Creder (Stan)_Goldstation V3 - audit

Security Assessment

CertiK Assessed on Dec 12th, 2024

CertiK Assessed on Dec 12th, 2024

Creder (Stan)_Goldstation V3 - audit

The security assessment was prepared by CertiK, the leader in Web3.0 security.

Executive Summary

TYPES	ECOSYSTEM	METHODS
DEX	EVM Compatible	Formal Verification, Manual Review, Static Analysis

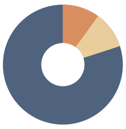
LANGUAGE	TIMELINE	KEY COMPONENTS
Solidity	Delivered on 12/12/2024	N/A

CODEBASE	COMMITS
source View All in Codebase Page	d6ccccea72da027c21ac01518aeaa2e973e2c414 View All in Codebase Page

Highlighted Centralization Risks

ⓘ Has blacklist/whitelist

Vulnerability Summary



10	1	0	0	9	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined

🟥 0	Critical		Critical risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
🟧 1	Major	1 Acknowledged	Major risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
🟨 0	Medium		Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
🟨 1	Minor	1 Acknowledged	Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.
🟦 8	Informational	1 Resolved, 7 Acknowledged	Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

TABLE OF CONTENTS | CREDER (STAN)_GOLDSTATION V3 - AUDIT

CODEBASE | CREDER (STAN)_GOLDSTATION V3 - AUDIT

Repository

source

Commit

d6ccccea72da027c21ac01518aeaa2e973e2c414

AUDIT SCOPE | CREDER (STAN)_GOLDSTATION V3 - AUDIT

69 files audited ● 33 files with Acknowledged findings ● 36 files without findings

ID	Repo	File	SHA256 Checksum
● PIS	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/base/PeripheryImmutableState.sol	f4611f54f13d0599648bf88fc5bba7fe8eb3bfc27f898c5cc0e2f27272ebca99
● PPG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/base/PeripheryPayments.sol	68cef83e01906a13f4a2bb1c12a9e99fad3e957eea6ddbb54bac30ba3b06a436
● PPW	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/base/PeripheryPaymentsWithFee.sol	4283f11d5dbd878b594cf4f99a8a0c13720d6dff98ee9bfbb63503391beeefc8
● PIG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/base/PoolInitializer.sol	ed0d234b15dab205f874522cc4c76761b584ecdebd89a45cdf1edb3d5e84ab88
● QGV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/lens/Quoter.sol	959673237ae3ada70936a072cffc425da4bb0200039bf3d95772dd147f1f9ca7
● QVG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/lens/QuoterV2.sol	ec3d92b99c6c195a8db21438309707102d6ee04a1337045e123ab5964df2689a
● CVG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/CallbackValidation.sol	c555690f8951945669c83eb7b788f1ff46943f2005b0766ff1241477f5236c05
● LAG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/LiquidityAmounts.sol	e4f117e062a91aec06bb03188c6f64f4442361b7a8550c2e892674e479074426
● PAG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/PoolAddress.sol	7f31f738e87f69d5b56ce80b3b3719abafdeb56821b6d6a9e0eb4f0f6c09996d
● SPM	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/SqrtPriceMathPartial.sol	8991b99be4675b8a746ea24e6309177e5884e0d41974885a9417f594954d3878
● THG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/TransferHelper.sol	7d02f695d41542209c5aa2b18b4041b53412e494491b3c44a361828261c366fd
● NFT	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/NFTDescriptorEx.sol	e8bb6a51756bec2e91b7b9bbd0a69cf4b261f5d9e17d1c7c92f58e36b094d3d7
● NPM	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/NonfungiblePositionManager.sol	a30fc587961f107b3e95467f173024cb50a71b4e9558445e895ad9505c5979ec

ID	Repo	File	SHA256 Checksum
NTP	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/NonfungibleTokenPositionDescriptor.sol	09b6bf612671dd2c405e96589846aff1bdec524f81b6c83546535aba1555b61a
SRG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/SwapRouter.sol	f96d3d9e0ff7e1ce3a936779508c780f220f7547446295cce078539ff4b83ec7
VMG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/V3Migrator.sol	397742b6985ff88f3334ba8768425fbb14c6490a13ab08091b24fe40cb3052f3
BMG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/BitMath.sol	32f71ea9156f55572a72efb0b2a913df88de66ff33d042043fb3e51a6050a557
FMG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/FullMath.sol	0a18f00afc2b99b3226898303319bf0a9108ace44c8871491571f53de2f0bf0d
LGS	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/LowGasSafeMath.sol	394107ff2dbbaded5612452af5e77b4af9d0871b096c1514b0ea659b862fc46f
SCG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/SafeCast.sol	9aed494b56d3dd16b7d6535583ded2cdfb03dc80aaa919347b13d35fd597e8bf
SPV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/SqrtPriceMath.sol	36eeb343e0b1809cd76b2ec72a336923aa24f857965966543b065e660b2ebc6e
TBG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/TickBitmap.sol	8452c484e6caad95411358d8c1763810e715b3d13697c83665657619472d3b0a
THV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/TransferHelper.sol	ccb87429b290eb6ed429648a7131f68ce0151a74f3ed27de78aacd28015e4590
PVF	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/PancakeV3Factory.sol	6f4364c4b9761586f7b6eb71bf2344485e12eb01851419da4c1c81ad266d2a00
PVP	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/PancakeV3Pool.sol	37a20491a5cd229d78ccfb1e79f15f7b65e85beac8c17cda76ffc9c2302671d7
PVD	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/PancakeV3PoolDeployer.sol	2189a9d27ee5726c9a863b8b0576055bb13356676264638709ffddf7d6077fad
PVL	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-LM-Pool/PancakeV3LmPool.sol	cbe90c2a216055fdaf106099c5ff2f4d5d1099dcd026d3bdc813d5254144bfbd
PVM	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-LM-Pool/PancakeV3LmPoolDeployer.sol	12df2d49c9adc9c778f39106cbd1a32e332117ec3ad59a2f83b6481cc4b0ec38

ID	Repo	File	SHA256 Checksum
● MCK	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-MasterChefV3/keeper/MasterChefV3KeeperV1.sol	8246b675b5a8716181902f6d2abc30f620f8f6c7e92d713377f8691399155cd3
● MCG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-MasterChefV3/keeper/MasterChefV3KeeperV2.sol	fb613e852e82a197ea5b229381e56c1ec84cfb2a3647b9b1bdb314f3b3e0d3f5
● MCR	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-MasterChefV3/receiver/MasterChefV3Receiver.sol	3076aecf84ba5c65794c5bf059f9cb86a8bfa80add98e7c4b2ed903c98c30c1d
● MCM	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-MasterChefV3/receiver/MasterChefV3ReceiverV2.sol	208d421d2698760aaeb6c7a617115bc2cf952c9bea36be9de5460b2d372ee292
● MCV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-MasterChefV3/MasterChefV3.sol	457ad0e15b2db94d3926e2704ce58e7c517b1e9e7a2fcf7ed1496323cfbb4aed
○ BTG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/base/BlockTimestamp.sol	e5ca9a8b6b9e0cafcd9a9966b05228a1572f82fccee396d2e0eff5f8aa9bb1f4
○ ERC	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/base/ERC721Permit.sol	d917dd488471948d666b4c929f9df7a3b4133db6874de2c8c2a1a2e713c0e984
○ LMG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/base/LiquidityManagement.sol	0fd5f7311ee692976ea3f4df752dd65f9afc6dcf71ac10ce1c6ab952de9a0de5
○ MGV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/base/Multicall.sol	029ad0bcade48ff32da51094a3fb245fd7d8324c4fb4dd20fb4b2614efc9618c
○ PVG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/base/PeripheryValidation.sol	40877c212ebd04f41a3c582bbbf8ad925f31b2a4f7f129352f55777c8fd584a0
○ SPG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/base/SelfPermit.sol	48bb499a5e2bb8063788faf42ba0abd71cbd63392aa4d4c12531b530419d6afa
○ PFG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/examples/PairFlash.sol	c3eb8634afaa355d6fffdafb1806c39936f1ed28e4b422a1e8343c7d1daa53de
○ PIM	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/lens/PancakeInterfaceMulticall.sol	c9ca6f322f4beba5b5c19ebf7413e1882d78312445a8218c68b4502af4fa9c00

ID	Repo	File	SHA256 Checksum
REA	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/lens/README.md	f5a21edc2580bf53396761017b02ee1c295eae987eced4955da2ced2f3b8b2a6
TLG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/lens/TickLens.sol	c380aa2f427f1e3005d322766c9c80a13350e69f8aaa32fdf89469a0d27c0552
BLG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/BytesLib.sol	abe5da07d5e9f890fc64ca7b9283fa88a81a0909e4510452bdfb470d4d49bddf
CIG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/ChainId.sol	f6520df5263c8938a53d2a53ee274d959ba63770c6e70c6863a5728a905ad751
HSG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/HexStrings.sol	ef3e21095654da1dd3272db0048b13a43491868d74862f39faf4b251fb59a1c5
NFD	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/NFTDescriptor.sol	9feab8bfd0b7e07bf2c3b8240da53fbee129f38c118a7f2c3a471c515714d5e4
NFS	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/NFTSVG.sol	64698b6c33d0da81917bd4cc898c0b8e9166179b7e13bb6d4787b856e864e1ec
OLG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/OracleLibrary.sol	c179d37b28dcfe13aff2e09681a88a4647e0dbc10e6146273abdb731d616f307
PGV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/Path.sol	42edaa8b6c577bee7a24b2f1d377fa7fb7649526a935040ccdd1a91a7f3b46a0
PTC	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/PoolTicksCounter.sol	cb76d6de5ead9e122f7bf6eba35590d523461303972b35b4b3485b7e27ced6b7
PKG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/PositionKey.sol	b811728b2a5081639f7186390533821b9407b71a3172d72fa14ed5c19a15c8fc
PVV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/PositionValue.sol	1c0c648f34e3a94e11f222e91bed9c89199c66dec27dfb16f33778b4455498f7
TRS	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/libraries/TokenRatioSortOrder.sol	f9b23b4efa07365c4102e7b088672e18c052c1e59c04bfc1535dc73525df1df1
NTD	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/NonfungibleTokenPositionDescriptorOffChain.sol	a8085b77a34122dae1358e8dff09ed0bb91dc84e7ca52a6c1548abe990d9229a

