AUDITED BY
LOVELY INSPECTOR

SMART CONTRACT

SECURITY AUDIT

SUI

inspector.lovely.finance

LOVELY INSPECTOR

TABLE OF CONTENTS

Table of Contents 2

Disclaimer 3

Audit Scope 4

Proposed Smart Contract Features 5

Audit Summary 6

Key Technical Metrics 7

Business Risk Analysis 8

Code Quality 9

Documentation 9

Use of Dependencies 9

Project Website Performance Audit 10

Level of Criticality 11

Audit Findings Table 12

Audit Findings 13

Centralization 14

Conclusion 15

Addendum

- Logic Diagram 16
- Security Assessment Report 17
- Solidity Static Analysis 19
- Compliance Analysis 22

Software Analysis Result 27

INSPECTOR Lovely Info 28

DISCLAIMER

This is a comprehensive report based on our automated and manual examination of cybersecurity vulnerabilities and framework flaws of the project's smart contract. Reading the full analysis report is essential to build your understanding of the project's security level. It is crucial to take note, though we have done our best to perform this analysis and report, that you should not rely on our research and cannot claim what it states or how we created it. Before making any judgments, you have to conduct your own independent research. We will discuss this in more depth in the following disclaimer - please read it fully. DISCLAIMER: You agree to the terms of this disclaimer by reading this report or any portion thereof. Please stop reading this report and remove and delete any copies of this report that you download and/or print if you do not agree to these conditions. Scan and verify the report's presence in the GitHub repository by a QR code on the title page. This report is for non-reliability information only and does not represent investment advice. No one shall be entitled to depend on the report or its contents, and Inspector Lovely and its affiliates shall not be held responsible to you or anyone else, nor shall Inspector Lovely provide any guarantee or representation to any person with regard to the accuracy or integrity of the report. Without any terms, warranties, or other conditions other than as set forth in that exclusion Inspector Lovely excludes hereby all representations, warrants, conditions, and other terms (including, without limitation, guarantees implied by the law of satisfactory quality, fitness for purposes and the use of reasonable care and skills). The report is provided as "as is" and does not contain any terms and conditions. Except as legally banned, Inspector Lovely disclaims all responsibility and responsibilities, and no claim against Inspector Lovely is made to any amount or type of loss or damages (without limitation, direct, indirect, special, punitive, consequential, or pure economic loses or losses) that may be caused by you or any other person, or any damages or damages, including without limitations (whether innocent or negligent). Security analysis is based only on smart contracts. No applications or operations were reviewed for security. No product code has been reviewed.

LOVELY INSPECTOR

AUDIT SCOPE

Name	Code Review and Security Analysis Report for SUI Smart Contract
Platform	Sui Blockchain
Language	Sui Move
File 1	Channel.move
File 2	Utils.move
File 3	Validators.move
File 4	Gateway.move
Github Commit Hash	c06e7657ae4486d2f0f4cc175f9fc2c51c6cf505
Audit Date	November 16th, 2023

LOVELY INSPECTOR

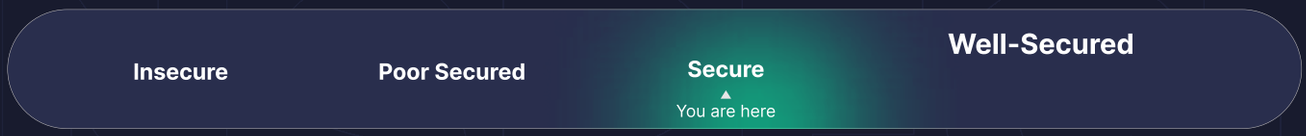
PROPOSED SMART CONTRACT FEATURES

Claimed Feature Detail	Our Observation
File 1 Channel.move Functionality Specifications: • Update Channel object. Anyone can create their own `Channel` to target from the outside and there's no limitation to the data stored inside it. • Create a new `ApprovedCall` object to be sent to another chain. Is called by the gateway when a message is "picked up" by the relayer. • Consume an approved call hot potato object sent to this `Channel` from another chain. For Capability-locking, a mutable reference to the `Channel.data` field is returned.	Validated
File 2 Utils.move Functionality Specifications: • The last byte of the signature should be normalized to either 1 or 0. • Add a prefix to the bytes	Validated
File 3 Validators.move Functionality Specifications: • Current owner can transfer the ownership.	Validated
File 4 Gateway.move Functionality Specifications: • Creates a new `ApprovedCall` object which must be delivered to the matching `Channel`.	Validated

LOVELY INSPECTOR

AUDIT SUMMARY

According to the standard audit assessment, the Customer`s solidity-based smart contracts are **“Secured”**. Also, these contracts contain owner control, which does not make them fully decentralized.



