CERTIK

Code Security Assessment

Stellaswap #2

Mar 16th, 2022

Table of Contents

Summary

This report has been prepared for Stellaswap to discover issues and vulnerabilities in the source code of the Stellaswap #2 project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

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.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	Stellaswap #2
Platform	Other
Language	Solidity
Codebase	https://github.com/stellaswap/core/commit/a20e85bc0bacbad189fc4fd8669e4c870f24e5cd
Commit	cfdfb469121c8cf1465362624bf35317cbd7f34

Audit Summary

Delivery Date	Mar 16, 2022 UTC
Audit Methodology	Static Analysis, Manual Review

Vulnerability Summary

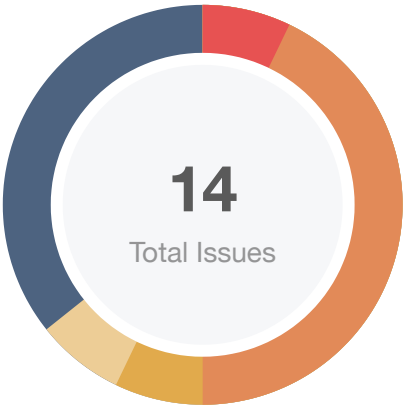
Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Mitigated	Resolved
Critical	1	0	0	0	0	0	1
Major	6	0	0	4	0	2	0
Medium	1	0	0	0	0	0	1
Minor	1	0	0	0	0	0	1
Informational	5	0	0	5	0	0	0
Discussion	0	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
MAT	amm/libraries/Math.sol	a553dd23aa798c18e1b2a19b2f64a2ba8144df56e212f20bab346be5c37287bb
SMB	amm/libraries/SafeMath.sol	7564e2cd86bfd4342b4b9ce0e734196fa15642876bc1a3515c519e955dae19ec
SSV	amm/libraries/StellaSwapV2Library.sol	05c0e1c775dd982e911fb117817f47a0e1c0a30df20d7bfadf6bfd16e73942e4
THB	amm/libraries/TransferHelper.sol	28b7bb5e8ac8fb0a3ccdaf50b85d2dbc22804d5b64d08f13924e94206ef823e1
UQB	amm/libraries/UQ112x112.sol	24283a562d299a5e4133d3f05304eec8e75a1b18c6907dd2a8f399eea0b16524
SSE	amm/StellaSwapV2ERC20.sol	b55ffba733a091220cedd0ee895de6226bb31d820fa74227829d13e898f2011a
SSF	amm/StellaSwapV2Factory.sol	2730ed9f7adac7fb204972dc18e88babda05b9dd2eca494589290d46835a986e
SSP	amm/StellaSwapV2Pair.sol	147410092df2de991d512a43826c189f8e7764d99c76b0069305714dd51ce512
SSR	amm/StellaSwapV2Router.sol	3c47554385d5376cdbdd2eedf5beb83172f5930263176ef474c56396b00c10a5
SVR	amm/StellaSwapV2Router02.sol	69dd906f7f6111b3f571cb46d62c17a73bd446ec038e0471b9fdf450ed56001a
SDB	farms/StellaDistributor.sol	0e9370ef0d6e0e36899b063028201009c1a646012518b9c201350e85b6a5a0ba
FOW	forwarder/Forwarder.sol	4af0b1f8414ddde7ef9f9e4b3c604352daed1d0496be85a96faa22f6f434c87c
IFB	forwarder/IForwarder.sol	3294fe5789375da2ddb8a5583db8a551e560188ed9e9a447c2e4f597b817440b
EIP	gasless/EIP712Base.sol	aeed0cc740df31f9a4924ce7317c21ca339c7e1af665de2cd569116011628bfa
EIM	gasless/EIP712MetaTransaction.sol	7829192129de9feb36941fc48caada22c4d0a91b95fc2bff1bbdc1e125164895
GSB	gasless/GasSwap.sol	b5970f3cbf5f30e91a02dc3a78701770180ac86ae8a77193716073545fad139d
ISR	gasless/IStellaRouter.sol	d4e40dde6a711cd62fb39f0bda2db151e777472af888df101121a3590ed427e0
ITB	gasless/IToken.sol	4fa3559518641fcb37deecafd6cc3905a249174d684cf874f7161654fb2cdee2
MRB	gasless/MockRouter.sol	bffeae13ffc87f4134f00ebd28f774a3fa869f001e913a50c2f2468c3cefec2b
MUL	helpers/Multicall.sol	b2b0206d463529d8604c2da268b467f4c3672f6a0c1403a4caf25d0f3fa2a7c3
TIM	helpers/Timelock.sol	6a8d0738bf841b96de3ae4e14ad616aa1b212d54c009dbf417997b8b36a4cacd
STE	token/Stella.sol	0bc750d932845f72496b7f6439a16012d1bc7fa9346f4bbf282b9edb1222867b

ID	File	SHA256 Checksum
ISE	utils/IStellaERC20.sol	fc18f0df2b25b3027695ea417661fa9456fa5894f1a1aa03268cee6f2cf786ab
MER	utils/MockERC20.sol	7deae230cb15359254bff662a33df1f170942a9b07ef2b61c29cce3bf57426cf
SVB	vault/StellaVault.sol	67d228ec3ae99eafc3c214524e836a7876d67fb466bf32e74c5cda6c47231a6b

Findings



Critical	**1** (7.14%)
Major	**6** (42.86%)
Medium	**1** (7.14%)
Minor	**1** (7.14%)
Informational	**5** (35.71%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
A20-01	Unlocked Compiler Version	Language Specific	Informational	Acknowledged
A20-02	SafeMath Not Used	Mathematical Operations	Informational	Acknowledged
A20-03	Variables That Could Be Declared as `constant`	Gas Optimization	Informational	Acknowledged
AMM-01	Centralization Risk in AMM	**Centralization / Privilege**	**Major**	Acknowledged
GSB-01	Centralization Risk in GasSwap.sol	**Centralization / Privilege**	**Major**	Acknowledged
GSB-02	Potential Reentrancy Attack	Logical Issue	Minor	Resolved
SDB-01	Inappropriate Upper Limits for Fees	Logical Issue	Medium	Resolved
SDB-02	Centralization Risk in StellaDistributor.sol	**Centralization / Privilege**	**Major**	Mitigated
SSE-01	Missing Error Messages	Coding Style	Informational	Acknowledged
SSP-01	Unknown Implementation of `migrator.desiredLiquidity()`	**Centralization / Privilege**	**Major**	Acknowledged
STE-01	Initial Token Distribution	**Centralization / Privilege**	**Major**	Mitigated

ID	Title	Category	Severity	Status
SVB-01	Function emergencyWithdraw() allows user to bypass the lockdown duration check	Logical Issue	Critical	Resolved
SVB-02	Missing Emit Events	Coding Style	Informational	Acknowledged
SVB-03	Centralization Risk in StellaVault.sol	**Centralization / Privilege**	**Major**	Acknowledged

A20-01 | Unlocked Compiler Version

Category	Severity	Location	Status
Language Specific	Informational	token/Stella.sol: 2 vault/StellaVault.sol: 2 forwarder/Forwarder.sol: 2	Acknowledged

Description

The contract has unlocked compiler version. An unlocked compiler version in the source code of the contract 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 different compiler versions. This can lead to an ambiguity when debugging as compiler specific bugs may occur in the codebase that would be hard to identify over a span of multiple compiler versions rather than a specific one.