ID	Repo	File	SHA256 Checksum
NTO	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Periphery/NonfungibleTokenPositionDescriptorOffChainV2.sol	9041d1e442dd614203d15079de17c3a2de449930b65f4a732db0bf1893f382ff
FPG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/FixedPoint128.sol	cfc3aef8851f183492547dccc168bf72398fba2aad4c4d9d4784f542a8ccda34
FPV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/FixedPoint96.sol	219deb88ffbcdefa482be35051db586378e8523062bee592dd2c5fa7fb47ebd6
LMV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/LiquidityMath.sol	84d20a16d5346f6ec4c12dff4df23dda5d46e52d33f18aaaaac2e9e36ce4a072
OGV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/Oracle.sol	e77c590445158e991b377da4ce33d42c98d5ac842cdd1ad6cf1c7ba4c541a457
PGC	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/Position.sol	d87d5ecd8531d9311e0953462b56ccd6453b65107cc62f43602f59a4edccb806
SMG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/SwapMath.sol	cdb205f8790e6c8a3587bd3db6eec6fba874afca1c0c6e890d87452f7aadc902
TGV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/Tick.sol	d938c31db4532ea087d90c38379ddc0a4ee5709b44a421abf87961e0730d008c
TMG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/TickMath.sol	2d8f33ec1f957582b70c6fcef4eaedbeac081c01f6f23fb5e9ba7eb4c16ef5e6
UMG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-Core/libraries/UnsafeMath.sol	4d02353eb503e3111e25bd50104ac9b279f99e88d848e455262a3fbeb55c50e7
LTG	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-LM-Pool/libraries/LmTick.sol	4d8c9e5693284e02a88ae39550250e2dba2dc0c6f8c01dba29291c1a579902bd
SCV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-MasterChefV3/libraries/SafeCast.sol	308851a754c1b946d5664f11db11d622e30bfd3d35015186294bfc3524535644
MGM	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-MasterChefV3/utils/Multicall.sol	24945f705c61471f6338630710b81589d5a106dc74e42c0db71fc32a7358b585
EGV	CrederLabs/Goldstation-V3-Contracts	Goldstation-V3-MasterChefV3/Enumerable.sol	b07a199e4befd5186d6e5d6307ffba3b09b1ae8e6b78549ae41dc37a8c714aca

APPROACH & METHODS | CREDER (STAN)_GOLDSTATION V3 - AUDIT

This report has been prepared for Creder to discover issues and vulnerabilities in the source code of the Creder (Stan)_Goldstation V3 - audit project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Formal Verification, Manual Review, and Static Analysis techniques.

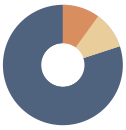
The auditing process pays special attention to the following considerations:

- 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:

- Testing the smart contracts against both common and uncommon attack vectors;
- 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.

FINDINGS | CREDER (STAN)_GOLDSTATION V3 - AUDIT



10	0	1	0	1	8
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Creder (Stan)_Goldstation V3 - audit. Through this audit, we have uncovered 10 issues ranging from different severity levels. Utilizing the techniques of Formal Verification, Manual Review & Static Analysis to complement rigorous manual code reviews, we discovered the following findings:

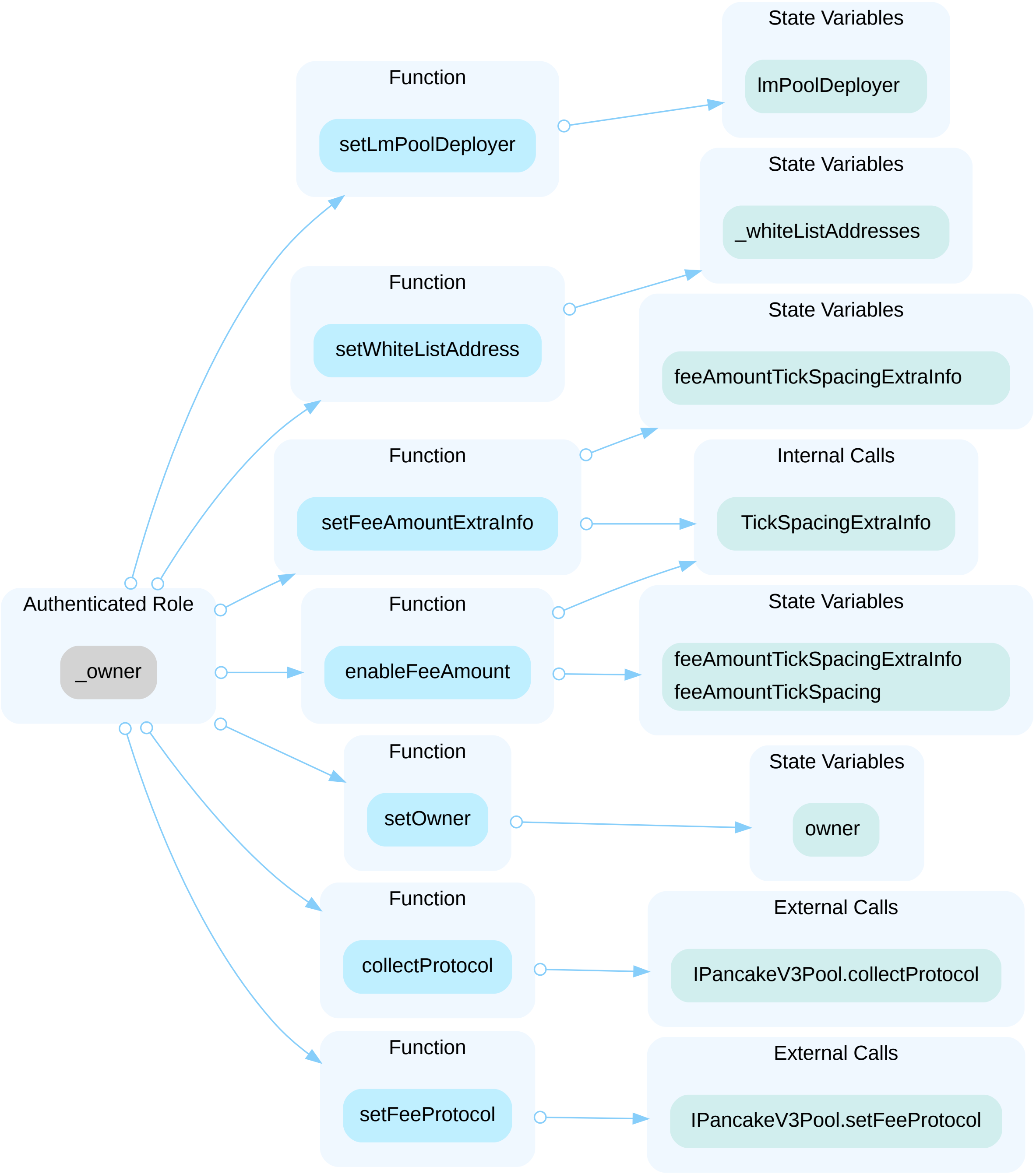
ID	Title	Category	Severity	Status
GVC-01	**Centralization Risks**	**Centralization**	**Major**	● **Acknowledged**
PVD-01	Front-Running Risk Due To Lack Of Access Control	Access Control, Volatile Code	Minor	● Acknowledged
GMC-01	Contracts May Fail To Resume If Owner Renounce Ownership During Pause	Design Issue	Informational	● Acknowledged
GVC-02	Missing Zero Address Validation	Volatile Code	Informational	● Acknowledged
GVC-03	Underscore Prefix For Non-External Variables	Code Optimization	Informational	● Acknowledged
GVC-04	`safeTransfer` Function Requires Token Existence Check Before Being Called	Logical Issue	Informational	● Acknowledged
GVC-05	Potential Out-Of-Bounds Access In `pendingCake()` Function	Volatile Code	Informational	● Acknowledged
GVC-07	Missing Error Messages	Coding Style	Informational	● Acknowledged
GVC-08	Missing Emit Events	Coding Style	Informational	● Acknowledged
GVV-01	Unused Custom Error	Coding Issue	Informational	● Resolved