We used various tools like Slither, Solhin,t, and Remix IDE. At the same time, this finding is based on a critical analysis of the manual audit.

All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the Audit Overview section. General overview is presented in AS-IS section and all identified issues can be found in the Audit overview section.

We found 0 critical, 0 high, 0 medium and 0 low, and 0 very low level issues.

Investors Advice: Technical audit of the smart contract does not guarantee the ethical nature of the project. Any owner-controlled functions should be executed by the owner with responsibility. All investors/users are advised to do their due diligence before investing in the project.

LOVELY INSPECTOR

KEY TECHNICAL METRICS

MAIN CATEGORY	SUBCATEGORY	RESULT
Contract Programming	Solidity version is not specified	Passed
	Solidity version is too old	Passed
	Integer overflow/underflow	Passed
	Function input parameters lack check	Passed
	Function input parameters check bypass	Passed
	Function access control lacks management	Passed
	Critical operation lacks event log	Passed
	Human/contract checks bypass	Passed
	Random number generation/use vulnerability	N/A
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Passed
	Features claimed	Passed
	Other programming issues	Passed
Code Specification	Function visibility not explicitly declared	Passed
	Var. storage location not explicitly declared	Passed
	Use keywords/functions to be deprecated	Passed
	Unused code	Passed
Gas Optimization	“Out of Gas” Issue	Passed
	High consumption ‘for/while’ loop	Passed
	High consumption ‘storage’ storage	Passed
	Assert() misuse	Passed
Business Risk	The maximum limit for mintage is not set	Passed
	“Short Address” Attack	Passed
	“Double Spend” Attack	Passed

Overall Audit Result: PASSED

LOVELY INSPECTOR

BUSINESS RISK ANALYSIS

CATEGORY	RESULT
Buy Tax	0%
Sell Tax	0%
Cannot Buy	Not Detected
Cannot Sell	Not Detected
Max Tax	0%
Modify Tax	Not Detected
Fee Check	No
Is Honeypot	Not Detected
Trading Cooldown	Not Detected
Can Pause Trade?	No
Pause Transfer?	No
Max Tax?	No
Is it Anti-whale?	No
Is Anti-bot?	Not Detected
Is it a Blacklist?	Not Detected
Blacklist Check	No
Can Mint?	No
Is it Proxy?	No
Can Take Ownership?	Yes
Hidden Owner?	Not Detected
Self Destruction?	Not Detected
Auditor Confidence	High

Overall Audit Result: PASSED

LOVELY INSPECTOR

CODE QUALITY

This audit scope has 1 smart contract. Smart contract contains Libraries, Smart contracts, inherits, and Interfaces. This is a compact and well written smart contract.

The libraries in Pendle Token are part of its logical algorithm. A library is a different type of smart contract that contains reusable code. Once deployed on the blockchain (only once), it is assigned a specific address and its properties/methods can be reused many times by other contracts in the Pendle Token.

The EtherAuthority team has not provided scenario and unit test scripts, which would have helped to determine the integrity of the code in an automated way.

Code parts are well commented on in the smart contracts. Ethereum's NatSpec commenting style is recommended.

DOCUMENTATION

We were given a a sui smart smart contract code in the form of an github web link.

As mentioned above, code parts are well commented on. and the logic is straightforward. So it is easy to quickly understand the programming flow as well as complex code logic. Comments are very helpful in understanding the overall architecture of the protocol.

Another source of information was its official website: https://sui.io which provided rich information about the project architecture and tokenomics.

USE OF DEPENDENCIES

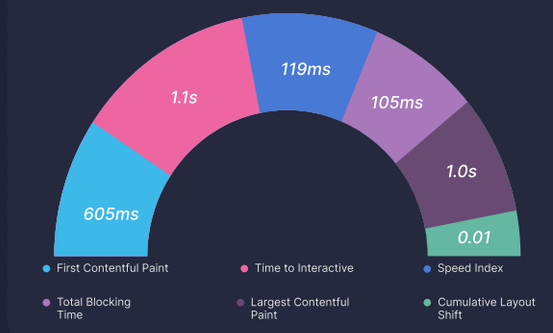
As per our observation, the libraries are used in this smart contract infrastructure that are based on well-known industry standard open-source projects.

Apart from libraries, its functions are not used in external smart contract calls.