Recommendation

We advise that the compiler version is instead locked at the lowest version possible that the contract can be compiled at. For example, for version `v0.8.0` the contract should contain the following line:

```
pragma solidity 0.8.0;
```

A20-02 | SafeMath Not Used

Category	Severity	Location	Status
Mathematical Operations	Informational	farms/StellaDistributor.sol: 421~424 vault/StellaVault.sol: 433~436	Acknowledged

Description

SafeMath from OpenZeppelin is not used in the following lines which makes them possible for underflow and will lead to an inaccurate calculation result.

```
421         pool.accStellaPerShare =
422             pool.accStellaPerShare +
423             (((stellaReward * 1e12) / pool.totalLp) * lpPercent) /
424             1000;
```

Recommendation

We advise the client to use OpenZeppelin's SafeMath library for all of the mathematical operations.

Reference: https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/utils/math/SafeMath.sol

A20-03 | Variables That Could Be Declared As `constant`

Category	Severity	Location	Status
Gas Optimization	Informational	token/Stella.sol: 13, 14 gasless/GasSwap.sol: 11	Acknowledged

Description

The linked variables could be declared as `constant` since these state variables are never modified.

Recommendation

We recommend to declare these variables as `constant`.

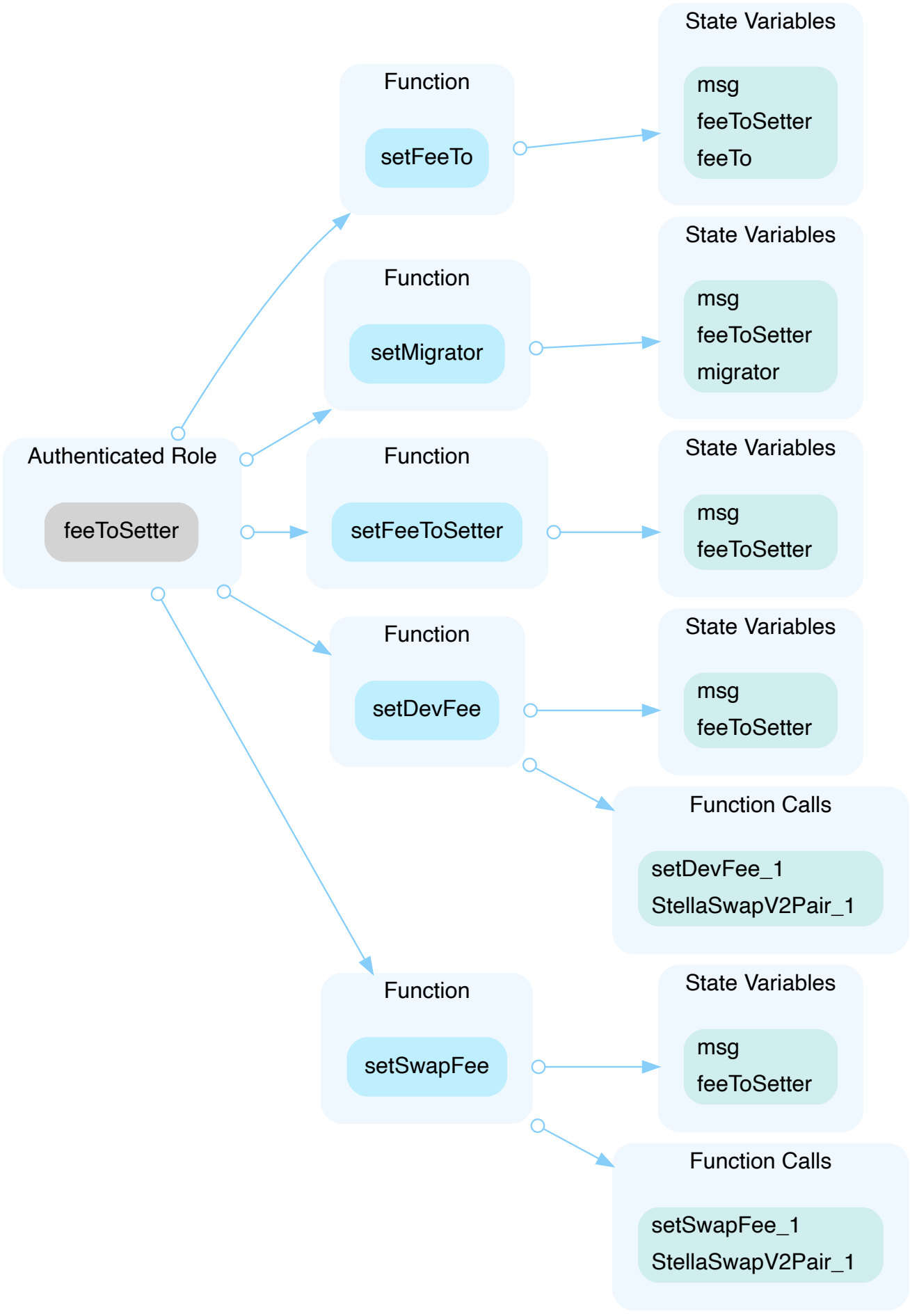
AMM-01 | Centralization Risk In AMM

Category	Severity	Location	Status
Centralization / Privilege	Major	amm/StellaSwapV2Factory.sol: 48~51, 53~56, 58~61, 63~67, 69~72 amm/StellaSwapV2Pair.sol: 133~161, 75~93	Acknowledged