GVC-01 | CENTRALIZATION RISKS

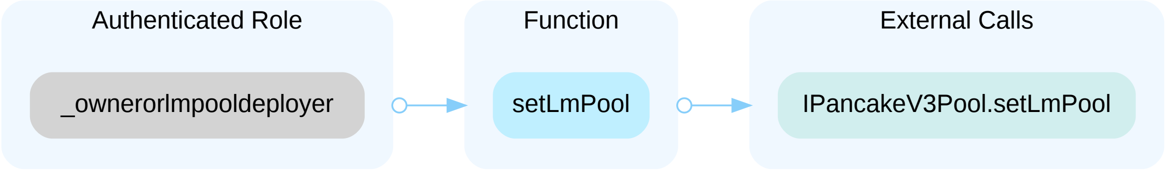
Category	Severity	Location	Status
Centralization	● **Major**	**Goldstation-V3-Core/PancakeV3Factory.sol: 60, 83, 89, 104, 112, 126, 131, 135, 144; Goldstation-V3-Core/PancakeV3Pool.sol: 865, 877, 899; Goldstation-V3-Core/PancakeV3PoolDeployer.sol: 45; Goldstation-V3-LM-Pool/PancakeV3LmPool.sol: 58, 82, 96; Goldstation-V3-LM-Pool/PancakeV3LmPoolDeployer.sol: 37; Goldstation-V3-MasterChefV3/MasterChefV3.sol: 245, 250, 257, 268, 304, 466, 716, 762, 776, 785, 793; Goldstation-V3-MasterChefV3/keeper/MasterChefV3KeeperV1.sol: 73, 81, 90, 98, 106, 112, 116; Goldstation-V3-MasterChefV3/keeper/MasterChefV3KeeperV2.sol: 65, 73, 82, 90, 98, 104, 108; Goldstation-V3-MasterChefV3/receiver/MasterChefV3Receiver.sol: 51, 70, 84, 93; Goldstation-V3-MasterChefV3/receiver/MasterChefV3ReceiverV2.sol: 45, 58, 67; Goldstation-V3-Periphery/NFTDescriptorEx.sol: 488, 494; Goldstation-V3-Periphery/V3Migrator.sol: 34; Goldstation-V3-Periphery/base/PeripheryPayments.sol: 14**	● **Acknowledged**

Description

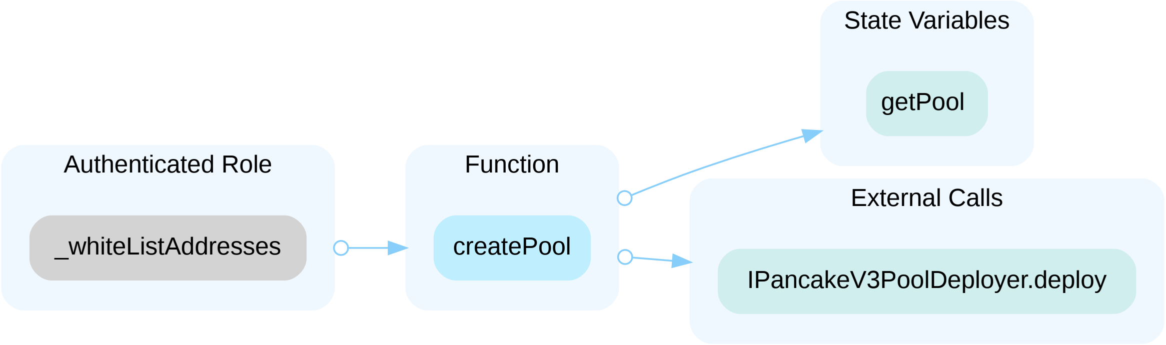
In the contract `PancakeV3Factory` , the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and set the lmPoolDeployer address, set whitelist address state, set fee amount extra info, enable a fee amount with tick spacing, set the contract owner, collect protocol fees from the specified pool, and set fee protocol for a pool.



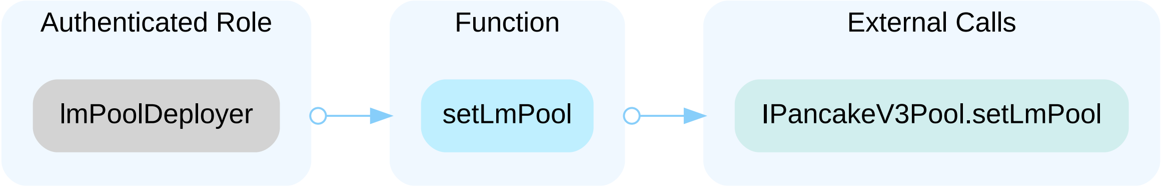
In the contract `PancakeV3Factory` , the role `_ownerorlmpooldeployer` has authority over the functions shown in the diagram below. Any compromise to the `_ownerorlmpooldeployer` account may allow the hacker to take advantage of this authority and set the liquidity mining pool.



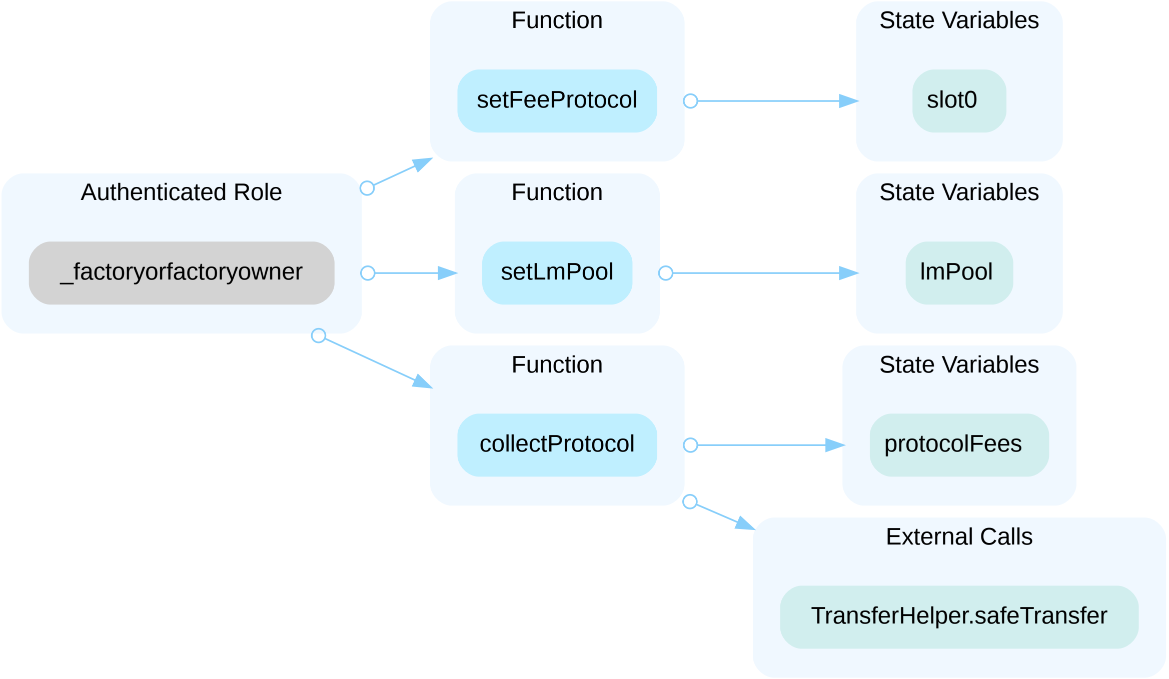
In the contract `PancakeV3Factory` , the role `_whiteListAddresses` has authority over the functions shown in the diagram below. Any compromise to the `_whiteListAddresses` account may allow the hacker to take advantage of this authority and create a liquidity pool.



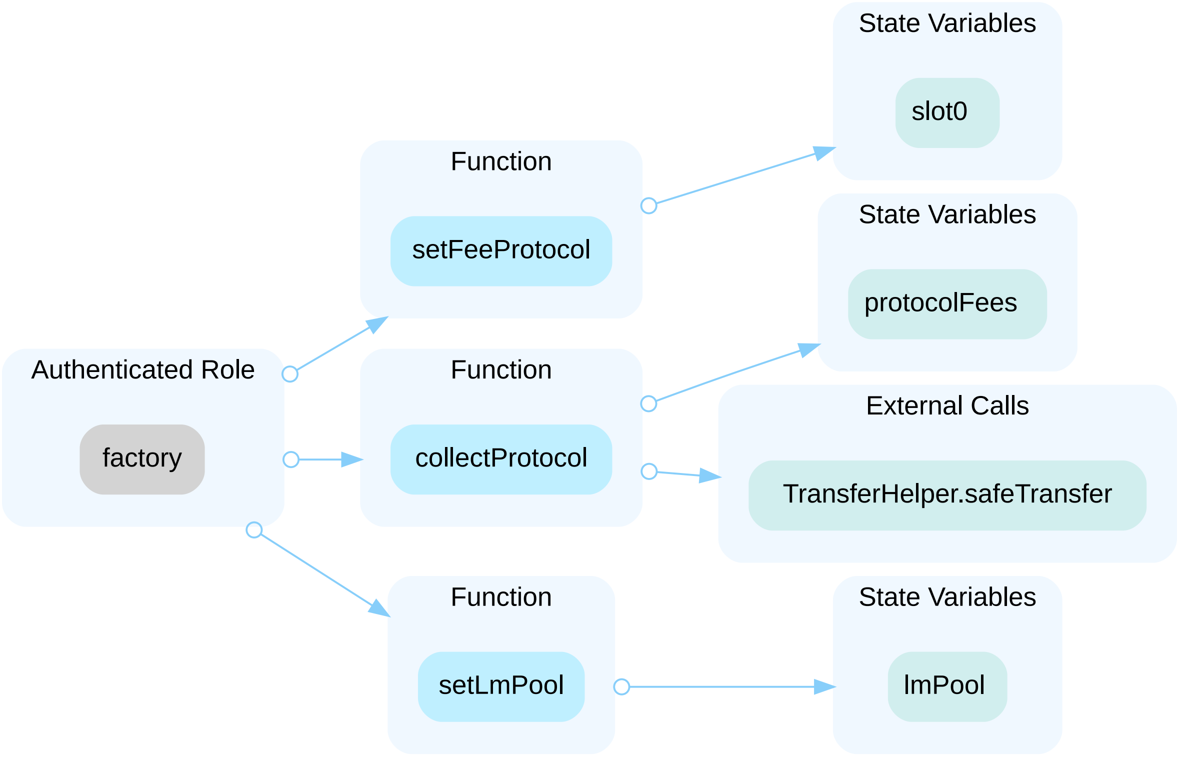
In the contract `PancakeV3Factory` , the role `lmPoolDeployer` has authority over the functions shown in the diagram below. Any compromise to the `lmPoolDeployer` account may allow the hacker to take advantage of this authority and set the liquidity mining pool.