LOVELY INSPECTOR

PROJECT WEBSITE PERFORMANCE AUDIT

Performance Metrics



Browser Timings

Redirect Duration **0ms**	Connection Duration **26ms**	Backend Duration **93ms**
Time to First Byte **119ms**	First Paint **605ms**	DOM Interactive Time **957ms**
DOM Content Loaded **1.1s**	Onload Time **2.1s**	Fully Loaded Time **2.2s**

Grade

B	Performance 94%	Structure 81%

Web Vitals

LCP	TBT	CLS
1.0s	119ms	0.01

Top Issues

IMPECT	AUDIT
High	Avoid enormous network payloads (LCP)

URL	SIZE
https://assets-global.website-files.com/6425f546844727ce5fb9e5ab/643652b4ec653a05c178a0c2_-2393330798549273605homepage_community-transcode.mp4	1.04MB
https://assets-global.website-files.com/6425f546844727ce5fb9e5ab/64377ee0d96a223b1d5c6700_01_Homepage_Hero-transcode.mp4	974KB
https://assets-global.website-files.com/6425f546844727ce5fb9e5ab/6430980fe2d0578e8cd69453_background.jpeg	860KB
https://assets-global.website-files.com/6425f546844727ce5fb9e5ab/64377eef160cb44e527145c6_02_Homepage_BuildWithConfidence-transcode.mp4	448KB
https://js.hsforms.net/forms/embed/v2.js	179KB
https://assets-global.website-files.com/6425f546844727ce5fb9e5ab/js/sui-io-dev-3459.299157ac1.js	125KB
https://www.googletagmanager.com/gtag/js?id=G-RDW50T5ML7&l=dataLayer&cx=c	90.9KB
https://www.googletagmanager.com/gtm.js?id=GTM-5C9KVWT	64.6KB
https://assets-global.website-files.com/6425f546844727ce5fb9e5ab/css/sui-io-dev-3459.bb614befd.css	59.9KB
https://fonts.gstatic.com/s/inter/v13/UcC73FwrK3iLTeHuS_fvQtMwCp50KnMa1ZL7.woff2	46.1KB

LOVELY INSPECTOR

LEVEL OF CRITICALITY

RISK LEVEL	DESCRIPTION
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to token loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial
Med	Medium-level vulnerabilities are important to fix; however, they can't lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can't have significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored.

LOVELY INSPECTOR

AUDIT FINDINGS TABLE

	Total	Resolved	UnResolved	Acknowledged
High Severity Issues Found	0	0	0	0
Moderate Severity Issues Found	0	0	0	0
Medium Severity Issues	0	0	0	0
Low Severity Issues	0	0	0	0
Informational Observations	0	0	0	0

The sui - Audit report identifies 0 issues with varying severity levels, discovered through manual review and static analysis techniques, alongside rigorous code reviews, highlighting the need for further investigation and vulnerability identification.

The smart contract is considered to **pass the audit,** as of the audit date, if no high severity or moderate severity issues are found.

LOVELY INSPECTOR

AUDIT FINDINGS

Critical Severity	No Critical severity vulnerabilities were found.
High Severity	No High severity vulnerabilities were found.
Medium	No Medium severity vulnerabilities were found.
Low	No Low severity vulnerabilities were found.
Very Low / Informational / Best practices:	No Very Low severity vulnerabilities were found.

LOVELY INSPECTOR

CENTRALIZATION

This smart contract has some functions that can be executed by the Admin (Owner) only. If the admin wallet's private key is compromised, then it would create trouble. Following are Admin functions:

Validators.move

- transfer_operatorship: Current operator can transfer the ownership.
- call_contract: Send an event from an authorized channel to call a contract on the destination chain.

To make the smart contract 100% decentralized, we suggest renouncing ownership of the smart contract once its function is completed.

LOVELY INSPECTOR

CONCLUSION

We were given a contract code in the form of github web links. And we have used all possible tests based on given objects as files. We had not observed any issues in the smart contracts. So, **it's good to go for the production.**

Since possible test cases can be unlimited for such smart contracts protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan everything.

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. Smart Contract's high-level description of functionality was presented in the As-is overview section of the report.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security state of the reviewed smart contract, based on standard audit procedure scope, is **"Secured".**

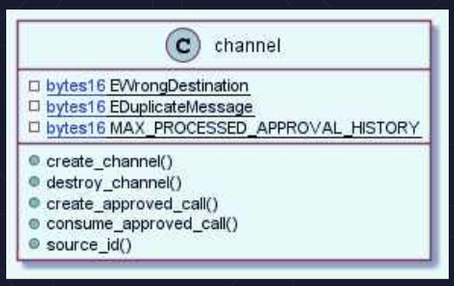
LOVELY INSPECTOR

ADDENDUM

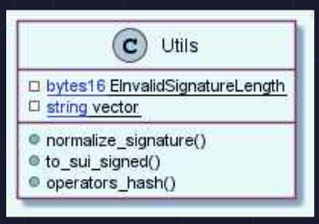
Code Flow Diagram

Sui

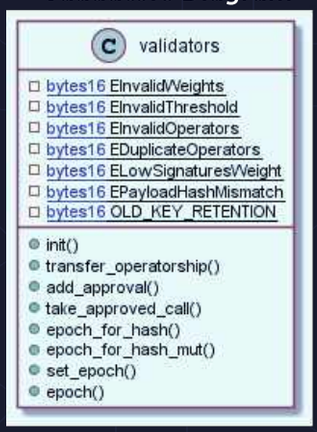
Channel Diagram



Utils Diagram



Validators Diagram



Validators Diagram

LOVELY INSPECTOR

SECURITY ASSESSMENT REPORT

Slither is a Solidity static analysis framework that uses vulnerability detectors, displays contract details and provides an API for writing custom analyses. It helps developers identify vulnerabilities, improve code comprehension, and prototype custom analyses quickly. The analysis includes a report with warnings and errors, allowing developers to quickly prototype and fix issues.