Description

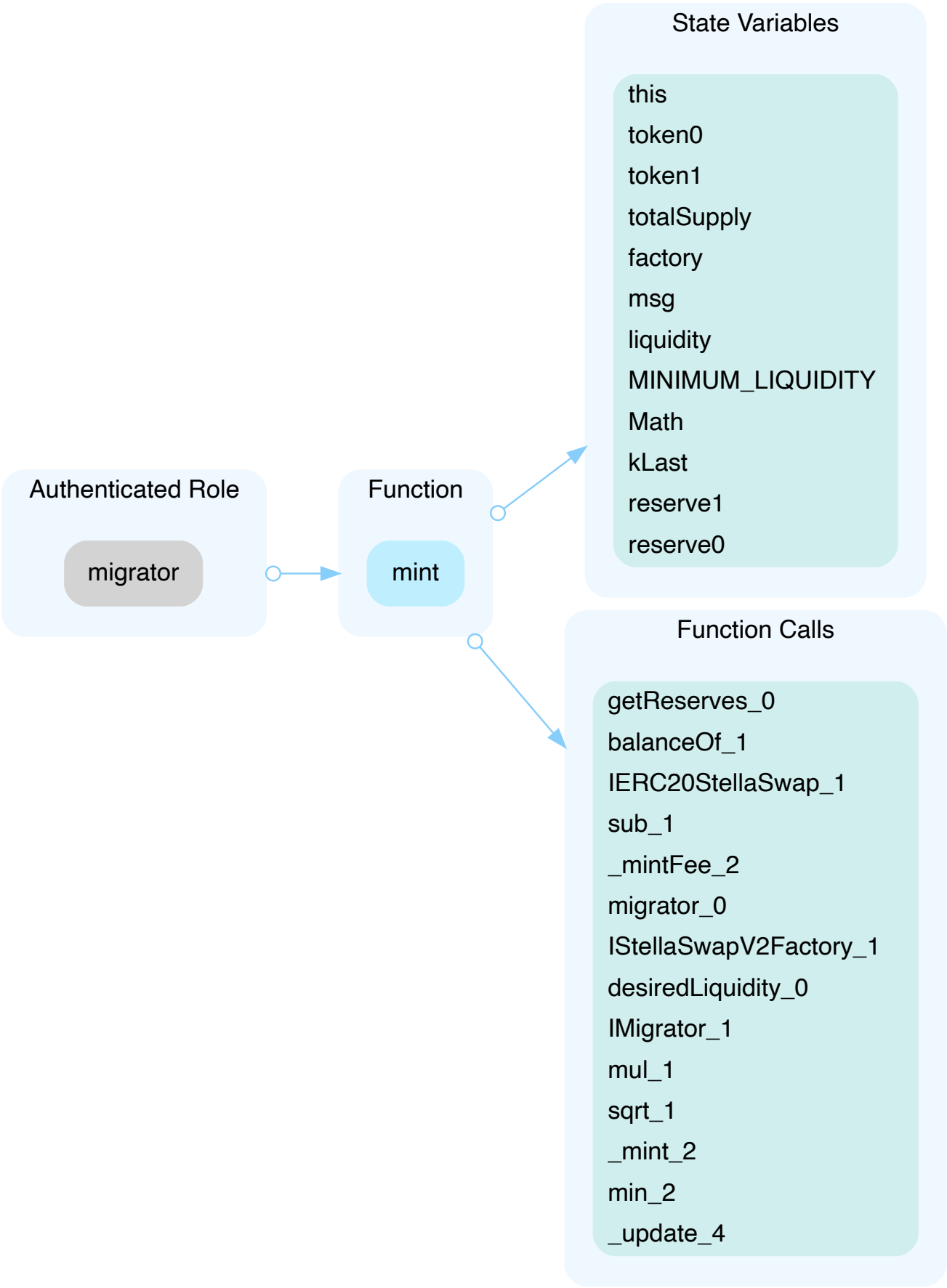
In the contract `StellaSwapV2Factory` the role `feeToSetter` has authority over the functions shown in the diagram below.

Any compromise to the `feeToSetter` account may allow the hacker to take advantage of this authority and set the swap fee to 100% or set the `Migrator` address to a malicious contract, causing loss or stolen of uses' asset.


Authenticated Role
feeToSetter
Function
setFeeTo
State Variables
msg
feeToSetter
feeTo
Function
setMigrator
State Variables
msg
feeToSetter
migrator
Function
setFeeToSetter
State Variables
msg
feeToSetter
Function
setDevFee
State Variables
msg
feeToSetter
Function Calls
setDevFee_1
StellaSwapV2Pair_1
Function
setSwapFee
State Variables
msg
feeToSetter
Function Calls
setSwapFee_1
StellaSwapV2Pair_1

In the contract `StellaSwapV2Pair` the role `migrator` has authority over the functions shown in the diagram below.

Any compromise to the `migrator` account may allow the hacker to take advantage of this authority and disrupt the initial liquidity offering, which might damage the project tokenomics.


State Variables
this
token0
token1
totalSupply
factory
msg
liquidity
MINIMUM_LIQUIDITY
Math
kLast
reserve1
reserve0
Authenticated Role
migrator
Function
mint
Function Calls
getReserves_0
balanceOf_1
IERC20StellaSwap_1
sub_1
_mintFee_2
migrator_0
IStellaSwapV2Factory_1
desiredLiquidity_0
IMigrator_1
mul_1
sqrt_1
_mint_2
min_2
_update_4

Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

Alleviation

StellaSwap Team:

feeToSetter moved behind Timelock:

https://moonbeam.moonscan.io/tx/0xdd04ef68aafd9719b9121ea29a55232255433b29ccfd2bb9e14e895c16b5b9ed