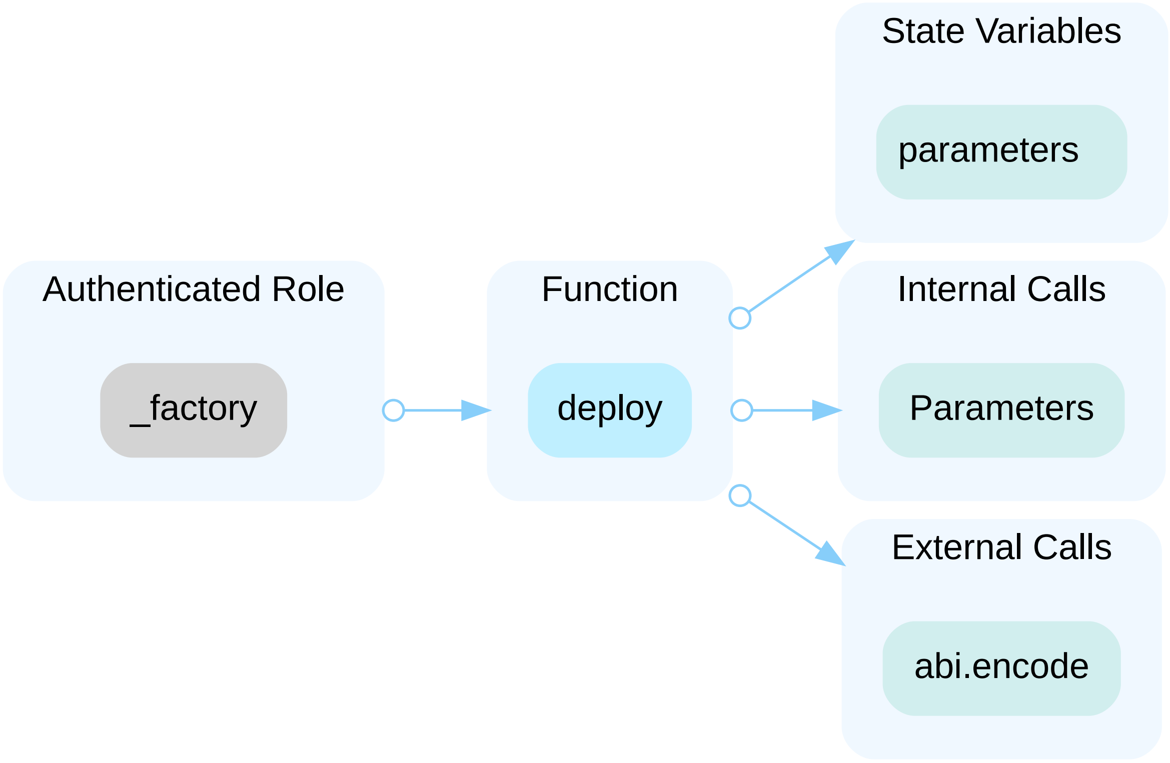
In the contract `PancakeV3Pool` , the role `_factoryorfactoryowner` has authority over the functions shown in the diagram below. Any compromise to the `_factoryorfactoryowner` account may allow the hacker to take advantage of this authority and set the fee protocol, set the liquidity mining pool, and collect protocol fees to recipient.



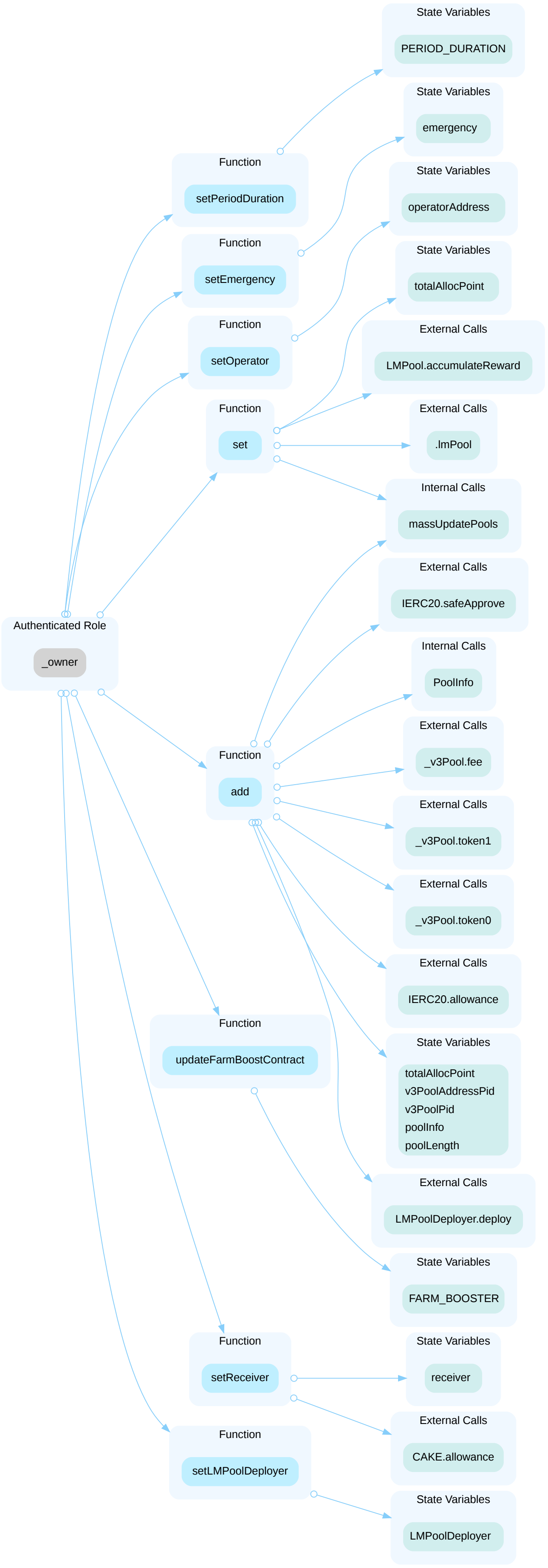
In the contract `PancakeV3Pool` , the role `factory` has authority over the functions shown in the diagram below. Any compromise to the `factory` account may allow the hacker to take advantage of this authority and set the fee protocol, collect protocol fees to a recipient, and set the lm pool address.



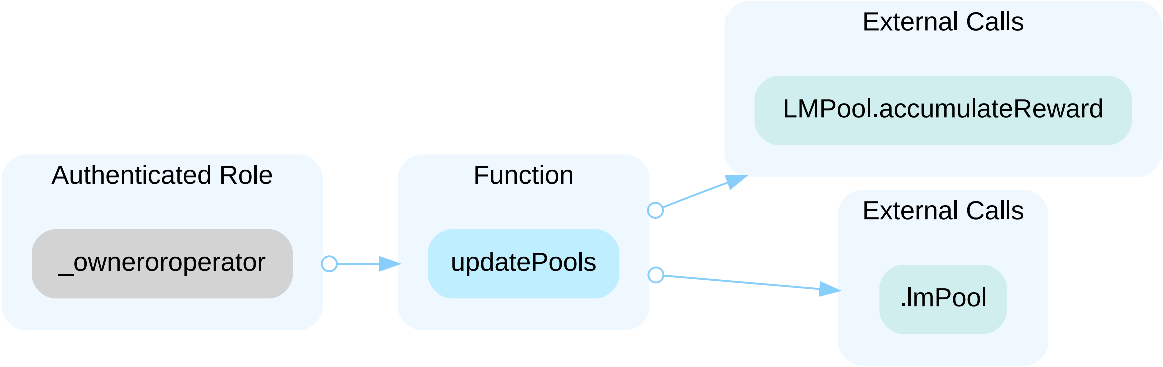
In the contract `PancakeV3PoolDeployer` , the role `_factory` has authority over the functions shown in the diagram below. Any compromise to the `_factory` account may allow the hacker to take advantage of this authority and deploy a new PancakeV3Pool instance.



