PALADIN
BLOCKCHAIN SECURITY

Smart Contract Security Assessment

Final Report

For MarsDAO

24 February 2022

paladinsec.co

info@paladinsec.co

Table of Contents

Table of Contents 2

Disclaimer 3

1 Overview 4

1.1 Summary 4

1.2 Contracts Assessed 4

1.3 Findings Summary 5

1.3.1 GovernanceMarsDAO 6

1.3.2 StratX 6

1.3.3 BStratX 6

1.3.4 MarsAutoFarm 7

1.3.5 MarsAutoFarmGovernance 7

2 Findings 8

2.1 GovernanceMarsDAO 8

2.1.2 Issues & Recommendations 9

2.2 StratX 10

2.2.1 Privileged Roles 10

2.2.2 Issues & Recommendations 11

2.3 BStratX 15

2.3.1 Privileged Roles 15

2.3.2 Issues & Recommendations 16

2.4 MarsAutoFarm 20

2.4.1 Privileged Roles 20

2.4.2 Issues & Recommendations 21

2.5 MarsAutoFarmGovernance 24

2.5.1 Issues & Recommendations 25

Disclaimer

Paladin Blockchain Security ("Paladin") has conducted an independent audit to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the codes that were provided for the scope of this audit. This audit report does not constitute agreement, acceptance or advocation for the Project that was audited, and users relying on this audit report should not consider this as having any merit for financial advice in any shape, form or nature. The contracts audited do not account for any economic developments that may be pursued by the Project in question, and that the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are completely free of exploits, bugs, vulnerabilities or deprecation of technologies. Further, this audit report shall not be disclosed nor transmitted to any persons or parties on any objective, goal or justification without due written assent, acquiescence or approval by Paladin.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence regardless of the findings presented in this report. Information is provided 'as is', and Paladin is under no covenant to the completeness, accuracy or solidity of the contracts audited. In no event will Paladin or its partners, employees, agents or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions and/or actions with regards to the information provided in this audit report.

Cryptocurrencies and any technologies by extension directly or indirectly related to cryptocurrencies are highly volatile and speculative by nature. All reasonable due diligence and safeguards may yet be insufficient, and users should exercise considerable caution when participating in any shape or form in this nascent industry.

The audit report has made all reasonable attempts to provide clear and articulate recommendations to the Project team with respect to the rectification, amendment and/or revision of any highlighted issues, vulnerabilities or exploits within the contracts provided. It is the sole responsibility of the Project team to sufficiently test and perform checks, ensuring that the contracts are functioning as intended, specifically that the functions therein contained within said contracts have the desired intended effects, functionalities and outcomes of the Project team.

Paladin retains full rights over all intellectual property (including expertise and new attack or exploit vectors) discovered during the audit process. Paladin is therefore allowed and expected to re-use this knowledge in subsequent audits and to inform existing projects that may have similar vulnerabilities. Paladin may, at its discretion, claim bug bounties from third-parties while doing so.

1 Overview

This report has been prepared for MarsDAO on the Binance Smart Chain. Paladin provides a user-centred examination of the smart contracts to look for vulnerabilities, logic errors or other issues from both an internal and external perspective.

1.1 Summary

Project Name	Mars DAO
URL	http://auto.farm
Platform	Binance Smart Chain
Language	Solidity

1.2 Contracts Assessed

Name	Contract	Live Code Match
GovernanceMarsDAO	0x585d49b69b0f5020243E9f3f89A9dbCc5D163FbB	MATCH
StratX	0x643A1442f92b508bE732ce58391A098687CB9bAF (all StratX and BStratX deployments should be checked against the code in this example contract)	MATCH
BStratX		
MarsAutoFarm	0x5aEF70fb368b930f3129a5EcD795a6Bb2678C338	MATCH
MarsAutoFarmGovernance	0x9431E5Ccc83A514BFAEdDe729853A7B20BfD83de	MATCH

1.3 Findings Summary

Severity	Found	Resolved	Partially Resolved	Acknowledged (no change made)
High	0	-	-	-
Medium	0	-	-	-
Low	10	2	-	8
Informational	11	2	-	9
Total	**21**	**4**	**-**	**17**

Classification of Issues

Severity	Description
High	Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, or impairment of the contract and its functions. Issues under this classification are recommended to be fixed with utmost urgency.
Medium	Bugs or issues that may be subject to exploit, though their impact is somewhat limited. Issues under this classification are recommended to be fixed as soon as possible.
Low	Effects are minimal in isolation and do not pose a significant danger to the project or its users. Issues under this classification are recommended to be fixed nonetheless.
Informational	Consistency, syntax or style best practices. Generally pose a negligible level of risk, if any.