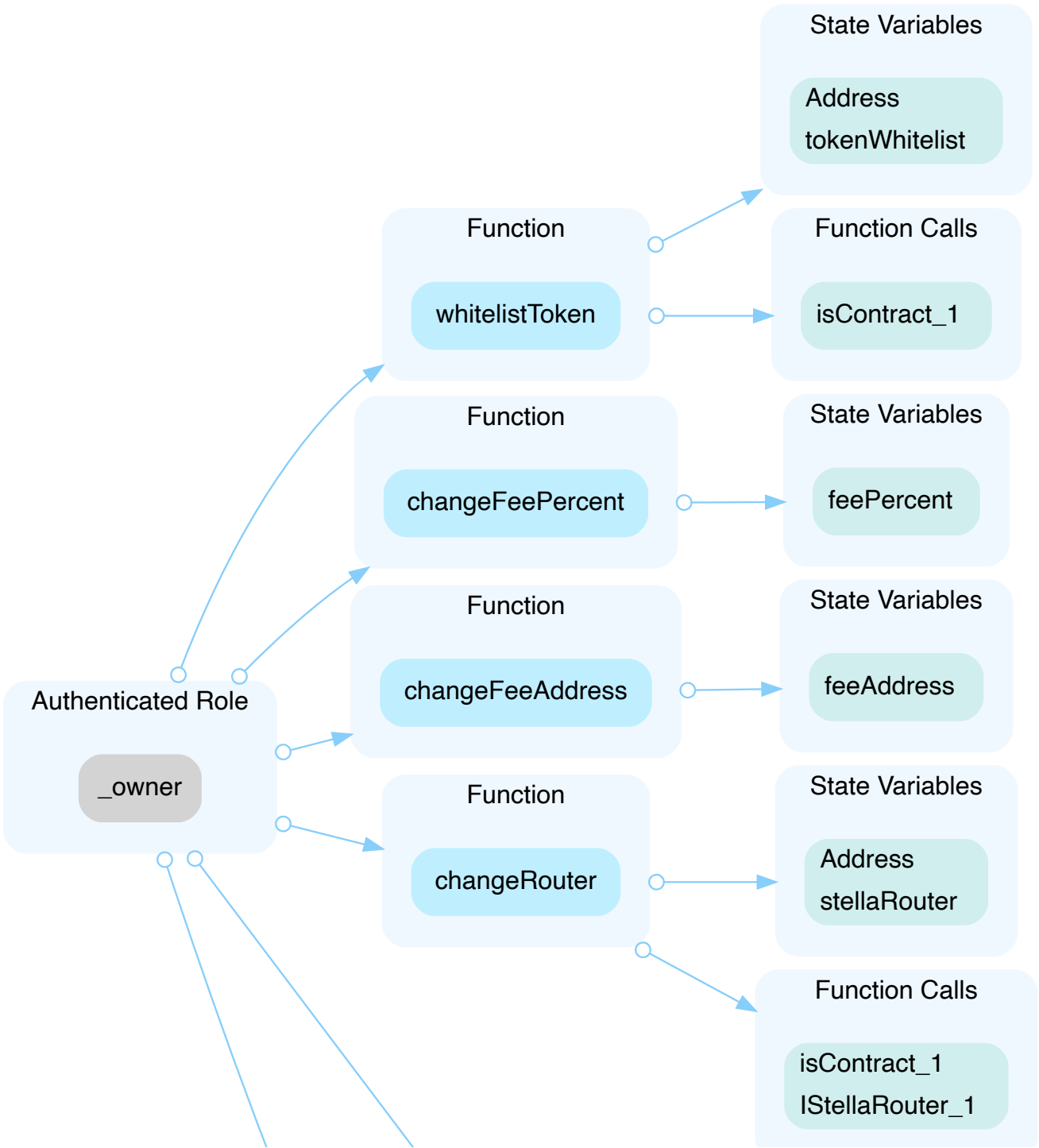
GSB-01 | Centralization Risk In GasSwap.sol

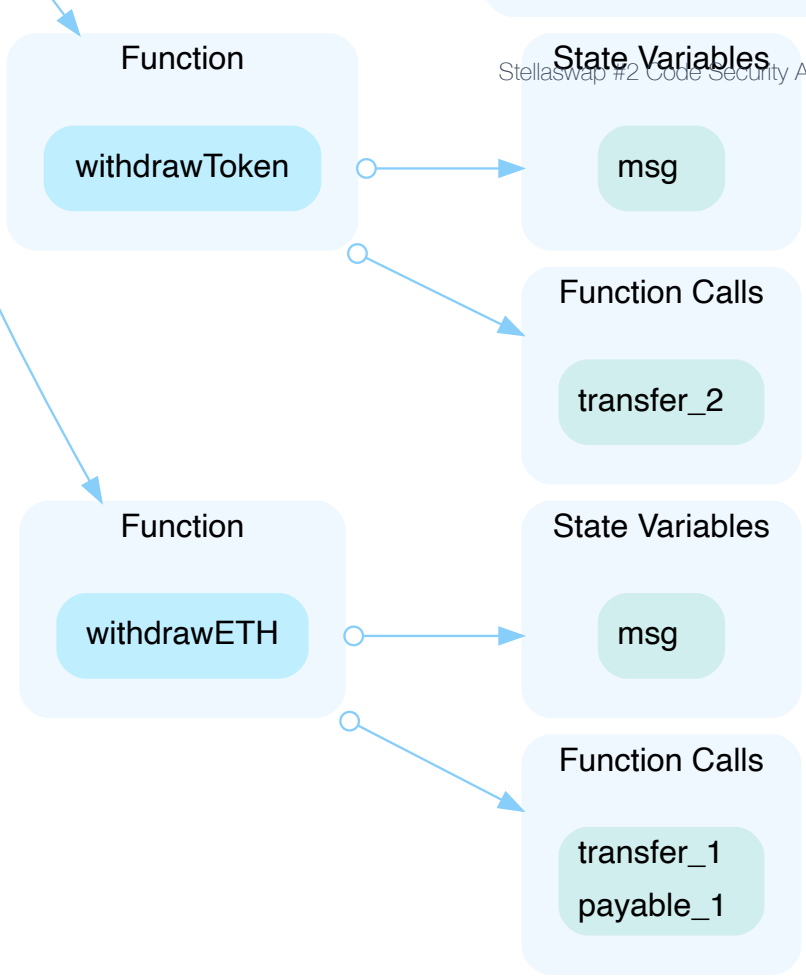
Category	Severity	Location	Status
Centralization / Privilege	Major	gasless/GasSwap.sol: 32~38, 40~43, 45~47, 49~52, 54~56, 59~61	Acknowledged

Description

In the contract `GasSwap` the role `_owner` has authority over the functions shown in the diagram below.

Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and modify critical settings in the `GasSwap` contract.





Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND

- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

Alleviation

StellaSwap Team:

We currently haven't deployed GasSwap contract and are planning not to.

The reason for this contract was to create "SwapForGas" feature where people that bridged into network can get gas.

We've partnered with Biconomy to provide gas-less transactions and will be using them: https://stellaswap.medium.com/stellaswap-partners-with-biconomy-for-gasless-transactions-on-moonbeam-2da760a5f6b5

GSB-02 | Potential Reentrancy Attack

Category	Severity	Location	Status
Logical Issue	Minor	gasless/GasSwap.sol: 64~129	Resolved

Description

A reentrancy attack can occur when the contract creates a function that makes an external call to another untrusted contract before resolving any effects.

If the attacker can control the untrusted contract, they can make a recursive call back to the original function, repeating interactions that would have otherwise not run after the external call resolved the effects.

```
1    payable(msgSender()).transfer(amount);
```

If the GLMR receiver is a contract, he can perform recursive callbacks in his `receive()` function. Such unexpected recursive callbacks may disrupt the operation of the `GasSwap` contract in some cases.

Recommendation

We recommend applying OpenZeppelin ReentrancyGuard library - `nonReentrant` modifier for the aforementioned functions to prevent reentrancy attack.