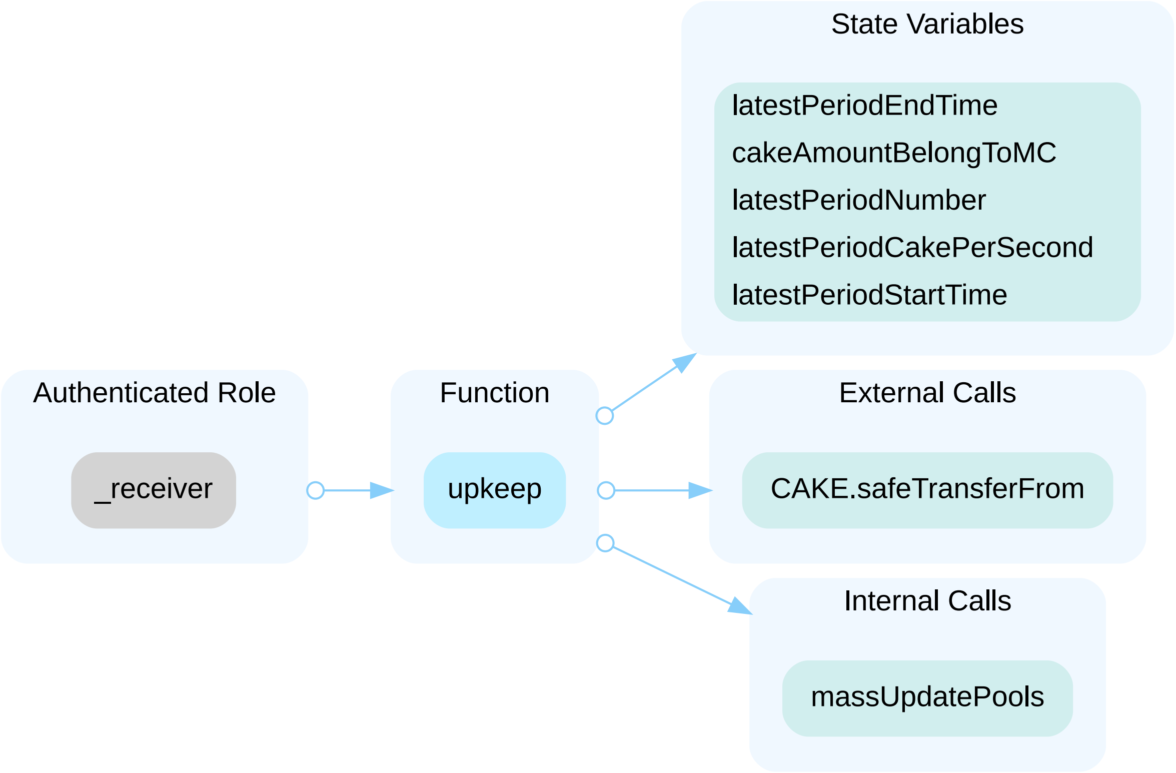
In the contract `MasterChefV3` , the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and set the period duration, set the emergency status, set the operator address, add a new liquidity pool, update farm boost contract address, set the receiver address, and set the LM pool deployer address.


State Variables
PERIOD_DURATION
State Variables
emergency
State Variables
operatorAddress
State Variables
totalAllocPoint
External Calls
LMPool.accumulateReward
External Calls
.lmPool
Internal Calls
massUpdatePools
External Calls
IERC20.safeApprove
Internal Calls
PoolInfo
External Calls
_v3Pool.fee
External Calls
_v3Pool.token1
External Calls
_v3Pool.token0
External Calls
IERC20.allowance
State Variables
totalAllocPoint
v3PoolAddressPid
v3PoolPid
poolInfo
poolLength
External Calls
LMPoolDeployer.deploy
State Variables
FARM_BOOSTER
State Variables
receiver
External Calls
CAKE.allowance
State Variables
LMPoolDeployer
Function
setPeriodDuration
Function
setEmergency
Function
setOperator
Function
set
Authenticated Role
_owner
Function
add
Function
updateFarmBoostContract
Function
setReceiver
Function
setLMPoolDeployer

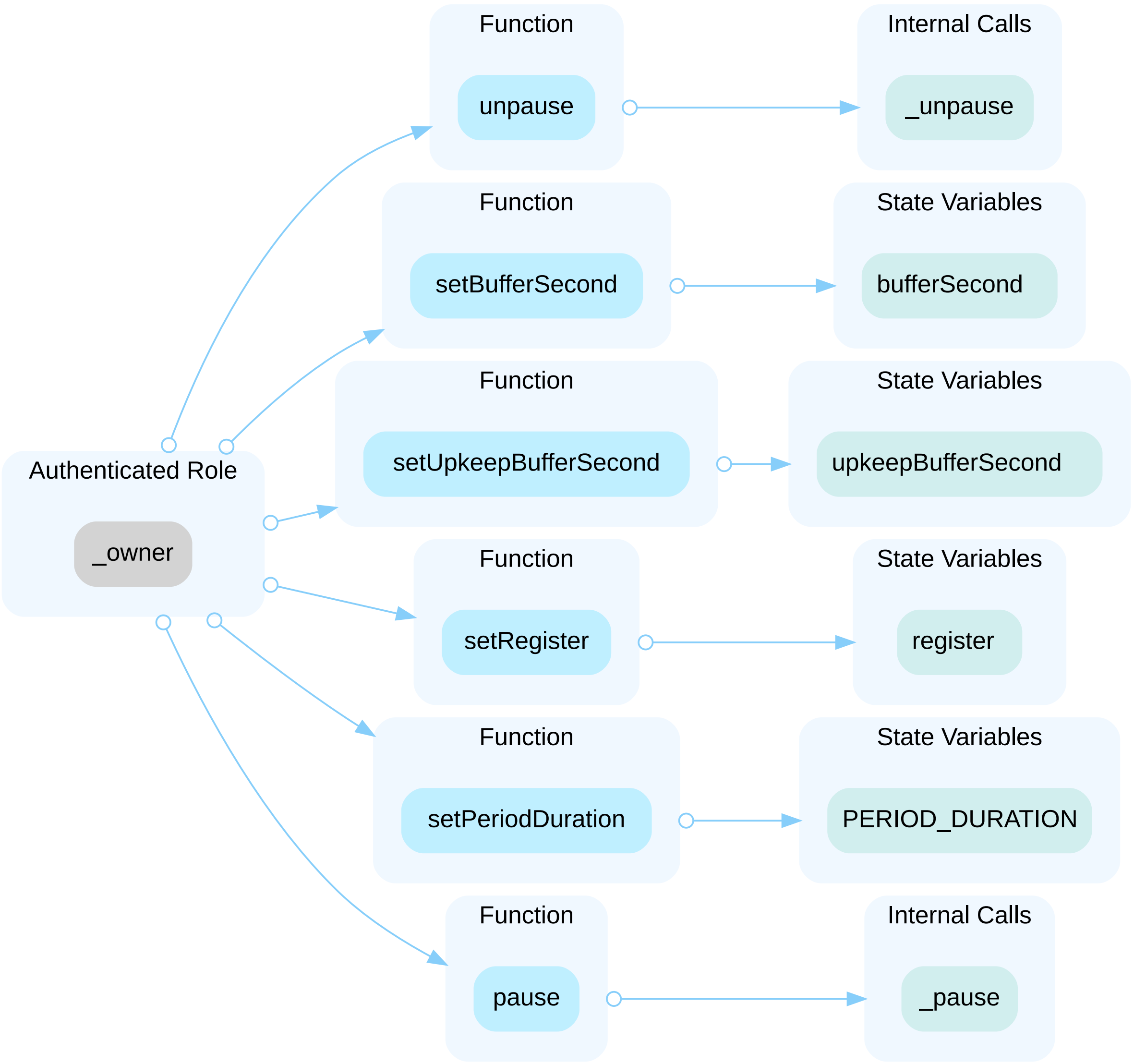
In the contract `MasterChefV3` , the role `_owneroroperator` has authority over the functions shown in the diagram below. Any compromise to the `_owneroroperator` account may allow the hacker to take advantage of this authority and update the reward for specified pools.



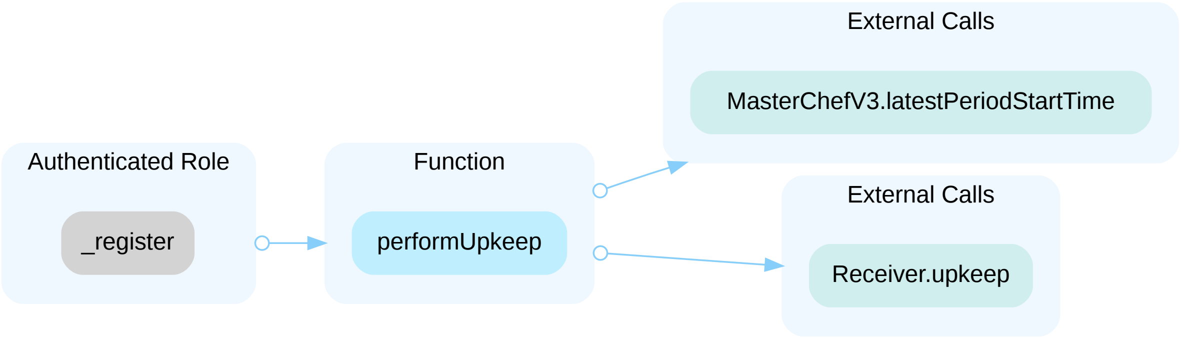
In the contract `MasterChefV3` , the role `_receiver` has authority over the functions shown in the diagram below. Any compromise to the `_receiver` account may allow the hacker to take advantage of this authority and perform upkeep by transferring and updating cake parameters.



