Smart Contract Audit for Forthewin

Overlord Security

June 6, 2022

Contents

1 Project Overview

Created by: Eddie Jung

Based on: Neo Blockchain

Date Conducted: April , 2022

Forthewin contract

Contracts: **FTWSwap, FTWStaking**
Github: `https://github.com/ForTheWinn/FTW-N3-Contracts`
Programming Language: **C#**
OS Env: **Neo 3.1.0**

2 Project Introduction

Forthewin ecosystem will create a platform where ordinary users and businesses can easily use both Fungible tokens and NFTs in their daily lives and find more use cases. The motivation is to give everyone the opportunity to create and manage both Fungible tokens and NFTs, help them be successful and allow their tokens to be more heavily adopted into every day life.

3 Findings and Recommendations

3.1 Summary

The following findings and recommendations after analyzing the **Forthewin** implementation. Any additional recommendations beyond what any scanning tools supply are included as necessary.

Severity	Number of findings
Critical	7
Medium	1
Low	8
Informational	5

Issue Id	Severity	Title	Category	Fixed
MS-01	Critical	Lack Inputs of Validation	Coding Practices	Fixed
MS-02	Critical	Issue of Lp Amount Calculation	Business Logic	Fixed
MS-03	Critical	Assert Instead of Throw	Coding practices	Fixed
MS-04	Low	Unnecessary Check for Max Value	Coding practices	Fixed
MS-05	Info	GAS Fee Optimization	Optimization	Fixed
MS-06	Info	Variable Name Style	Coding practices	Fixed
MS-07	Low	Expression is always true	Business Logic	Fixed
MS-08	Info	Redundant Inherit Declaration	Coding practices	Fixed
MS-09	Info	Typos	Coding practices	Fixed
MS-10	Low	Unnecessary Check for _deploy	Business Logic	Fixed
MS-11	Low	Page Calculation Mistake	Business Logic	Fixed
MS-12	Critical	Prefix Conflict	Coding practices	Fixed
MS-13	Low	Unnecessary Calculation for pair key	Business Logic	Fixed
MS-14	Low	Lack Inputs of Validation for AddPair	Coding practices	Fixed
MS-15	Info	Invalid Check in UpdatePairReward	Business Logic	Fixed
MS-16	Low	IsReEntered optimization	Coding practices	Fixed
MS-17	Medium	WitnessScope optimization	Coding Practices	Confirmed
MS-18	Low	Redundant Variables Declaration	Coding Practices	Fixed
MS-19	Critical	Invalid Check in Stake	Coding Practices	Fixed
MS-20	Critical	Contract Stuck Risk	Security Features	Confirmed
MS-21	Critical	Avoid using exception in public interface	Coding Practices	Fixed

3.2 Critical Vulnerabilities

MS-01: Lack Inputs of Validation

Lack of Inputs Validation in FTWswap

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.cs#L171

Description

There is no validation check for the input parameters "amountIn" Which means if the asset is not standard NEP-17 and amountIn $<$ 0 pass the "SafeTransfer". Any user can use "amountIn $<$ 0" to withdraw "fee" from the contract.

Solution

It is recommend to add this check. Assert(amountA $>$ 0, "Amount A should be more than 0.");

Status

The issue has been confirmed by team and fixed in commit `d462b6a`

MS-02: Issue of Lp Amount Calculation

Lp Amount Calculation in FTWswap

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.cs#L108

Description

Currently, contract use "amountA" as the lp amount. There is some issue in this formula.
For example The initial situation: User A add liquidity: X: 10000, Y: 10000 Then User A have Lp: 10000
Then the price changes The pool balance: X: 1, Y: 100000000
The User B add liquidity X: 1, Y: 100000000
The pool balance: X: 2, Y: 200000000
But the User B only have Lp amount 1. The User B can not get his asset back after remove liquidity.

Solution

Recommend to use formula: sqrt(amountA * amountB) to mark Lp amount

Status

The issue has been confirmed by team and fixed in commit `0d8a401`

MS-03: Assert Instead of Throw
Use Throw Exception in the FTWswap
Source Code link
https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.cs#L218
Description
Due to throw exception can be catched by contract. And **SafeTransfer** will not be roll back. Which will cause the asset lost of user.
Solution
Recommend to use **Assert** to instead of **Throw**;
Status
The issue has been confirmed by team and fixed in commit `55607ef`

MS-12: Prefix Conflict

Prefix Conflict

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.cs#L20-L21

Description

Because FTWSwap is a Nep11Token and Nep11Token has already taken some prefixes before, consider starting from 0x5.