Alleviation

The team heeded our advice and updated the code in commit 5cfdfb469121c8cf1465362624bf35317cbd7f34

SDB-01 | Inappropriate Upper Limits For Fees

Category	Severity	Location	Status
Logical Issue	Medium	farms/StellaDistributor.sol: 163~178	Resolved

Description

The current upper limit for fee percent can be set as high as 100%, which is not a reasonable value.

```
function setTeamPercent(uint256 _newTeamPercent) public onlyOwner {
      require(
          0 <= _newTeamPercent && _newTeamPercent <= 1000,
          "set team percent: invalid percent value"
      );
      require(
          treasuryPercent + _newTeamPercent + investorPercent <= 1000,
          "set team percent: total percent over max"
      );
      emit SetTeamPercent(teamPercent, _newTeamPercent);
      teamPercent = _newTeamPercent;
  }
```

Recommendation

We recommend adding a upper limit for total fee and set it to an appropriate value such as 10%.

Alleviation

The team heeded our advice and updated the code in commit [a54b5a67007eb041bb8b445c698af1e8c0f5f439](a54b5a67007eb041bb8b445c698af1e8c0f5f439).

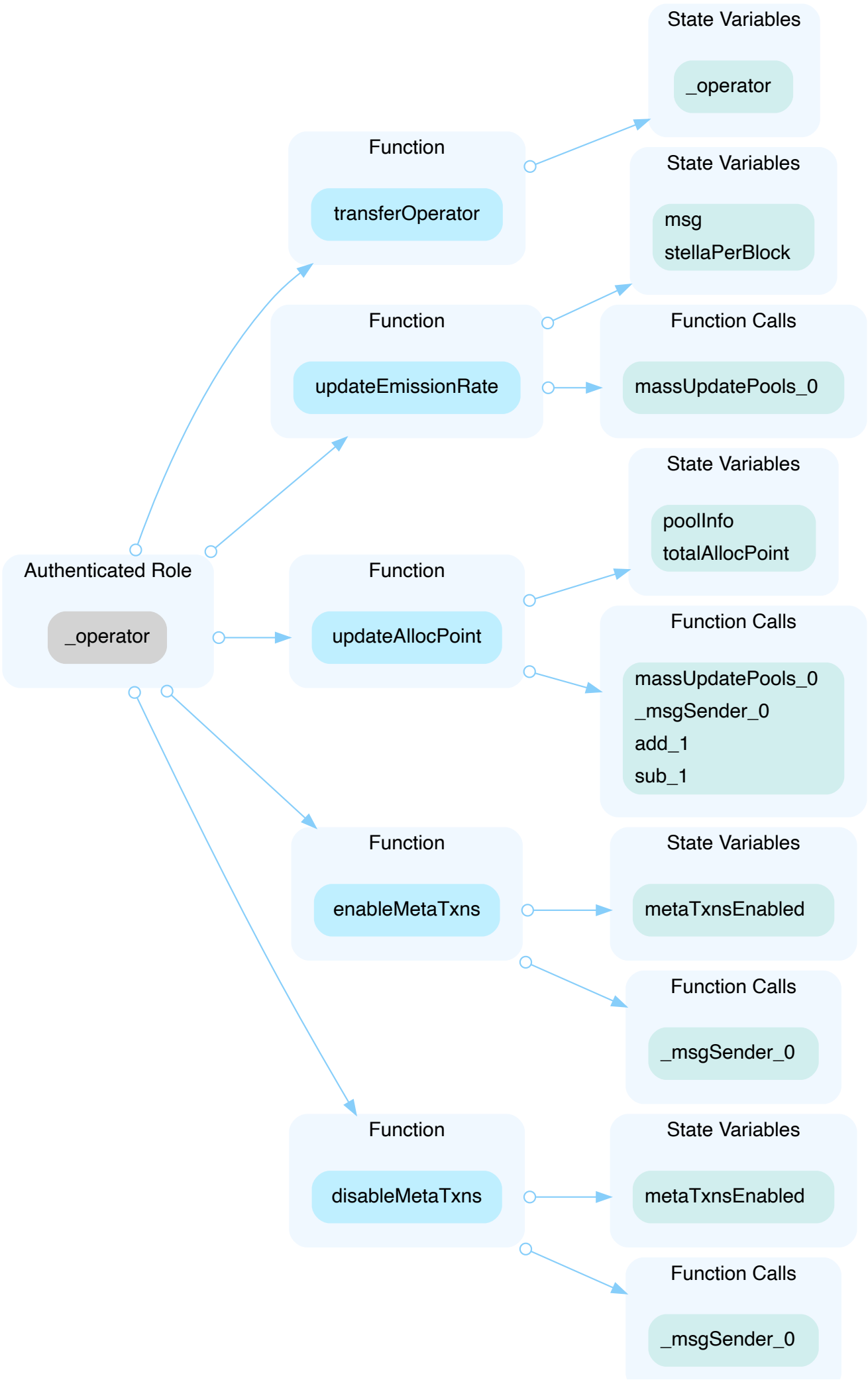
SDB-02 | Centralization Risk In StellaDistributor.sol

Category	Severity	Location	Status
Centralization / Privilege	Major	farms/StellaDistributor.sol: 251~258, 575~580, 582~601, 604~609, 612~617 , 261~271, 279~312, 315~339, 636~647, 656~667, 679~690, 670~677, 627 ~634, 650~654	Mitigated