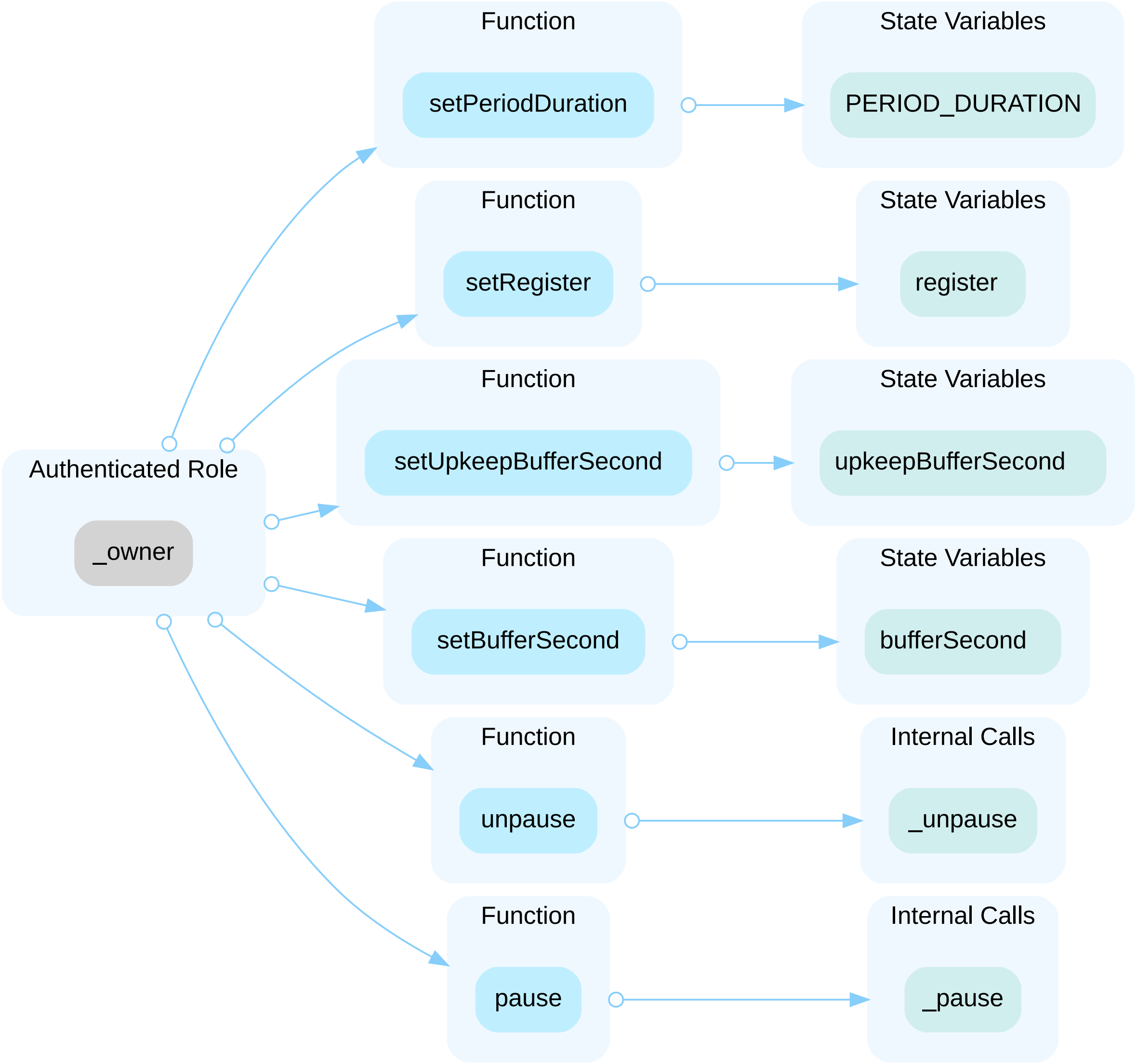
In the contract `MasterChefV3KeeperV1` , the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and unpause the contract, set the buffer second value, set upkeep buffer second and emit event, set the register address, set the period duration, and pause the contract.



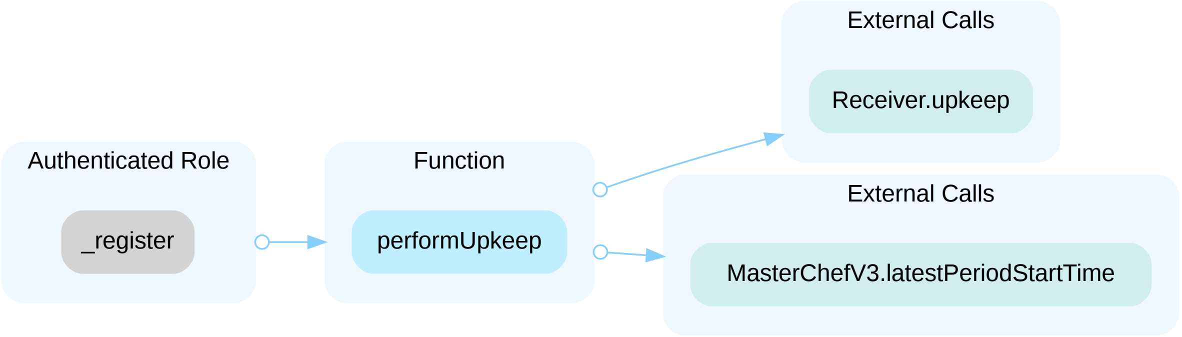
In the contract `MasterChefV3KeeperV1` , the role `_register` has authority over the functions shown in the diagram below. Any compromise to the `_register` account may allow the hacker to take advantage of this authority and perform upkeep if buffer time is exceeded.



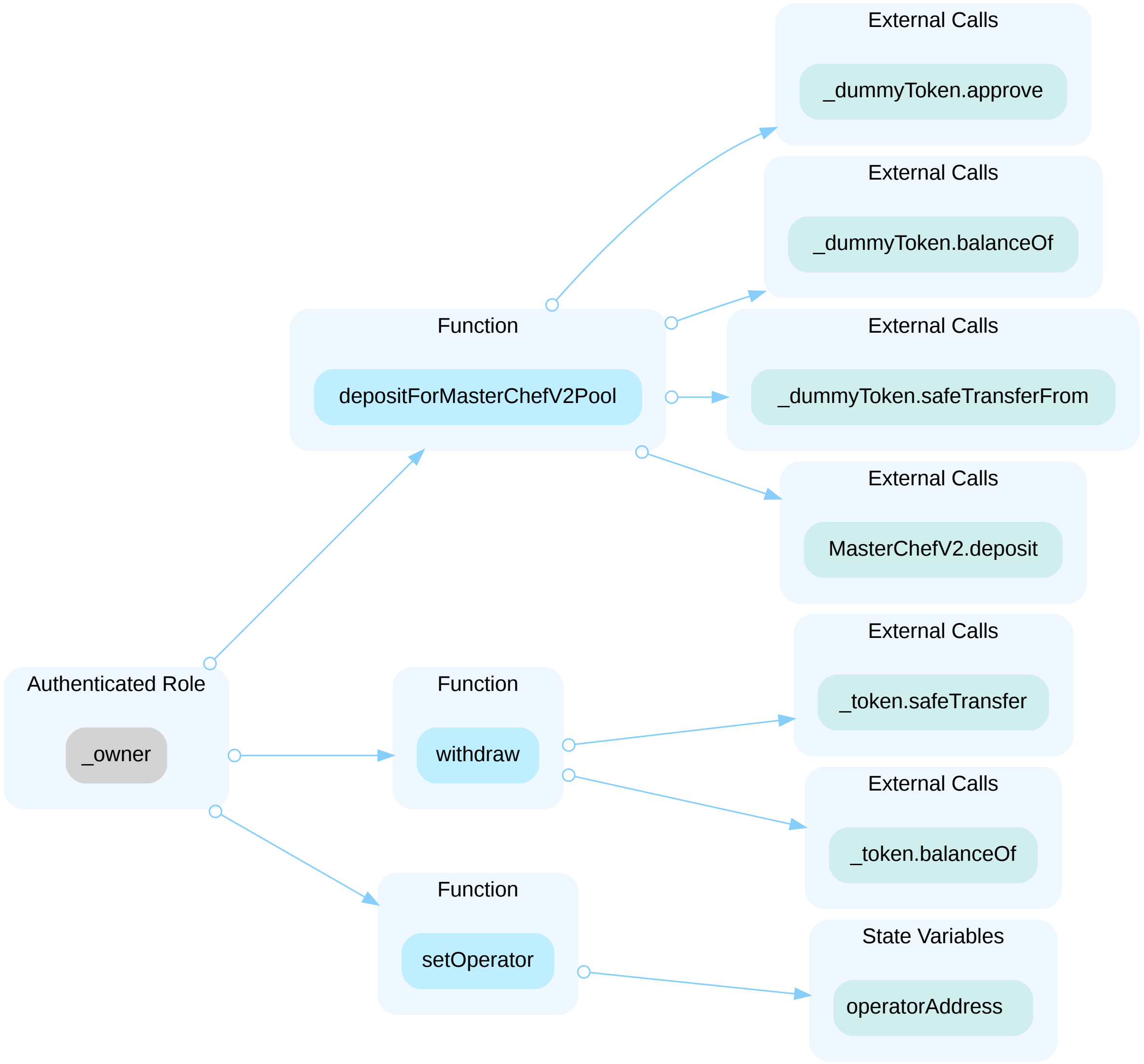
In the contract `MasterChefV3KeeperV2` , the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and set the period duration, set the register address, set the upkeep buffer second, set the buffer second value, unpause the contract, and pause contract operations.