Solution

Recommend to strart prefix starting from **0x5**

Status

The issue has been confirmed by team and fixed in commit `2735ee7`

MS-19: Invalid Check

Invalid Check in Stake

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/70bf0626e5f9b8611a0b9e6e1553ffd31e1ee632/contracts/FTWStaking/FTWStaking.cs#L52

Description

Use if (!currentStaking.Amount.IsZero !TVL.IsZero) instead of if (currentStaking.Amount.IsZero !TVL.IsZero). Otherwise, user will lose their money.

Solution

Use if (!currentStaking.Amount.IsZero !TVL.IsZero)

Status

The issue has been confirmed by team and fixed in commit `6c96022`

MS-20: Contract Stuck Risk

Contract Stuck Risk

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/70bf0626e5f9b8611a0b9e6e1553ffd31e1ee632/contracts/FTWStaking/FTWStaking.cs#L149

Description

If someone throw exception when you mint the token to them and they catch the exception after that, the contract will stuck there forever because of Re-entrance Check.

Solution

Adding a try catch to each external call so that no exception can be throw out from our contracts.

Status

The issue has been confirmed by team.

MS-21: Avoid using exception in public interface

Avoid using exception in public interface

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/70bf0626e5f9b8611a0b9e6e1553ffd31e1ee632/contracts/FTWStaking/FTWStaking.Helpers.cs#L17

Description

Do not use any exception in the public interface cause hackers may catch it in an attack contract.

Solution

Remove exception.

Status

The issue has been confirmed by team and fixed in commit `a21460b`

3.3 Medium Vulnerabilities

MS-17: WitnessScope optimization

WitnessScope optimization in FTWStaking

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.cs#L171

Description

Require user transfer their Nep11 from our contract will require user add witnessScope to their tx.signer, this may cause a warning on user's wallet.
If we put staking logics into OnNep11Payment, and do the staking logics after receiving those Nep11, the user will only need to send their Nep11 to us for staking. No more waning in their wallets.

Solution

Status

3.4 Low Vulnerabilities

MS-04: Unnecessary Check for Max Value
Unnecessary Check for Max Value
Source Code link
https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.cs#L219
Description
Due to input amount is fixed. It is unnecessary to check max value.
Solution
It is recommend to remove this.
Status
The issue has been confirmed by team and fixed in commit `8111adf`

MS-07: Expression is always true

Expression is always true

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.Nep11.cs#L27

Description

The statement of meta.LockUntil != null is always true, therefore this if check is useless.

Solution

It is recommend to remove this.

Status

The issue has been confirmed by team and fixed in commit `c0cc15a`

MS-10: Unnecessary Check

Unnecessary Check for _deploy

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.Owner.cs#L14-L21

Description

Due to method _deploy can not be called other than deploy and update by ContractManagement.
It is unnecessary to check whether "Contract already deployed".

Solution

Remove those check

Status

The issue has been confirmed by team and fixed in commit `3747638`

MS-11: Wrong Calculation

Page Calculation Mistake

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.Owner.cs#L14-L21

Description

To make sure the divider not be zero, we'd better set a minimum bound for itemsPerPage.
If the height is divisible by itemsPerPage, the totalPages will be not accurate.

Solution

using BigInteger itemsPerPage = 30 > height ? height : 30; itemsPerPage = 0 != itemsPerPage ? itemsPerPage : 1.
using BigInteger totalPages = (itemsPerPage + height - 1) / itemsPerPage;

Status

The issue has been confirmed by team and fixed in commit `3747638`

MS-13: Unnecessary Calculation for pair key
Unnecessary Calculation for pair key
Source Code link
https://github.com/ForTheWinn/FTW-N3-Contracts/blob/70bf0626e5f9b8611a0b9e6e1553ffd31e1ee632/contracts/FTWStaking/FTWStaking.Gets.cs#L40-L47
Description
here already exists a method named GetPairKey which has the same functionality.
Solution
Changed to return pair data instead of returning pairkey.
Status
The issue has been confirmed by team.

