4721a0d	4,027,454.453367327653854148	30.8717%
3	Null Address: 0x000...dEaD	507,923.584206003238006822	3.8934%
4	0x325caf6580dd8364d988677e26bf53e7e23635b4	312,334.610072412442906234	2.3941%
5	0x13ccb0b843cec81342ca43db1f184f3efc07840c	225,145.328398223700476791	1.7258%
6	0x499382088d259cf9ebbbd19f88d6d8d74bf42015	210,909.051128897073936299	1.6167%
7	0xfc24a83a657a1f3f299a5f801af8816e2d14ff46	174,938.240150351942087427	1.3410%
8	0x41f4334253d0e60a818113e4c2fffed86537a413	140,195.779232391527456745	1.0746%
9	0xa1d6bebd739e5317dbfaf345f049e378fd093af8	121,959.008590330513970873	0.9349%
10	0x5596faea2a7470744c03ae7346a8c6858e940d62	120,991.721845059835864533	0.9274%
11	0x200c8c88b3639d416fcc5bcf266a29615d29c0c1	119,023.999432603504970433	0.9124%
12	0x8b74e30b7824c424bcff3a3ea5fbbaa329e0fc4c	106,367.110940058169676088	0.8153%
13	0x312951c60050e90cf698ea4e499befe359eddbad	82,692.125388372183621444	0.6339%
14	0x5bc2a0f35ee957ca76bc269a93ad9d2900d15850	80,067.54750594449811428	0.6137%
15	0xe711c3612267c072151303678f25b4fd1fc667ae	50,362.321250003475793558	0.3860%
16	0x56874c1cddcc3ed944d1caa0aac9b1f03764c815	42,532.199800452453750511	0.3260%
17	0x58abdc25b330edac20e8190851fa6e47147f9bce	41,549.92512937032747342	0.3185%
18	0x43aaab66308ebc549d49db0e6e3dceeafdb39966	29,183.914829292327522861	0.2237%
19	0x4276f21f9de6f6cccaa5c42d35d8252b5780326d	24,952.247114197784095278	0.1913%
20	0xb5f9ee89c4fd4f01cb4ce5245602d15527241f4f	24,374.409699970235975736	0.1868%

Krill Token Distribution

Krill Token Contract Overview



Contract functions details

+ Context
    -[Int] _msgSender
    -[Int] _msgData

+ [Int] IERC20
    -[Ext] totalSupply
    -[Ext] balanceOf
    -[Ext] transfer
    -[Ext] allowance
    -[Ext] approve
    -[Ext] transferFrom

+[Lib] SafeMath
    -[Int] tryAdd
    -[Int] trySub
    -[Int] tryMul
    -[Int] tryDiv
    -[Int] tryMod
    -[Int] add
    -[Int] sub
    -[Int] mul
    -[Int] div
    -[Int] mod
    -[Int] sub
    -[Int] div
    -[Int] mod

+ ERC20 (Context, IERC20)
    - [Pub] < constructor>
    -[Pub] name
    -[Pub] symbol
    -[Pub] decimals
    -[Pub] totalSupply
    -[Pub] balanceOf
    -[Pub] transfer #
    -[Pub] allowance
    -[Pub] approve #
    -[Pub] transferFrom #
    -[Pub] increaseAllowance
    -[Pub] decreaseAllowance

Contract functions details

-[Int] _transfer #
-[Int] _mint#
-[Int] _burn #
-[Int] _approve #
-[Int] _setupDecimals #
-[Int] _beforeTokenTransfer #

+ Ownable (Context)
- [Int] <constructor>
-[Pub] owner
-[Pub] renounceOwnership #
-modifiers: onlyOwner
-[Pub] transferOwnership
-modifiers: onlyOwner

+ Krill (ERC20, Ownable)
-[Pub] mint #
-modifiers: onlyOwner

($) = payable function
= non-constant function

Issues Checking Status

No.	Title	Status
1.	Unlocked Compiler Version	Low issue
2.	Missing Input Validation	Passed
3.	Race conditions and Reentrancy. Cross-function race conditions.	Passed
4.	Possible delays in data delivery	Passed
5.	Oracle calls.	Passed
6.	Timestamp dependence.	Passed
7.	Integer Overflow and Underflow	Passed
8.	DoS with Revert.	Passed
9.	DoS with block gas limit.	Passed
10.	Methods execution permissions.	Passed
11.	Economy model of the contract.	Passed
12.	Private use data leaks.	Passed
13.	Malicious Event log.	Passed
14.	Scoping and Declarations.	Passed
15.	Uninitialized storage pointers.	Passed
16.	Arithmetic accuracy.	Passed
17.	Design Logic.	Passed
18.	Safe Open Zeppelin contracts implementation and usage.	Passed
19.	Incorrect Naming State Variable	Passed
20.	Too old version	Low issue

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that can't have a significant impact on execution.

Security Issues

Critical Severity Issues

No critical severity issue found.

High Severity Issues

No high severity issue found.

Medium Severity Issues

No medium severity issues found.

Low Severity Issues

Two low severity issue found..

1. Unlocked Compiler Version.

- **Description**

  The contract utilizes an unlocked compiler version. An unlocked compiler version in the contract's source code permits the user to compile it at or above a particular version. This, in turn, leads to differences in the generated bytecode between compilations due to differing compiler version numbers. This can lead to ambiguity when debugging as compiler-specific bugs may occur in the codebase that would be difficult to identify over a span of multiple compiler versions rather than a specific one.

- **Recommendation**

  It is advisable that the compiler version is alternatively locked at the lowest version possible so that the contract can be compiled. For example, for version >=0.6.0 <0.8.0 the contract should contain the following line:

  pragma solidity 0.7.4;

2. Too old compiler version.

- **Description**

  Contract has been deployed using too old compiler version.

- **Recommendation**

  It is advisable that the compiler version of solidity should be among the new compiler versions.

Centralization

Owner privileges:

- Krill Contract:
    - Owner can remove and transfer ownership.
    - Owner can mint new tokens.

This smart contract has some functions which can be executed by the Admin (Owner) only. If the admin wallet private key would be compromised, then it would create trouble but smart contract ownership has been renounced. Following are Admin functions functions:

- Transferownership
- Renounceownership
- Mint

Conclusion

Smart contract contains low severity issues! The further transfer and operations with the fund raised are not related to this particular contract.

HackSafe note: Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by Owner.