10K Swap

Audit Report

Mon Sep 25 2023

✉ contact@scalebit.xyz

https://twitter.com/scalebit_

ScaleBit

10K Swap Audit Report

1 Executive Summary

1.1 Project Information

Description	An AMM that advances with Ethereum, deployed on Starknet Mainnet
Type	Dex
Auditors	ScaleBit
Timeline	Fri Aug 25 2023 - Mon Sep 25 2023
Languages	Cairo
Platform	Starknet
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/10k-swap/10k_swap-contracts
Commits	c47f2e158687a757d0834893d5b01ba4131a55ec

1.2 Files in Scope

The following are the SHA1 hashes of the original reviewed files.

ID	File	SHA-1 Hash
L0KF	contracts/l0k_factory.cairo	52747c9276d0ed3bc24a13da77606f262370832c
L0KR	contracts/l0k_router.cairo	fb9cb1fbccf4e26ea7fdd14fc6c23360e14f858e
U1X1	contracts/libraries/uq112x112.cairo	5fefc1f6ff1d91bb7c19453e71671d1ffcca3f7b
L0KL	contracts/libraries/l0k_library.cairo	3395348fd97e67d7e098dcad0e172edeb94ee456
L0KE2	contracts/l0k_erc20.cairo	137fffdd11d14dec62c98a0b6f194284088b05bb
L0KP	contracts/l0k_pair.cairo	d8a597d0eee70e2fc0f4cdd7959a1c5dd5bf1f03
ADD	contracts/warplib/maths/add.cairo	fcbd2dc2b747a33725308eb30703f9a44365b3b3
NEQ	contracts/warplib/maths/neq.cairo	d1c9b92a6e03733464c9e99cca690ca7f3bfeca5
GT	contracts/warplib/maths/gt.cairo	895a692160fdff1d3948cb0447691db3d6eabbab
MUL	contracts/warplib/maths/mul.cairo	9e8cad803a207b18b02595259944f9dadf8e40e0
CWMUC	contracts/warplib/maths/utils.cairo	21f756f79c9aa877fe07ba8870400a130441a963
GE	contracts/warplib/maths/ge.cairo	54590d10bc4d16a9e3c9d95578a90c5b17adb66e

ICO	contracts/warplib/maths/int_conversions.cairo	6943dca237cef16ff5c93efae68ab0e095294986
DIV	contracts/warplib/maths/div.cairo	a7b0379d45100ec49d38baad2d476a46148fe048
MOD	contracts/warplib/maths/mod.cairo	e63cb3259918b6c9acdcd6a3b7155937cd0ed3b5

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	3	0	3
Informational	0	0	0
Minor	0	0	0
Medium	0	0	0
Major	3	0	3
Critical	0	0	0

1.4 ScaleBit Audit Breakdown

ScaleBit aims to assess repositories for security-related issues, code quality, and compliance with specifications and best practices. Possible issues our team looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Integer overflow/underflow
- Number of rounding errors
- Unchecked External Call
- Unchecked CALL Return Values
- Functionality Checks
- Reentrancy
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic issues
- Gas usage
- Fallback function usage
- tx.origin authentication
- Replay attacks
- Coding style issues

1.5 Methodology

The security team adopted the **"Testing and Automated Analysis"**, **"Code Review"** and **"Formal Verification"** strategy to perform a complete security test on the code in a way that is closest to the real attack. The main entrance and scope of security testing are stated in the conventions in the "Audit Objective", which can expand to contexts beyond the scope according to the actual testing needs. The main types of this security audit include:

(1) Testing and Automated Analysis

Items to check: state consistency / failure rollback / unit testing / value overflows / parameter verification / unhandled errors / boundary checking / coding specifications.

(2) Code Review

The code scope is illustrated in section 1.2.

(3) Audit Process

- Carry out relevant security tests on the testnet or the mainnet;
- If there are any questions during the audit process, communicate with the code owner in time. The code owners should actively cooperate (this might include providing the latest stable source code, relevant deployment scripts or methods, transaction signature scripts, exchange docking schemes, etc.);
- The necessary information during the audit process will be well documented for both the audit team and the code owner in a timely manner.

2 Summary

This report has been commissioned by 10K Swap to identify any potential issues and vulnerabilities in the source code of the 10k Swap smart contract, as well as any contract dependencies that were not part of an officially recognized library. In this audit, we have utilized various techniques, including manual code review and static analysis, to identify potential vulnerabilities and security issues.

During the audit, we identified 3 issues of varying severity, listed below.

ID	Title	Severity	Status
.EN-1	Private Key Exposure in File	Major	Acknowledged
L0K-1	Single-step `feeToSetter` Transfer Can be Dangerous	Major	Acknowledged
L0K1-1	Unrestricted Minting Vulnerability in `10k_erc20.mint()` Contract	Major	Acknowledged

3 Participant Process

Here are the relevant actors with their respective abilities within the 10k Swap Smart Contract:

Fee-setter

- The Fee-Setter can invoke the `setFeeTo` function to designate the recipient of fees.
- The Fee-Setter can invoke the `setFeeToSetter` function, enabling the transfer of the authority to designate the fee recipient.