Description

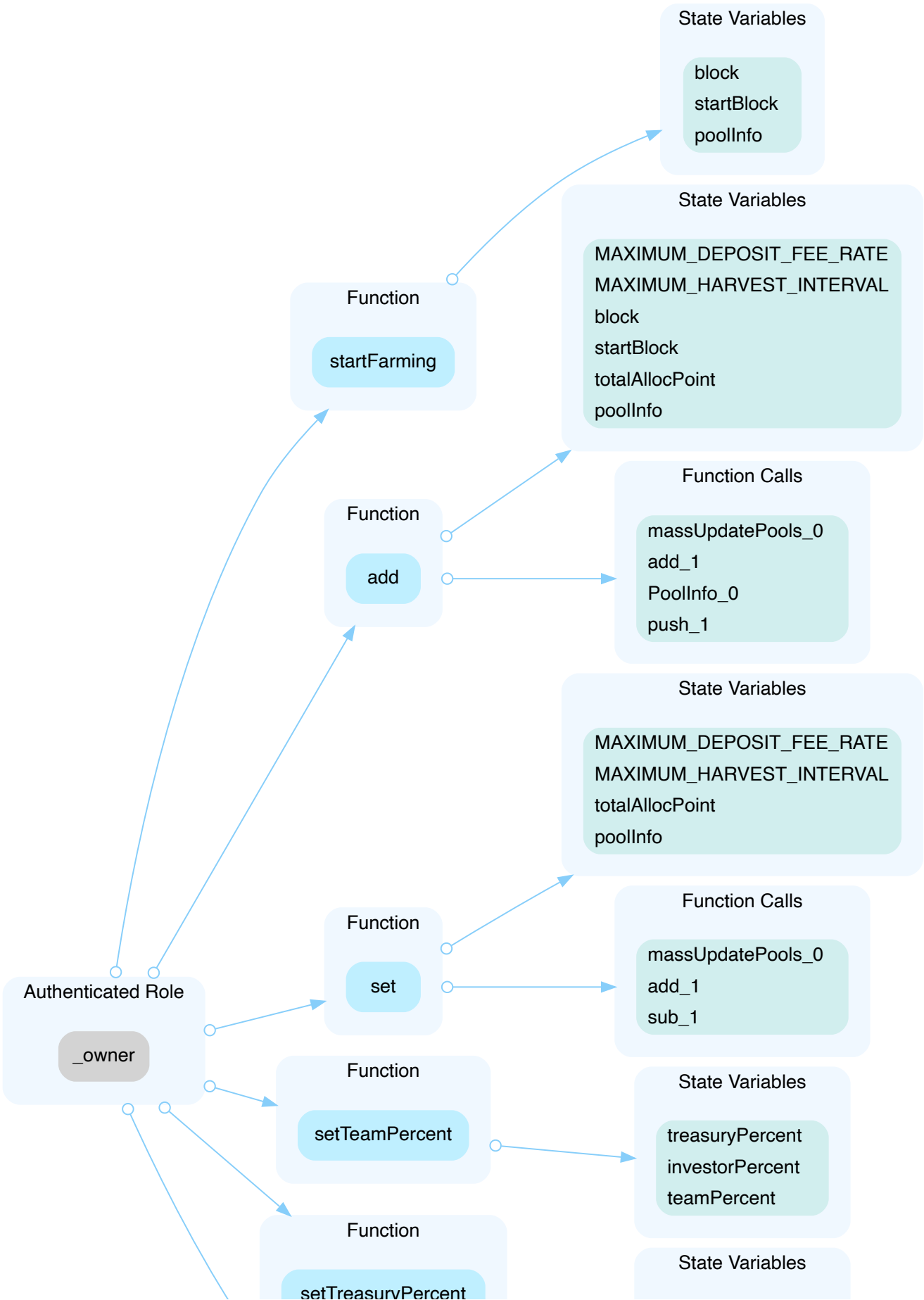
In the contract `StellaDistributor` the role `_operator` has authority over the functions shown in the diagram below.

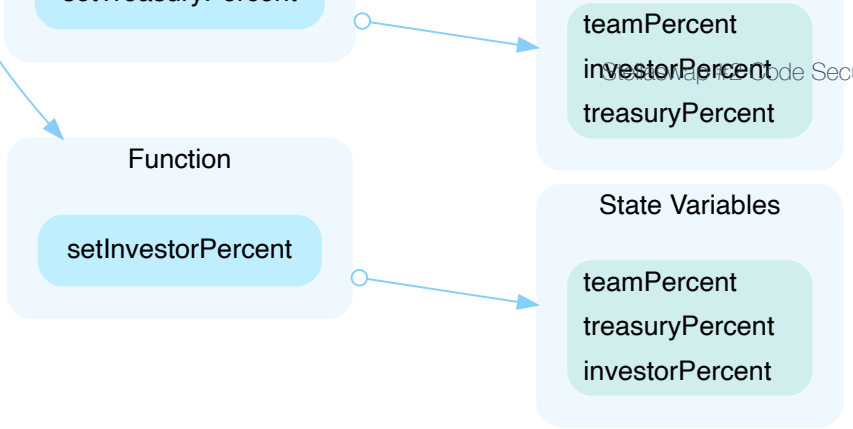
Any compromise to the `_operator` account may allow the hacker to take advantage of this authority and disrupt the operation of the `Distributor` contract.


State Variables
_operator
Function
transferOperator
State Variables
msg
stellaPerBlock
Function
updateEmissionRate
Function Calls
massUpdatePools_0
State Variables
poolInfo
totalAllocPoint
Authenticated Role
_operator
Function
updateAllocPoint
Function Calls
massUpdatePools_0
_msgSender_0
add_1
sub_1
Function
enableMetaTxns
State Variables
metaTxnsEnabled
Function Calls
_msgSender_0
Function
disableMetaTxns
State Variables
metaTxnsEnabled
Function Calls
_msgSender_0

In the contract `StellaDistributor` the role `_owner` has authority over the functions shown in the diagram below.

Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and disrupt the contract operation. e.g. Adding a new pool with extremely high allocpoints, setting the fees to 100%, or setting the reward lock time to as long as 90 days.





Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND

- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

Alleviation

StellaSwap Team:

StellaDistributor is now behind Timelock, moved ownership and Operator:

Ownership Transferred:
https://moonbeam.moonscan.io/tx/0x53becc98f9efe1916df38b9dde1a004b10239612e5091d74c79eb14b9b67dd8e

Operatorship Transferred:
https://moonbeam.moonscan.io/tx/0x5c8e5be56f29ab6248106694719e26e63412b63e72d58b64fb527998f857df8e

SSE-01 | Missing Error Messages

Category	Severity	Location	Status
Coding Style	Informational	amm/StellaSwapV2ERC20.sol: 76	Acknowledged

Description

The function call will revert when there is an inadequate allowance. It is better to provide a string message containing details about the error that will be passed back to the caller.

```
1   function transferFrom(address from, address to, uint value) external returns (bool)
{
2         if (allowance[from][msg.sender] != uint(-1)) {
3             allowance[from][msg.sender] = allowance[from][msg.sender].sub(value);
4         }
5         _transfer(from, to, value);
6         return true;
7     }
```

Recommendation

We recommend providing reasonable error message for the linked code.

SSP-01 | Unknown Implementation Of `migrator.desiredLiquidity()`

Category	Severity	Location	Status
Centralization / Privilege	**Major**	amm/StellaSwapV2Pair.sol: 145	Acknowledged

Description

`setMigrator()` function in `StellaSwapV2Factory` can set migrator contract to any address that is implemented from `IMigrator` interface by the owner. As result, invocation of `migrator.desiredLiquidity()` in function mint() may bring dangerous effects as it is unknown to the user.

The scope of the audit treats `Migrator` contract as black boxes and assumes their functional correctness.

However, in the real world, `Migrator` can be compromised and the contract controller can set arbitrary amounts of `desiredLiquidity`, which may lead to lost or stolen assets.