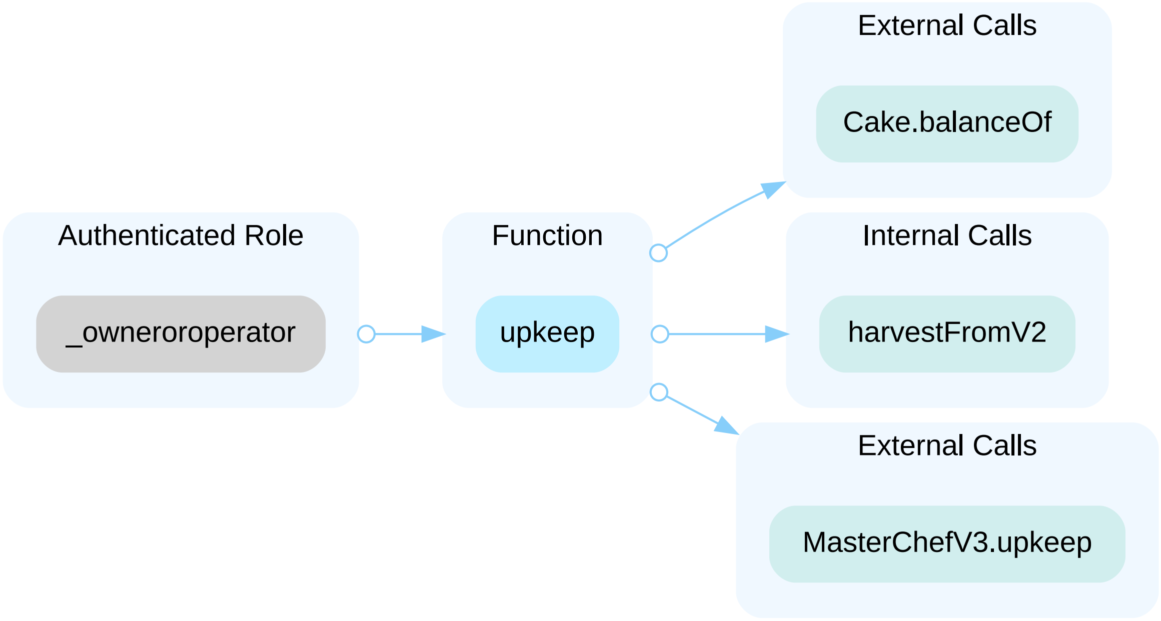
In the contract `MasterChefV3KeeperV2` , the role `_register` has authority over the functions shown in the diagram below. Any compromise to the `_register` account may allow the hacker to take advantage of this authority and perform upkeep operations if conditions are met.



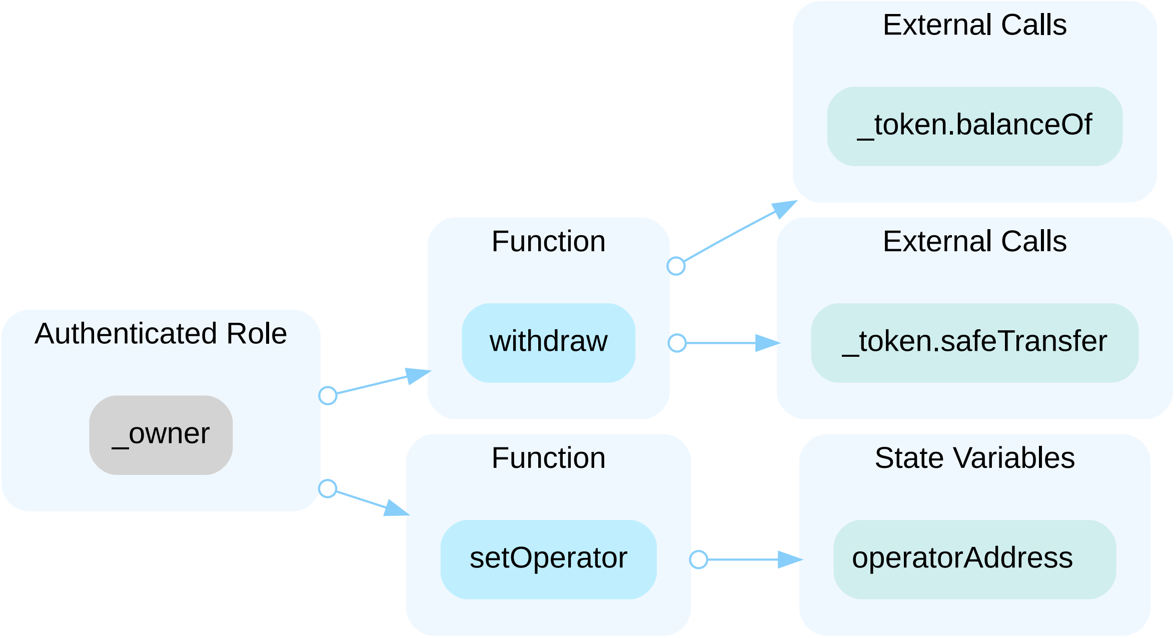
In the contract `MasterChefV3Receiver` , the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and deposit tokens for MasterChefV2 pool, withdraw tokens to owner, and set operator address.



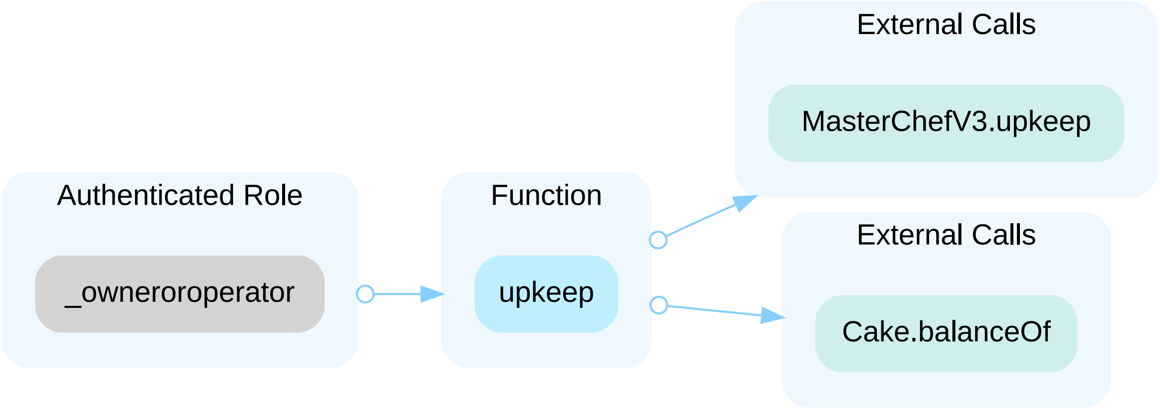
In the contract `MasterChefV3Receiver` , the role `_owneroroperator` has authority over the functions shown in the diagram below. Any compromise to the `_owneroroperator` account may allow the hacker to take advantage of this authority and perform maintenance with the specified amount and duration.



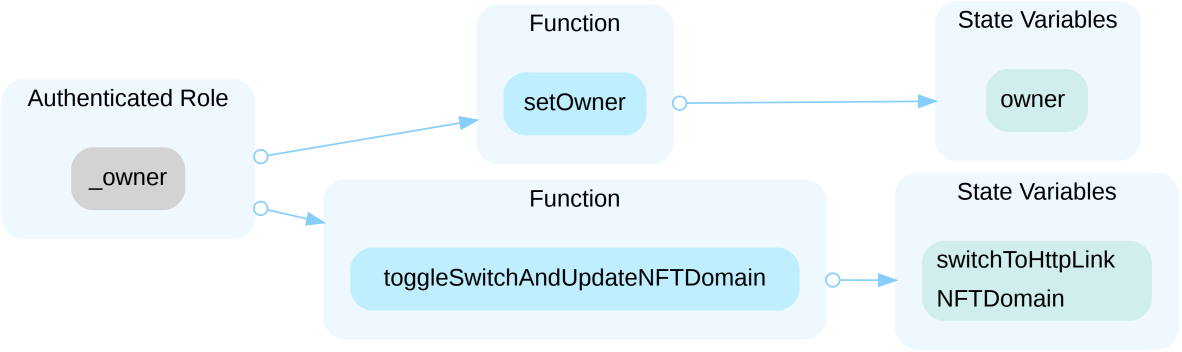
In the contract `MasterChefV3ReceiverV2` , the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority and withdraw all tokens in the contract, set the operator address.



In the contract `MasterChefV3ReceiverV2` , the role `_owneroroperator` has authority over the functions shown in the diagram below. Any compromise to the `_owneroroperator` account may allow the hacker to take advantage of this authority and perform upkeep on the MasterChefV3 contract.



In the contract `NFTDescriptorEx` , 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 set the contract owner, toggle switch, and update NFT domain.



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

[Cedar team, 12/9/2024]: We are going to establish DAO in the future. Until the DAO is established, the permitted addresses are managed by CTO.

[CertiK, 12/11/2024]: It is suggested to implement the aforementioned methods to avoid centralized failure. Also, CertiK strongly encourages the project team to periodically revisit the private key security management of all addresses related to centralized roles.

PVD-01 | FRONT-RUNNING RISK DUE TO LACK OF ACCESS CONTROL

Category	Severity	Location	Status
Access Control, Volatile Code	● Minor	Goldstation-V3-Core/PancakeV3PoolDeployer.sol: 30~36	● Acknowledged

Description

The specified function on the identified line allows public initialization of an important contract address.

However, despite the function has restriction that it can only be called once for initial setup, it remains vulnerable to front-running by malicious actors.

Although the team may effectively manage post-deployment risks by allowing contract replacement if necessary, errors could still arise during incorrect deployment handling.

Recommendation

We recommend enforcing access controls to ensure only trusted entities can execute the function, or integrating initialization within the `initialize` function or the constructor.

Alleviation

[Cedar team, 12/9/2024]: I've recognized this issue. The setFactoryAddress method will be called right after the PoolDeployer contract deployed. I will handle the front-running vulnerability by re-deploying contracts.