1.3.1 GovernanceMarsDAO

ID	Severity	Summary	Status
01	INFO	Inconsistency with the ERC20 standard	ACKNOWLEDGED
02	INFO	Unnecessary allowance check added in `mint`	ACKNOWLEDGED

1.3.2 StratX

ID	Severity	Summary	Status
03	LOW	The `burnRate` and `buyBackRate` are set to 100 percent initially	RESOLVED
04	LOW	Missing `whenNotPaused` modifier	ACKNOWLEDGED
05	LOW	There should be a check that the contract is not paused before making a call to `_helptoEarn`	ACKNOWLEDGED
06	INFO	No event emission present to capture the new states	ACKNOWLEDGED
07	INFO	Several functions can be declared as external	RESOLVED

1.3.3 BStratX

ID	Severity	Summary	Status
08	LOW	The `burnRate` and `buyBackRate` are set to 100 percent initially	RESOLVED
09	LOW	Missing `whenNotPaused` modifier	ACKNOWLEDGED
10	LOW	There should be a check that the contract is not paused before making a call to `_helptoEarn`	ACKNOWLEDGED
11	INFO	Events should be emitted for functions that alter states	ACKNOWLEDGED
12	INFO	Several functions can be declared as external	RESOLVED

1.3.4 MarsAutoFarm

ID	Severity	Summary	Status
13	LOW	Missing `nonReentrant` modifier	ACKNOWLEDGED
14	LOW	There should be a check for shares to be more than zero for overly small deposits	ACKNOWLEDGED
15	INFO	Several functions can be declared as external	ACKNOWLEDGED
16	INFO	Unnecessary parsing of `msg.sender` as `address`	ACKNOWLEDGED
17	INFO	Fee-on-transfer tokens are not supported	ACKNOWLEDGED

1.3.5 MarsAutoFarmGovernance

ID	Severity	Summary	Status
18	LOW	Use of non-standard packed mode	ACKNOWLEDGED
19	LOW	SafeMath is not used for arithmetic operations	ACKNOWLEDGED
20	INFO	Several functions can be declared as external	ACKNOWLEDGED
21	INFO	Unnecessary parsing of `msg.sender` as `address`	ACKNOWLEDGED

2 Findings

2.1 GovernanceMarsDAO

The GovernanceToken is a token contract which allows investors to in exchange for the New Mars DAO Token. The users provide the New Mars DAO Token which are sent to the burn address and the same amount of Governance tokens are minted for the user.

2.1.2 Issues & Recommendations

Issue #01	Inconsistency with the ERC20 standard
Severity	INFORMATIONAL
Location	Lines 21–27
Description	`msg.sender` is used to get the caller address. As per ERC20 standard, `_msgSender()` is the go-to criteria. In addition to that, `_msgSender()` returns users for meta transactions instead of the relayer.
Recommendation	Consider using `_msgSender()` instead of `msg.sender` to be consistent with the ERC20 standard.
Resolution	ACKNOWLEDGED

Issue #02	Unnecessary allowance check added in `mint`
Severity	INFORMATIONAL
Location	Lines 21–27
Description	As the contract is based on standard ERC20 token, the allowance check is already handled by the inherited contract. There is no need to use extra gas to perform the same functionality.
Recommendation	Consider removing the following code:

```
require(newMarsDAOToken.allowance(msg.sender, address(this))
>=amount,
    "Increase the allowance first,call the approve method"
);
```

Resolution	ACKNOWLEDGED

2.2 StratX

StratX is an automated farming contract, which allows users to deposit and withdraw to the PancakeFarm with the best possible outcome. This allows users to farm the wanted token and allows the owner to set up a strategy for possible paths between two tokens. At the deployment of the contract, the owner is set to the `marsAutoFarmAddress`.