Recommendation

`Migrator` contract is out of the audit scope. We encourage the team to constantly monitor the statuses of the `Migrator` contract and ensure its security and functionality correctness.

Alleviation

[StellaSwap Team]:

This function was inherited from fork of SushiSwap's AMM. This method allows users to migrate their LP on other DEX. This function is only triggered when the Pair is new and no liquidity has been locked.

After moving Factory to Timelock, we will constantly monitor state of contract using Openzepplin's Defender Sentinel.

This should not affect initiliazed liquidity pairs.

STE-01 | Initial Token Distribution

Category	Severity	Location	Status
Centralization / Privilege	**Major**	token/Stella.sol: 20	Mitigated

Description

All of the `Stella` tokens are sent to the contract deployer when deploying the contract. This could be a centralization risk as the deployer can distribute `Stella` tokens without obtaining the consensus of the community.

Recommendation

We recommend the team be transparent regarding the initial token distribution process, and the team shall make enough efforts to restrict the access of the private key.

Alleviation

StellaSwap Team:

100k tokens were initial minting for liquidity lock. The rest tokens has been moved to locker: 0x8995066b7F1FB3Abe3c88040b677d03d607A0b58

The purpose of these tokens are Protocol Controlled Value (Treasury) and are meant to be used for VC fund raise, airdrops, advisors vesting, marketing and other stuff that we require tokens for.

SVB-01 | Function EmergencyWithdraw() Allows User To Bypass The Lockdown Duration Check

Category	Severity	Location	Status
Logical Issue	Critical	vault/StellaVault.sol: 515~537	Resolved

Description

The `emergencyWithdraw` function is not disabled in the vault contract, which allows users to bypass the lockdown duration check in this contract. Users can withdraw their funds at any time with the `emergencyWithdraw` function.

This implementation conflicts with the official document:

Users can stake their STELLA in Booster Vaults, which are single-asset STELLA-only pools. STELLA holders can lock their tokens into Booster vaults to generate higher APYs, with a longer time duration equating to a greater yield rate. These vaults are time-locked, meaning that users can lock their STELLA across a range of time horizons (e.g. 1w, 1m or 1y). Once locked, the tokens cannot be withdrawn until the timelock finishes.

Recommendation

We recommend the removing function `emergencyWithdraw` from the codebase.

Alleviation

Function `emergencyWithdraw()` was removed from code base in commit [491766576f843d0120050311e7b6b8943d539558]

SVB-02 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	vault/StellaVault.sol: 269~279	Acknowledged

Description

There should always be events emitted in the sensitive functions that are controlled by centralization roles.

Recommendation

It is recommended emitting events for the sensitive functions that are controlled by centralization roles.

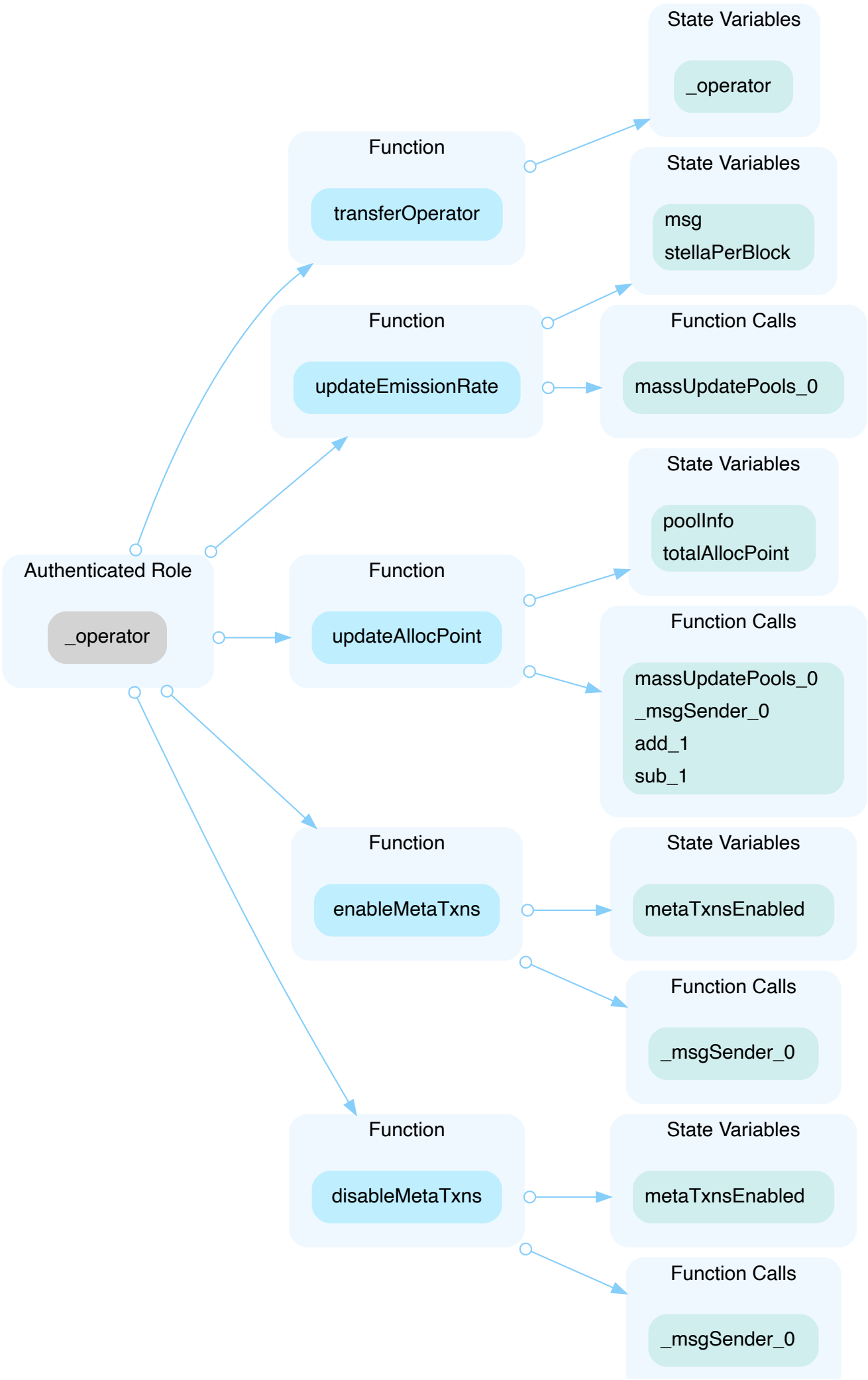
SVB-03 | Centralization Risk In StellaVault.sol