GMC-01 | CONTRACTS MAY FAIL TO RESUME IF OWNER RENOUNCE OWNERSHIP DURING PAUSE

Category	Severity	Location	Status
Design Issue	● Informational	Goldstation-V3-MasterChefV3/keeper/MasterChefV3KeeperV1.sol: 16; Goldstation-V3-MasterChefV3/keeper/MasterChefV3KeeperV2.sol: 16	● Acknowledged

Description

The contract inherits from `Pausable` and `Ownable` at the same time.

If the owner of a smart contract renounces ownership while the contract is paused, it means that there will be no one with the necessary permissions to unpause the contract. This could result in a permanent state of pause, effectively freezing all contract functionality that is dependent on the pause state.

Recommendation

Consider modifying the `renounceOwnership` function to include a condition that checks whether the contract is paused.

GVC-02 | MISSING ZERO ADDRESS VALIDATION

Category	Severity	Location	Status
Volatile Code	● Informational	Goldstation-V3-Core/PancakeV3Factory.sol: 37, 85, 127; Goldstation-V3-Core/PancakeV3PoolDeployer.sol: 33; Goldstation-V3-LM-Pool/PancakeV3LmPoolDeployer.sol: 32; Goldstation-V3-MasterChefV3/MasterChefV3.sol: 190, 192; Goldstation-V3-MasterChefV3/keeper/MasterChefV3KeeperV1.sol: 53; Goldstation-V3-MasterChefV3/keeper/MasterChefV3KeeperV2.sol: 48; Goldstation-V3-MasterChefV3/receiver/MasterChefV3Receiver.sol: 42; Goldstation-V3-MasterChefV3/receiver/MasterChefV3ReceiverV2.sol: 35; Goldstation-V3-Periphery/NFTDescriptorEx.sol: 489; Goldstation-V3-Periphery/NonfungiblePositionManager.sol: 77; Goldstation-V3-Periphery/NonfungibleTokenPositionDescriptor.sol: 34, 36; Goldstation-V3-Periphery/V3Migrator.sol: 31; Goldstation-V3-Periphery/base/PeripheryImmutableState.sol: 17, 18, 19	● Acknowledged

Description

The cited address input is missing a check that it is not `address(0)` .

Recommendation

We recommend adding a check the passed-in address is not `address(0)` to prevent unexpected errors.

GVC-03 | UNDERSCORE PREFIX FOR NON-EXTERNAL VARIABLES

Category	Severity	Location	Status
Code Optimization	● Informational	Goldstation-V3-MasterChefV3/MasterChefV3.sol: 60; Goldstation-V3-Periphery/NFTDescriptorEx.sol: 46, 48; Goldstation-V3-Periphery/SwapRouter.sol: 38; Goldstation-V3-Periphery/lens/Quoter.sol: 25; Goldstation-V3-Periphery/lens/QuoterV2.sol: 28	● Acknowledged

Description

The current contract doesn't follow the naming convention specified by Solidity DOC:

The state variable `variable` is used as `private` or `internal` and is not exposed publicly. It should have `an underscore prefix` like `_varaible`. Leading underscores allow you to immediately recognize the intent of such functions, but more importantly, if you change a function from non-external to external (including public) and rename it accordingly, this forces you to review every call site while renaming. This can be an important manual check against unintended external functions and a common source of security vulnerabilities (avoid find-replace-all tooling for this change).

Recommendation

To mitigate this issue, it is recommended to follow the naming conventions, including:

- Rename the variable by adding underscore prefix.

GVC-04 `safeTransfer` FUNCTION REQUIRES TOKEN EXISTENCE CHECK BEFORE BEING CALLED

Category	Severity	Location	Status
Logical Issue	● Informational	Goldstation-V3-Core/libraries/TransferHelper.sol: 19~21; Goldstation-V3-Periphery/libraries/TransferHelper.sol: 34	● Acknowledged

Description

The `safeTransfer` function lacks a verification step to confirm the existence of the ERC20 token contract prior to initiating the transfer. Consequently, it falls upon the user to ensure the token's existence before making the call.

Scenario

Performing `safeTransfer` without token existence check allows malicious people to pair with a qualified token like ETH with dubious tokens that they can destroy later, and most importantly, to run the safeTransfer function even if the token contract is later destroyed.

Recommendation

Check for the existence of the ERC20 token contract and confirm the contract's existence.

Alleviation

[Cedar team, 12/9/2024]: Issue acknowledged. I won't make any changes for the current version. Goldstation is a DEX though, exposing pool list is controlled by Goldstation foundation.

GVC-05 | POTENTIAL OUT-OF-BOUNDS ACCESS IN `pendingCake()` FUNCTION

Category	Severity	Location	Status
Volatile Code	● Informational	Goldstation-V3-MasterChefV3/MasterChefV3.sol: 224, 699; Goldstation-V3-Periphery/NonfungiblePositionManager.sol: 102	● Acknowledged

Description

The `pendingCake()` function retrieves information from the `userPositionInfos[]` array using an `_tokenId` . However, there is no check to ensure that the `_tokenId` is within the valid range of the array. If the `_tokenId` exceeds the length of the `userPositionInfos[]` array, an out-of-bounds access error could occur, leading to a contract revert.

Recommendation

Consider adding a bounds check in the function to ensure that the `_tokenId` is within the valid range of the `userPositionInfos[]` array before accessing it.

GVC-07 | MISSING ERROR MESSAGES

Category	Severity	Location	Status
Coding Style	● Informational	Goldstation-V3-Core/PancakeV3Factory.sol: 65, 67, 74, 90, 94, 95, 117; Goldstation-V3-Core/PancakeV3Pool.sol: 123, 154, 165, 198, 207, 479, 866~869; Goldstation-V3-Core/libraries/BitMath.sol: 14, 54; Goldstation-V3-Core/libraries/FullMath.sol: 34, 43, 120; Goldstation-V3-Core/libraries/LowGasSafeMath.sol: 12, 20, 28, 36, 44; Goldstation-V3-Core/libraries/SafeCast.sol: 11, 18, 25; Goldstation-V3-Core/libraries/SqrtPriceMath.sol: 52, 91, 110, 111, 133, 134, 162; Goldstation-V3-Core/libraries/TickBitmap.sol: 28; Goldstation-V3-Periphery/NonfungiblePositionManager.sol: 189, 257, 260, 297; Goldstation-V3-Periphery/SwapRouter.sol: 62, 199; Goldstation-V3-Periphery/base/PeripheryPaymentsWithFee.sol: 23, 44; Goldstation-V3-Periphery/base/PoolInitializer.sol: 19; Goldstation-V3-Periphery/lens/Quoter.sol: 43, 59; Goldstation-V3-Periphery/lens/QuoterV2.sol: 46, 68; Goldstation-V3-Periphery/libraries/CallbackValidation.sol: 34; Goldstation-V3-Periphery/libraries/LiquidityAmounts.sol: 14; Goldstation-V3-Periphery/libraries/PoolAddress.sol: 30; Goldstation-V3-Periphery/libraries/SqrtPriceMathPartial.sol: 31	● Acknowledged

Description

The **require** can be used to check for conditions and throw an exception if the condition is not met. It is better to provide a string message containing details about the error that will be passed back to the caller.

Recommendation

We advise adding error messages to the linked **require** statements.

GVC-08 | MISSING EMIT EVENTS

Category	Severity	Location	Status
Coding Style	● Informational	Goldstation-V3-Core/PancakeV3PoolDeployer.sol: 45; Goldstation-V3-LM-Pool/PancakeV3LmPool.sol: 58, 82, 96; Goldstation-V3-MasterChefV3/MasterChefV3.sol: 466, 762	● Acknowledged

Description

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

Recommendation

It is recommended emitting events for the sensitive functions that are controlled by privileged roles/addresses.

Alleviation

[Cedar team, 12/9/2024]: Issue acknowledged. Most of the parent contracts are emitting events. I won't make any changes for the current version.

GVV-01 | UNUSED CUSTOM ERROR

Category	Severity	Location	Status
Coding Issue	● Informational	Goldstation-V3-MasterChefV3/MasterChefV3.sol: 110; Goldstation-V3-MasterChefV3/receiver/MasterChefV3ReceiverV2.sol: 19	● Resolved

Description

The smart contract contains one or more custom error definitions that are not used, which can lead to unnecessary complexity and reduced maintainability.

```
110      error NoBalance();
```

- `NoBalance` is declared but never used.

```
19      error NoBalance();
```

- `NoBalance` is declared but never used.

Recommendation

It is advised to ensure that all necessary custom errors are used, and remove redundant custom errors.

Alleviation

[Cedar team, 12/9/2024]: Removed unused custom errors : https://github.com/CrederLabs/Goldstation-V3-Contracts/commit/44ea1e14352e8ebfaaaf63fe5bacec22f43a67ac

APPENDIX | CREDER (STAN)_GOLDSTATION V3 - AUDIT

Finding Categories

Categories	Description
Coding Style	Coding Style findings may not affect code behavior, but indicate areas where coding practices can be improved to make the code more understandable and maintainable.
Coding Issue	Coding Issue findings are about general code quality including, but not limited to, coding mistakes, compile errors, and performance issues.
Access Control	Access Control findings are about security vulnerabilities that make protected assets unsafe.
Volatile Code	Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases and may result in vulnerabilities.
Logical Issue	Logical Issue findings indicate general implementation issues related to the program logic.
Centralization	Centralization findings detail the design choices of designating privileged roles or other centralized controls over the code.
Design Issue	Design Issue findings indicate general issues at the design level beyond program logic that are not covered by other finding categories.

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 | CERTIK

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.

Elevating Your Entire Web3 Journey

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.