2.2.1 Privileged Roles

The following functions can be called by the owner of the contract:

- `activateStrategy`
- `deposit`
- `withdraw`

The following functions can be called by the admin of the contract:

- `pause`
- `unpause`
- `setRouter0`
- `setRouter1`
- `setRouter2`
- `setBurnRate`
- `setGov`
- `setSwapSlippage`
- `inCaseTokensGetStuck`

2.2.2 Issues & Recommendations

Issue #03	The burnRate and buyBackRate are set to 100 percent initially
Severity	LOW SEVERITY
Location	Line 54 Line 59

Description	Upon deployment, the burnRate and buyBackRate are set to 100 percent.
	
	For example, consider this function:
	
	`function distributeReward() external {`
	`  uint256 rewardAmount=IERC20(marsTokenAddress).balanceOf(address(this));`
	`  //min amount 1e7`
	`  if(rewardAmount>1e7 && sharesTotal>0){`
	`  uint256 burnAmount=rewardAmount.mul(burnRate).div(MaxBP);`
	`  IERC20(marsTokenAddress).safeTransfer(burnAddress, burnAmount);`
	`  rewardAmount=rewardAmount.sub(burnAmount);`
	`  IERC20(marsTokenAddress).safeIncreaseAllowance(marsAutoFarmAddress, rewardAmount);`
	`  require(IMarsAutoFarm(marsAutoFarmAddress).chargePool(marsPid,rewardAmount,sharesTotal),`
	`  "pool charging fail");`
	`  }`
	`}`
	
	For the initial value 100% for burnRate, the burn amount will be equal to the reward amount, thus burning all rewards.
	
	On the other hand, in case of buy back, all the rewards will get used under buy back.

Recommendation	Consider changing the buyBackRate and burnRate to a lower value to allow users to get the rewards.
Resolution	RESOLVED The initial buyback rate is changed to 48.45% and burn rate is changed to 20%.

Issue #04	Missing whenNotPaused modifier
Severity	LOW SEVERITY
Location	`Line 201-203`
Description	The `farm` function is missing the `whenNotPaused` modifier. This allows the user to stake even when the deposits are paused.
Recommendation	Consider adding the `whenNotPaused` modifier to the function.
Resolution	ACKNOWLEDGED The pause is not applied so that the user will be able to collect their rewards.

Issue #05	There should be a check that the contract is not paused before making a call to `_helptoEarn`
Severity	LOW SEVERITY
Location	`Lines 219-254`
Description	In `withdraw`, `if(!isEmergency){_helpToEarn();}` should also check that it is not paused before `helpToEarn` is called.
Recommendation	Consider adding a boolean flag which is set to true once the call is made to the `setRouter` functions. `if(!isEmergency && !paused()){_helpToEarn();}`
Resolution	ACKNOWLEDGED The pause is not applied as per the business logic.

Issue #06	No event emission present to capture the new states
Severity	INFORMATIONAL
Location	Lines 92-142 Lines 175-199 Lines 219-258 Lines 376-387 Lines 401-428 Lines 438-445 Lines 447-454 Lines 456-463 Lines 465-469 Lines 471-475 Lines 477-485 Lines 487-490
Description	Events should be emitted for functions that modify the underlying state of the contract.
Recommendation	Consider adding the events to specify the states which are being modified.
Resolution	ACKNOWLEDGED

Issue #07	Several functions can be declared as `external`
Severity	INFORMATIONAL
Description	Following functions under the contract can be declared as external to avoid any kind of unwanted exposure. - `deposit` - `pause` - `setRouter0, setRouter1, & setRouter2` - `setGov` - `setBuybackRate` - `setBurnRate` - `inCaseTokensGetStuck` - `convertDustToEarned` - `farm` Apart from being a best practice when the function is not used within the contract, this can lead to lower gas usage in certain cases.
Recommendation	Consider marking the above functions as `external`.
Resolution	RESOLVED

2.3 BStratX

BStratX is an automated farming contract, which allows users to deposit and withdraw into the Biswap Masterchef with the best possible outcome. This allows users to farm the wanted token and allows the owner to set up a strategy for possible paths between two tokens. At the deployment of the contract, the owner is set to the `marsAutoFarmAddress`. This contract is very similar in functionality to StratX — the major difference being it interacts with BiSwap Protocol.