Category	Severity	Location	Status
Centralization / Privilege	Major	vault/StellaVault.sol: 259~266, 607~612, 614~633, 636~641, 644~649, 269~279, 287~322, 325~351, 668~679, 688~699, 711~722, 702~709, 659~666, 682~686	Acknowledged

Description

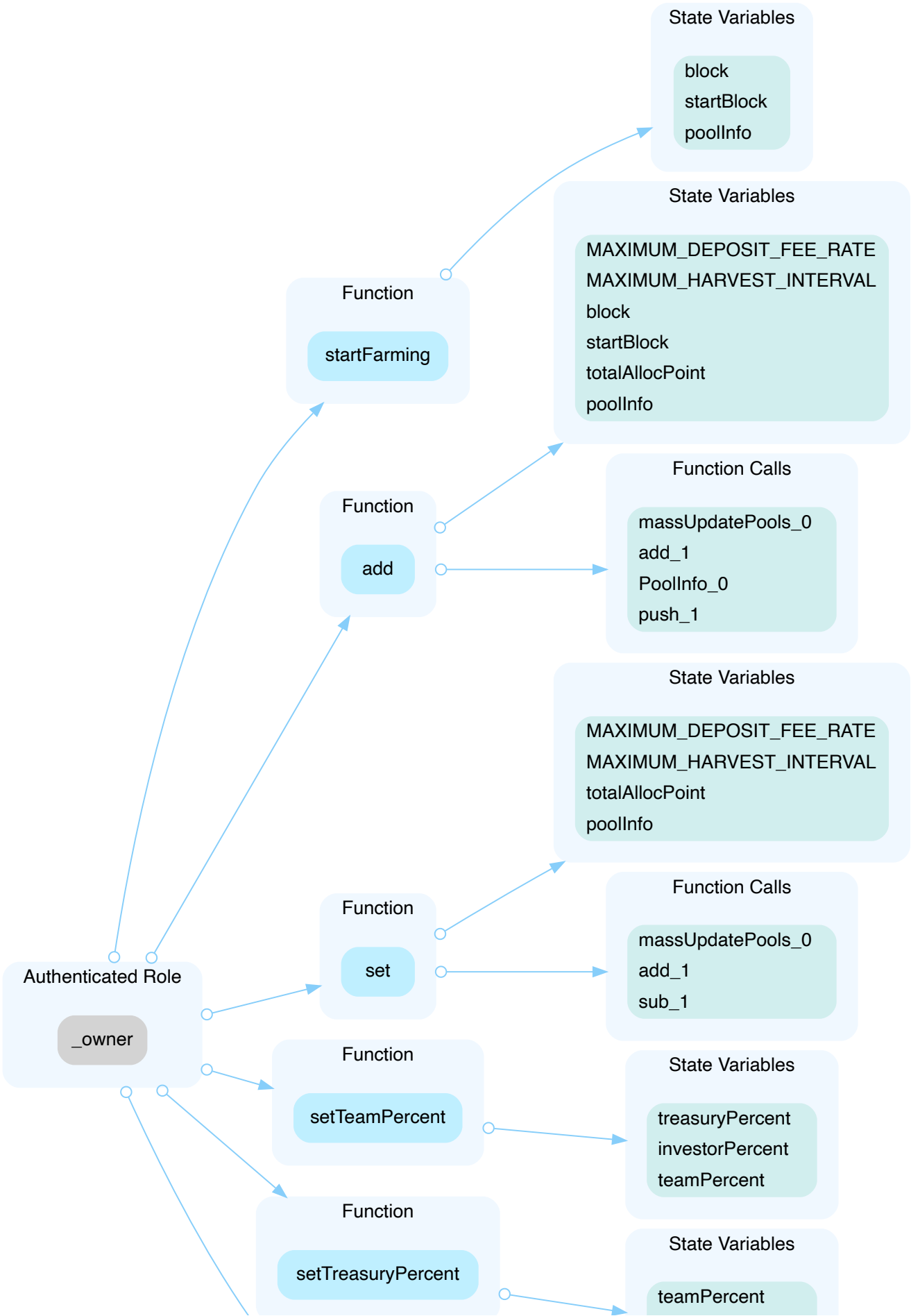
In the contract `StellaVault` the role `_operator` has authority over the functions shown in the diagram below.

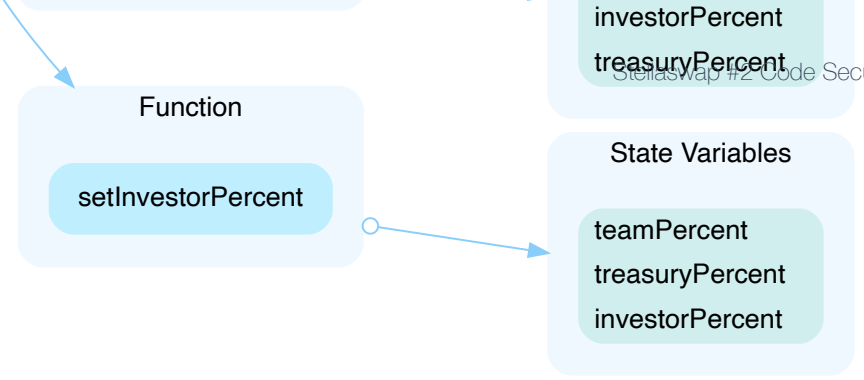
Any compromise to the `_operator` account may allow the hacker to take advantage of this authority and modify critical settings in the Vault contract.


State Variables
_operator
Function
transferOperator
State Variables
msg
stellaPerBlock
Function
updateEmissionRate
Function Calls
massUpdatePools_0
State Variables
poolInfo
totalAllocPoint
Authenticated Role
_operator
Function
updateAllocPoint
Function Calls
massUpdatePools_0
_msgSender_0
add_1
sub_1
Function
enableMetaTxns
State Variables
metaTxnsEnabled
Function Calls
_msgSender_0
Function
disableMetaTxns
State Variables
metaTxnsEnabled
Function Calls
_msgSender_0

In the contract `StellaVault` the role `_owner` has authority over the functions shown in the diagram below.

Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and disrupt the contract operation. e.g. Adding a new pool with extremely high allocpoints or setting the fees to 100%.





Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets.

Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

Alleviation

StellaSwap Team:

Operator to Timelock:
https://moonbeam.moonscan.io/tx/0x87324bec78046327fb2ff871256b9f51d0f637841a64702f449a4fbde11c215d

Owner to Timelock:
https://moonbeam.moonscan.io/tx/0xc78efbda28f6a60120ec52862966390a7c320c414864eea0fc898239f905b0d1

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Mathematical Operations

Mathematical Operation findings relate to mishandling of math formulas, such as overflows, incorrect operations etc.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

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 CertiK 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. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK'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 CertiK 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.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS

AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING

MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.