GUARDIAN AUDITS

SMART CONTRACT SECURITY AUDIT OF

INFINITY LOTTO

Summary

Audit Firm: Guardian Audits

Client Firm: Infinity Lotto

Final Report Date - Preliminary Report

Audit Summary

After a line by line manual analysis and automated review, Guardian has concluded that:

- Infinity Lotto's smart contracts have a MEDIUM RISK SEVERITY
- Infinity Lotto's smart contracts have an **ACTIVE OWNERSHIP**
- Important owner privileges – authorize, unauthorize, transferOwnership, addStakingContract, removeBadStakingContract, setAutomatedMarketMakerPair, updateClaimWait, setMaxWalletPercent_base1000, tradingStatus, cooldownEnabled, enable_blacklist, manage_blacklist, setSellMultiplier, multiAirdrop, multiAirdrop_fixed, addTeamDivWallet, removeTeamDivWallet, setIsFeeExempt, setGoldenModeTaxByIs0, setIsTimelockExempt, setIsTxLimitExempt, setIsMaxWalletExempt, setContractFees, setFeeContract, setSwapBackSettings, setDistributorSettings, withdrawlToken, updateRouter
- Infinity Lotto's smart contract owner has multiple "write" privileges. Centralization risk correlated to the active ownership is **HIGH**

Notice that the examined smart contracts are not resistant to internal exploit. For a detailed understanding of risk severity, source code vulnerability, and potential attack vectors, refer to the complete audit report below.

🔗 Blockchain network: **BSC**

✅ Verify the authenticity of this report on Guardian's GitHub: https://github.com/guardianaudits

Table of Contents

Project Overview

Project Summary

Project Name	InfinityLotto
Language	Solidity
Codebase	https://bscscan.com/address/0x9144Ab67d29a6B9819655A850036a7Db7DE4bbe8#code
Commit	N/A

Audit Summary

Delivery Date	Preliminary Report
Audit Methodology	Static Analysis, Manual Review

Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0	0	0
High	1	1	0	0	0	0
Medium	0	0	0	0	0	0
Low	11	11	0	0	0	0

Audit Scope & Methodology

Scope

ID	File	SHA-1 Checksum
LOT	InfinityLotto2.sol	e2bb5cd8ab64c39835fcbcefccab7a025514e707

Methodology

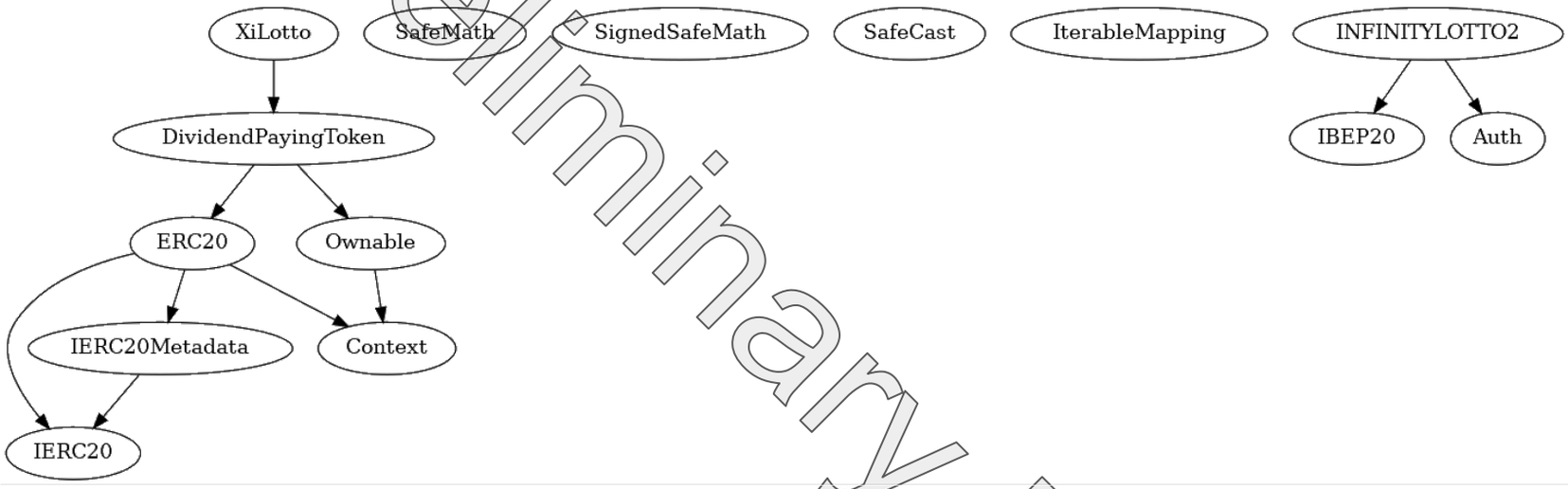
The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross-referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

Vulnerability Classifications

Vulnerability Level	Classification
🔴 Critical	Easily exploitable by anyone, causing loss/manipulation of assets or data.
🟠 High	Arduously exploitable by a subset of addresses, causing loss/manipulation of assets or data.
🟡 Medium	Inherent risk of future exploits that may or may not impact the smart contract execution.
🟢 Low	Minor deviation from best practices.