User

- Users can call the `addLiquidity` function to add liquidity.
- Users can use the `removeLiquidity` function to remove liquidity.
- Users can invoke the `swapExactTokensForTokens` function to acquire the maximum output tokens based on the specified input tokens.
- Users can use the `swapTokensForExactTokens` function to exchange with the minimum input tokens required for the desired output tokens.
- Users can utilize the `swapExactTokensForTokensSupportingFeeOnTransferTokens` function to support transfer fee tokens and obtain the highest amount of output tokens using the specified input tokens.
- Users can create a new pair pool by calling the `createPair` function.
- Users can update the reserves to the current balance by calling the `sync` function.
- Users can withdraw excess amounts by invoking the `skim` function.

4 Findings

.EN-1 Private Key Exposure in File

Severity: Major

Status: Acknowledged

Code Location:

.env.example#L1-9

Descriptions:

When a private key is stored in a file without proper protection or encryption, or if it's inadvertently committed to a public repository, it becomes exposed and can be accessed by unauthorized individuals.

```
OZ_ACCOUNT_ADDRESS =
0x058c64c1abd1183482f140ce4dc5202b766ef6fec03c535fcddc65aa16e2dbbd
OZ_ACCOUNT_PRIVATE_KEY = 0xbdd640fb06671ad11c80317fa3b1799d

ARGENT_ACCOUNT_SALT = 0x42
ARGENT_ACCOUNT_ADDRESS =
0x6f8a9a7d6ec0293734ba6d33e674bc0663e56d37eaf544cd23a8e1ba817219e
ARGENT_ACCOUNT_PRIVATE_KEY =
0x66826acbe6ab1e8612124c0cb413b17695119148aabfe010b1851a9b78ea295

OZ_ACCOUNT_ADDRESS_1 =
0x07b0c3fc6a435b97ec3f9d0938085fea0ebd41244bdd92e3f64d8f2cceaf8aa5
OZ_ACCOUNT_PRIVATE_KEY_1 = 0xbdd640fb06671ad11c80317fa3b1799a
```

Suggestion:

It is recommended not to store the private key locally.

L0K-1 Single-step `feeToSetter` Transfer Can be Dangerous

Severity: Major

Status: Acknowledged

Code Location:

contracts/l0k_factory.cairo#L155-162

Descriptions:

Single-step `feeToSetter` transfer means that if a wrong address was passed when transferring `feeToSetter` it can mean that role is lost forever. If the setter permissions is given to the wrong address within this function, the bad actor can update the address where fees from the contract are sent. He could receive an undue amount of these tokens.it will cause irreparable damage to the contract.

```
@external
func setFeeToSetter{syscall_ptr : felt*, pedersen_ptr : HashBuiltin*, range_check_ptr}(
    feeToSetter : felt
) -> ():
    _onlyFeeToSetter()
    _feeToSetter.write(feeToSetter)
    return ()
end
```

Suggestion:

It is recommended to use a two-step `feeToSetter` transfer pattern.

L0K1-1 Unrestricted Minting Vulnerability in `10k_erc20.mint()` Contract

Severity: Major

Status: Acknowledged

Code Location:

contracts/l0k_erc20.cairo#L113-119

Descriptions:

The `10k_erc20.mint()` function allows external entities to mint a specified amount of ERC20 tokens to a given address. However , it lacks permission checks. This means that, any external entity or user can call this mint function and create new tokens at will.

```
@external
func mint{syscall_ptr : felt*, pedersen_ptr : HashBuiltin*, range_check_ptr}(
    to : felt, amount : Uint256
):
    ERC20._mint(to, amount)
    return ()
end
```

Suggestion:

It is recommended to introduce an ownership mechanism to the contract.

Appendix 1

Issue Level

- **Informational** issues are often recommendations to improve the style of the code or to optimize code that does not affect the overall functionality.
- **Minor** issues are general suggestions relevant to best practices and readability. They don't post any direct risk. Developers are encouraged to fix them.
- **Medium** issues are non-exploitable problems and not security vulnerabilities. They should be fixed unless there is a specific reason not to.
- **Major** issues are security vulnerabilities. They put a portion of users' sensitive information at risk, and often are not directly exploitable. All major issues should be fixed.
- **Critical** issues are directly exploitable security vulnerabilities. They put users' sensitive information at risk. All critical issues should be fixed.

Issue Status

- **Fixed:** The issue has been resolved.
- **Partially Fixed:** The issue has been partially resolved.
- **Acknowledged:** The issue has been acknowledged by the code owner, and the code owner confirms it's as designed, and decides to keep it.

Appendix 2

Disclaimer

This report is based on the scope of materials and documents provided, with a limited review at the time provided. Results may not be complete and do not include all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your own risk. A report does not imply an endorsement of any particular project or team, nor does it guarantee its security. These reports should not be relied upon in any way by any third party, including for the purpose of making any decision to buy or sell products, services, or any other assets. TO THE FULLEST EXTENT PERMITTED BY LAW, WE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, IN CONNECTION WITH THIS REPORT, ITS CONTENT, RELATED SERVICES AND PRODUCTS, AND YOUR USE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NOT INFRINGEMENT.