2.3.1 Privileged Roles

The following functions can be called by the owner:

- `activateStrategy`
- `deposit`
- `withdraw`

The following functions can be called by the admin:

- `pause`
- `unpause`
- `setRouter0`
- `setRouter1`
- `setRouter2`
- `setBurnRate`
- `setGov`
- `setSwapSlippage`
- `inCaseTokensGetStuck`

2.3.2 Issues & Recommendations

Issue #08	The burnRate and buyBackRate are set to 100 percent initially
Severity	LOW SEVERITY
Location	Line 56 Line 61

Description	Upon deployment, the burnRate and buyBackRate are set to 100 percent.
	
	For example consider this function:
	
	`function distributeReward() external {`
	`  uint256 rewardAmount=IERC20(marsTokenAddress).balanceOf(address(this));`
	`  //min amount 1e7`
	`  if(rewardAmount>1e7 && sharesTotal>0){`
	`  uint256 burnAmount=rewardAmount.mul(burnRate).div(MaxBP);`
	`  IERC20(marsTokenAddress).safeTransfer(burnAddress, burnAmount);`
	`  rewardAmount=rewardAmount.sub(burnAmount);`
	`  IERC20(marsTokenAddress).safeIncreaseAllowance(marsAutoFarmAddress, rewardAmount);`
	`  require(IMarsAutoFarm(marsAutoFarmAddress).chargePool(marsPid,rewardAmount,sharesTotal),`
	`  "pool charging fail");`
	`  }`
	`}`
	
	For the initial value 100% for burnRate, the burn amount will be equal to the reward amount, thus burning all rewards.
	
	On the other hand, in case of buy back, all the rewards will get used under buy back.

Recommendation	Consider changing the buyBackRate and burnRate to a lower value to allow users to get the rewards.
Resolution	RESOLVED The initial buyback rate is changed to 48.45% and burn rate is changed to 20%.

Issue #09	Missing whenNotPaused modifier
Severity	LOW SEVERITY
Location	Lines 201-203
Description	The `farm` function is missing the `whenNotPaused` modifier.
Recommendation	Consider adding the `whenNotPaused` modifier with the function.
Resolution	ACKNOWLEDGED The pause is not applied so that the user is able to collect their rewards.

Issue #10	There should be a check that the contract is not paused before making a call to `_helptoEarn`
Severity	LOW SEVERITY
Location	Lines 224-257
Description	In `withdraw`, `if(!isEmergency){_helpToEarn();}` should also check that it is not paused before `helpToEarn` is called.
Recommendation	Consider adding a boolean flag which is set to true once the call is made to the `setRouter` functions. `if(!isEmergency && !paused()){_helpToEarn();}`
Resolution	ACKNOWLEDGED The pause is not applied as per the business logic.

Issue #11	Events should be emitted for functions that alter states
Severity	INFORMATIONAL
Location	Lines 96-146 Lines 180-204 Lines 224-257 Lines 359-386 Lines 388-399 Lines 413-442 Lines 452-459 Lines 461-468 Lines 470-477 Lines 479-483 Lines 485-489 Lines 491-499 Lines 501-504
Description	There are multiple functions which modify the underlying state of the contract. Events should be emitted for these functions.
Recommendation	Consider adding events to specify the states which are being modified. This will also help users to track the modifications.
Resolution	ACKNOWLEDGED

Issue #12	Several functions can be declared as `external`
Severity	INFORMATIONAL
Description	Following functions under the contract can be declared as external to avoid any kind of unwanted exposure. - `deposit` - `pause` - `setRouter0, setRouter1, & setRouter2` - `setGov` - `setBuybackRate` - `setBurnRate` - `inCaseTokensGetStuck` - `convertDustToEarned` - `farm` Apart from being a best practice when the function is not used within the contract, this can lead to lower gas usage in certain cases.
Recommendation	Consider marking the above functions as `external`.
Resolution	RESOLVED

2.4 MarsAutoFarm

MarsAutoFarm is the Masterchef for the Mars Protocol, which allows the owner to add pools based on a defined strategy. The users can deposit and withdraw the LP tokens to earn over the same. The contract also receives the rewards from the Auto Farm Contracts.