Inheritance Graph



Findings & Resolutions

ID	Title	Category	Severity	Status
LOT-1	Centralization Risk	Centralization / Privilege	High	Unresolved
LOT-2	Multiplication on Result of Division	Precision	Low	Unresolved
LOT-3	Constant Modifiers	Mutability	Low	Unresolved
LOT-4	Immutable Modifiers	Mutability	Low	Unresolved
LOT-5	SafeMath Operations	Best Practices	Low	Unresolved
LOT-6	Superfluous Mapping	Optimization	Low	Unresolved
LOT-7	Internal Functions	Best Practices	Low	Unresolved
LOT-8	External Modifiers	Best Practices	Low	Unresolved
LOT-9	Lack of CamelCase	Best Practices	Low	Unresolved
LOT-10	Typo	Typo	Low	Unresolved
LOT-11	Residual MaxWalletExemption	Logical Error	Low	Unresolved
LOT-12	Boolean Redundancy	Optimization	Low	Unresolved

LOT-1 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	High	InfinityLotto.sol	Unresolved

Description

Privileged addresses have authority over many functions that may be used to negatively disrupt the project. Some important privileges include:

- `owner` can blacklist an address preventing it from swapping ILOTTO tokens. Additionally, the token pair or exchange router could be blacklisted, which would cease trading.
- `owner` can set `isMaxWalletExempt` to false for the pair contract, thus halting all trading.
- `owner` can label an address a `teamDivWallet` which would instantaneously create more XiLotto dividend tokens and would dilute the future dividends of other holders. Furthermore, this allows the team to to dump on the market without any cooldown.
- `owner` can toggle `tradingStatus` to `false` which would prevent users from transferring funds and cease trading on all exchanges.
- `owner` can `updateClaimWait` to an arbitrarily long period, potentially preventing XiLotto holders from ever claiming their dividends.
- Any `authorized` address can update the `router` to a potentially malicious contract that causes a denial-of-service via reverting, or siphons funds upon execution of `swapBack` rather than distributing these funds as dividends.
- `owner` can set the `cooldownTimerInterval` to an arbitrarily long length of time, making it so that addresses are potentially only able to buy ILotto2 once. Additionally, `owner` can combine an extremely long `cooldownTimerInterval` with adding the exchange's router as an `automatedMarketMakerPair`, this way any address can potentially only buy/sell one time.

Recommendation

Currently the `owner` address is not a multi-sig. Ensure that the privileged addresses are multi-sig and/or introduce timelock for improved community oversight. Optionally introduce `require` statements to limit the scope of the exploits that can be carried out by the privileged addresses.

LOT-2 | Multiplication On Result Of Division

Category	Severity	Location	Status
Precision	Low	InfinityLotto.sol: 1885	Unresolved

Description

In the function `takeFee`: `uint256 feeAmount = amount.div(feeDenominator * 100).mul(totalFee).mul(multiplier)` performs multiplication on the result of division, which leads to a loss in precision.

For example: 100.div(100 * 100).mul(10).mul(100) = 0 but 100.mul(10).mul(100).div(100 * 100) = 10

Recommendation

Change to `uint256 feeAmount = (amount * totalFee * multiplier) / (feeDenominator * 100)`.

Resolution

LOT-3 | Constant Modifiers

Category	Severity	Location	Status
Mutability	Low	InfinityLotto.sol	Unresolved

Description

Contract variables such as `MAX_INT`, `RWRD`, `DEAD`, `ZERO`, `_totalSupply`, `feeDenominator` can be declared `constant`.

Recommendation

Declare the variables `constant`.

Resolution

LOT-4 | Immutable Modifiers

Category	Severity	Location	Status
Mutability	Low	InfinityLotto.sol	Unresolved

Description

The `WBNB` and `distributor` variables are never modified after they are set in the constructor, and should therefore be declared `immutable`.

Recommendation

Declare the variables `immutable`.

Resolution

LOT-5 | SafeMath Operations

Category	Severity	Location	Status
Best Practices	Low	InfinityLotto.sol	Unresolved

Description

There is no need for `add`, `sub`, `mul`, and `div` in Solidity version `^0.8.0` as there are already implicit overflow and underflow checks.

Recommendation

Use language provided operators `+`, `-`, `*`, `/` to save on gas.