MS-14: Lack Inputs of Validation

Lack Inputs of Validation for AddPair

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/70bf0626e5f9b8611a0b9e6e1553ffd31e1ee632/contracts/FTWStaking/FTWStaking.Owner.cs#L16-L22

Description

To prevent from adding a pair twice by mistake, it's better to check whether this pair exists before or not.

Solution

Add a check if pair exists first.

Status

The issue has been confirmed by team and fixed in commit `ed558e5`

MS-16: IsReEntered optimization

IsReEntered optimization

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/70bf0626e5f9b8611a0b9e6e1553ffd31e1ee632/contracts/FTWSwap/FTWSwap.Helpers.cs#L50-L58
https://github.com/ForTheWinn/FTW-N3-Contracts/blob/70bf0626e5f9b8611a0b9e6e1553ffd31e1ee632/contracts/FTWStaking/FTWStaking.Helpers.cs#L63-L72

Description

The other cheaper method in NEO is to use the **syscall InvocationCounter**.
This syscall charges only 16 gas instead of storage operation's 32768 gas.
Don't afraid entering a function multi-times such as implementing ClaimMulti by multiple Claim. The InvocationCounter only increase when other contract call us from external ContractCall.

Solution

Try to use the **syscall InvocationCounter**.

Status

The issue has been confirmed by team.

MS-18: Redundant Variables Declaration
Redundant Variables Declaration
Source Code link
https://github.com/ForTheWinn/FTW-N3-Contracts/blob/70bf0626e5f9b8611a0b9e6e1553ffd31e1ee632/contracts/FTWStaking/FTWStaking.cs#L22-L27
Description
name and lockUntil is useless in staking, omit them will lower the GAS fee.
Solution
Remove unused variable.
Status
The issue has been confirmed by team and fixed in commit `08cd5e6`.

3.5 Informational Vulnerabilities

MS-05: GAS Fee Optimization
GAS Fee Optimization
Source Code link
https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.Helpers.cs#L23
Description
[MethodImpl(MethodImplOptions.AggressiveInlining)] on these short private functions to save the transaction gas fee.
Solution
It is recommend to use this to optimize GAS
Status
The issue has been confirmed by team and fixed in commit `30c08d4`

MS-06: Variable Name Style

Variable Name Style

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.states.cs#L36

Description

Name style of these variables are not consistent:
public BigInteger totalItems;
public BigInteger totalPages;
public BigInteger currentpage;

Solution

Suggestion is currentPage.

Status

The issue has been confirmed by team and fixed in commit `3f4e3b4`

MS-08: Redundant Inherit Declaration

Redundant Inherit Declaration

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.cs#L17

Description

You only need to add parent class : Nep11Token<ShareToken> in one file.

Solution

You only need to add parent class : Nep11Token<ShareToken> in one file.

Status

The issue has been confirmed by team and fixed in commit `11d9b4e`

MS-09: Typos

Typos

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/15ed8a94832745b5a1527cd0b86ff1d45b7f48c0/contracts/FTWSwap/FTWSwap.cs#L31

Description

Assert(Runtime.CheckWitness(account), "You are not onwer.");
Assert(GetDecimals(tokenA) == 8, "We only support toekns with 8 decimals.");
Assert(GetDecimals(tokenB) == 8, "We only support toekns with 8 decimals.");

Solution

Fix the typos

Status

The issue has been confirmed by team and fixed in commit `56db754`

MS-15: Invalid Check

Invalid Check in UpdatePairReward

Source Code link

https://github.com/ForTheWinn/FTW-N3-Contracts/blob/70bf0626e5f9b8611a0b9e6e1553ffd31e1ee632/contracts/FTWStaking/FTWStaking.Owner.cs#L51-L54

Description

If you don't want the user lose money, the condition should be Assert(dailyReward >= 0, "Please check dailyStakingReward.");.
To keep code's consistency, consider change AddPair's checking from Assert(dailyReward > 0, ... to Assert(dailyReward >= 0, ...

Solution

Status

The issue has been confirmed by team and fixed in commit `ab0a553`.

4 Conclusion

In this audit, we have analyzed the Forthewin swap and staking design and implementation. The current code base is well organized and those identified issues are promptly confirmed and fixed.

Meanwhile, we need to emphasize that smart contracts as a whole are still in an early, but active stage of development. To improve this report, we greatly appreciate any constructive feedbacks or suggestions, on our methodology, audit findings, or potential gaps in scope/coverage.