2.4.1 Privileged Roles

The following functions can be called by the owner:

- `add`

The following functions can be called by the native strategy:

- `chargePool`
- `updateLastEarnBlock`

2.4.2 Issues & Recommendations

Issue #13	Missing nonReentrant modifier
Severity	LOW SEVERITY
Location	Lines 80-97
Description	As the chargePool function makes an external call and states are being modified after the external calls, it is recommended to add a reentrancy guard, to be on the safe side.
Recommendation	Consider adding the nonReentrant modifier with the function definition.
Resolution	ACKNOWLEDGED

Issue #14	There should be a check for shares to be more than zero for overly small deposits
Severity	LOW SEVERITY
Location	Lines 117-144
Description	In deposit, there should be a check that sharesAdded > 0 to ensure that the user does not get 0 shares for overly small deposits.
Recommendation	Consider adding: `require(sharesAdded > 0, "received zero shares");`
Resolution	ACKNOWLEDGED

Issue #15	Several functions can be declared as `external`
Severity	INFORMATIONAL
Description	The following functions within the contract can be declared as external to avoid any kind of unwanted exposure. - `poolLastEarnBlock` - `deposit` - `withdraw` - `emergencyWithdraw` Apart from being a best practice when the function is not used within the contract, this can lead to lower gas usage in certain cases.
Recommendation	Consider changing them to external functions.
Resolution	ACKNOWLEDGED

Issue #16	Unnecessary parsing of `msg.sender` as `address`
Severity	INFORMATIONAL
Location	`Line 87`
Description	There are several locations in the code where `msg.sender` is parsed as `address`. This is unnecessary as `msg.sender` is already an `address`. The extra gas cost can be avoided.
Recommendation	Consider replacing all instances of `address(msg.sender)` with `msg.sender`.
Resolution	ACKNOWLEDGED

Issue #17	Fee-on-transfer tokens are not supported
Severity	INFORMATIONAL
Description	Currently, transfers do not support fee-on-transfer tokens.
Recommendation	Consider adding a before and after balance check for transfers to make sure that the balance is modified as per the amount specified. A comment or update can be added for the users to notify that the protocol is currently not supporting the same.
Resolution	ACKNOWLEDGED

2.5 MarsAutoFarmGovernance

Mars Auto Farm Governance is a contract which provides the feature to set up a proposal. The mars token amount provided for the proposal is burned and the governance token is transferred to the contract. The users can provide votes on the proposal to make the required changes to the protocol.

2.5.1 Issues & Recommendations

Issue #18	Use of non-standard packed mode
Severity	LOW SEVERITY
Location	`Lines 193-210`
Description	`abi.encodePacked` provides a non standard packed mode where types smaller than 32 bytes are not padded, dynamic types are encoded without the lengths and arrays are encoded in place. This type of encoding is avoided for function calls as it might lead to misleading or wrong calls.
Recommendation	Consider using `abi.encode` to package data for function calls.
Resolution	ACKNOWLEDGED

Issue #19	SafeMath is not used for arithmetic operations
Severity	LOW SEVERITY

Description	SafeMath is not being used within the MarsAutoFarmGovernance Contract. This may lead to overflows/underflows for arithmetic operations.
	
	For example under the function cast votes, it might lead to overflow with when dealing with huge number of votes:
	`if(support){`
	`  proposal.YES+=votes;`
	`}else{`
	`  proposal.NO+=votes;`
	`}`

Recommendation	Consider using SafeMath for arithmetic operations or upgrade the compiler to Solidity version >= 0.8.0.
Resolution	ACKNOWLEDGED

Issue #20	Several functions can be declared as external
Severity	INFORMATIONAL
Description	The following functions can be declared as external to avoid any kind of unwanted exposure. - `proposalsCount` - `getActions` - `state` Apart from being a best practice when the function is not used within the contract, this can lead to lower gas usage in certain cases.
Recommendation	Consider marking the above functions as external.
Resolution	ACKNOWLEDGED

Issue #21	Unnecessary parsing of `msg.sender` as `address`
Severity	INFORMATIONAL
Location	`Lines 136,142,220,255`
Description	There are several locations in the code where `msg.sender` is parsed as `address`. This is unnecessary as `msg.sender` is already an `address`. The extra gas cost can be avoided.
Recommendation	Consider replacing all instances of `address(msg.sender)` with `msg.sender`.
Resolution	ACKNOWLEDGED

PALADIN
BLOCKCHAIN SECURITY