Resolution

LOT-6 | Superfluous Mapping

Category	Severity	Location	Status
Optimization	Low	InfinityLotto.sol: 1500	Unresolved

Description

In the XiLotto contract, the `tokenHoldersMap` is declared as an `IterableMapping`, but it is only ever used to access the values of the keys. Therefore the use of `IterableMapping` is gas inefficient and it can be replaced as a list.

Recommendation

Replace the `tokenHoldersMap` with a list of tokenHolders.

Resolution

LOT-7 | Internal Functions

Category	Severity	Location	Status
Best Practices	Low	InfinityLotto.sol	Unresolved

Description

Internal functions should be denoted with a preceding _: checkTxLimit, shouldTakeFee, takeFee, shouldSwapBack, swapBack, swapAndSendToDiv.

Recommendation

Rename these functions to _checkTxLimit, _shouldTakeFee, _takeFee, _shouldSwapBack, _swapBack, _swapAndSendToDiv.

Resolution

LOT-8 | External Modifiers

Category	Severity	Location	Status
Best Practices	Low	InfinityLotto.sol	Unresolved

Description

Many `public` functions can be declared `external`: `tradingStatus`, `cooldownEnabled`, `enable_blacklist`, `manage_blacklist`, `getCirculatingSupply`, `addTeamDivWallet`, `setAutomatedMarketMakerPair`.

Recommendation

Declare these functions `external` as they are never called internally.

Resolution

LOT-9 | Lack of CamelCase

Category	Severity	Location	Status
Best Practices	Low	InfinityLotto.sol: 1979, 1983	Unresolved

Description

Function names should adhere to camelCase: `enable_blacklist`, `manage_blacklist`.

Recommendation

Introduce camelCase instead of snake_case.

Resolution

LOT-10 | Typo

Category	Severity	Location	Status
Typos	Low	InfinityLotto.sol: 2078	Unresolved

Description

`withdrawlToken` should be `withdrawalToken`.

Recommendation

Fix spelling for cleaner code.

Resolution

LOT-11 | Residual MaxWalletExemption

Category	Severity	Location	Status
Logical Error	Low	InfinityLotto.sol: 1743	Unresolved

Description

When `removeBadStakingContract` is called, `isMaxWalletExempt` is not reset to `false`.

Recommendation

If this is not intended, perform `isMaxWalletExempt[badStakingContract] = false;` or `delete isMaxWalletExempt[badStakingContract]`.

Resolution

LOT-12 | Boolean Redundancy

Category	Severity	Location	Status
Optimization	Low	InfinityLotto.sol: 2093	Unresolved

Description

In `updateRouter` the final `if` statement condition is `automatedMarketMakerPairs[pair] != true`, but this is redundant since the mapping values are booleans themselves.

Recommendation

Replace the `if` statement condition with a more gas efficient `!automatedMarketMakerPairs[pair]`.

Resolution

Auditor's Verdict

After a line by line manual analysis and automated review, Guardian has concluded that:

- Infinity Lotto's smart contracts have a MEDIUM RISK SEVERITY
- Infinity Lotto's smart contracts have an **ACTIVE OWNERSHIP**
- Important owner privileges – authorize, unauthorize, transferOwnership, addStakingContract, removeBadStakingContract, setAutomatedMarketMakerPair, updateClaimWait, setMaxWalletPercent_base1000, tradingStatus, cooldownEnabled, enable_blacklist, manage_blacklist, setSellMultiplier, multiAirdrop, multiAirdrop_fixed, addTeamDivWallet, removeTeamDivWallet, setIsFeeExempt, setGoldenModeTaxByIs0, setIsTimelockExempt, setIsTxLimitExempt, setIsMaxWalletExempt, setContractFees, setFeeContract, setSwapBackSettings, setDistributorSettings, withdrawlToken, updateRouter
- Infinity Lotto's smart contract owner has multiple "write" privileges. Centralization risk correlated to the active ownership is **HIGH**

Disclaimer

This report is not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Guardian to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. Guardian’s position is that each company and individual are responsible for their own due diligence and continuous security. Guardian’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by Guardian is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

Notice that smart contracts deployed on the blockchain are not resistant from internal/external exploit. Notice that active smart contract owner privileges constitute an elevated impact to any smart contract’s safety and security. Therefore, Guardian does not guarantee the explicit security of the audited smart contract, regardless of the verdict.

About Guardian Audits

Founded in 2022 by DeFi experts, Guardian Audits is a leading audit firm in the DeFi smart contract space. With every audit report, Guardian Audits upholds best-in-class security while achieving our mission to relentlessly secure DeFi.

To learn more, visit https://guardianaudits.com

To view our audit portfolio, visit https://github.com/guardianaudits

To book an audit, message https://t.me/guardianaudits