We did the analysis of the project together. Below are the results.

Slither Log >> Channel.move

```
Pragma version^0.8.1 (Channel.sol#4) allows old versions
solc-0.8.1 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Contract channel (Channel.sol#5-53) is not in CapWords
Struct channel.store (Channel.sol#12-16) is not in CapWords
Function channel.create_channel(uint256,uint256) (Channel.sol#26-28) is not in mixedCase
Parameter channel.create_channel(uint256,uint256).T (Channel.sol#26) is not in mixedCase
Parameter channel.create_channel(uint256,uint256).TxContext (Channel.sol#26) is not in mixedCase
Function channel.destroy_channel(uint256) (Channel.sol#29-31) is not in mixedCase
Parameter channel.destroy_channel(uint256).Channel (Channel.sol#29) is not in mixedCase
Function channel.create_approved_call(address,uint256,uint256,address,address) (Channel.sol#33-41) is not in mixedCase
Parameter channel.create_approved_call(address,uint256,uint256,address,address).cmd_id (Channel.sol#34) is not in mixedCase
Parameter channel.create_approved_call(address,uint256,uint256,address,address).source_chain (Channel.sol#35) is not in mixedCas
e
Parameter channel.create_approved_call(address,uint256,uint256,address,address).source_address (Channel.sol#36) is not in mixedC
ase
Parameter channel.create_approved_call(address,uint256,uint256,address,address).target_id (Channel.sol#37) is not in mixedCase
Function channel.consume_approved_call(uint256) (Channel.sol#43-47) is not in mixedCase
Parameter channel.consume_approved_call(uint256).Channel (Channel.sol#44) is not in mixedCase
Function channel.source_id(address) (Channel.sol#49-51) is not in mixedCase
Parameter channel.source_id(address).Channel (Channel.sol#49) is not in mixedCase
Constant channel.EWrongDestination (Channel.sol#7) is not in UPPER_CASE_WITH_UNDERSCORES
Constant channel.EDuplicateMessage (Channel.sol#8) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

channel.EWrongDestination (Channel.sol#7) is never used in channel (Channel.sol#5-53)
channel.EDuplicateMessage (Channel.sol#8) is never used in channel (Channel.sol#5-53)
channel.MAX_PROCESSED_APPROVAL_HISTORY (Channel.sol#10) is never used in channel (Channel.sol#5-53)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
Channel.sol analyzed (1 contracts with 84 detectors), 23 result(s) found
```

Slither Log >> Utils.move

```
Utils.normalize_signature(address).vector (Utils.sol#9) shadows:
        - Utils.vector (Utils.sol#8) (state variable)
Utils.to_sui_signed(address).vector (Utils.sol#13) shadows:
        - Utils.vector (Utils.sol#8) (state variable)
Utils.operators_hash(address,uint256,uint256).vector (Utils.sol#17) shadows:
        - Utils.vector (Utils.sol#8) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

Pragma version^0.8.1 (Utils.sol#4) allows old versions
solc-0.8.1 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Function Utils.normalize_signature(address) (Utils.sol#9-11) is not in mixedCase
Function Utils.to_sui_signed(address) (Utils.sol#13-15) is not in mixedCase
Function Utils.operators_hash(address,uint256,uint256) (Utils.sol#17-19) is not in mixedCase
Constant Utils.EInvalidSignatureLength (Utils.sol#6) is not in UPPER_CASE_WITH_UNDERSCORES
Constant Utils.vector (Utils.sol#8) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Utils.EInvalidSignatureLength (Utils.sol#6) is never used in Utils (Utils.sol#5-21)
Utils.vector (Utils.sol#8) is never used in Utils (Utils.sol#5-21)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
Utils.sol analyzed (1 contracts with 84 detectors), 12 result(s) found
```

LOVELY INSPECTOR

Slither Log >> Validators.move

```
Pragma version^0.8.1 (Validators.sol#4) allows old versions
solc-0.8.1 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
```

```
Contract validators (Validators.sol#5-82) is not in CapWords
Parameter validators.init(uint256).TxContext (Validators.sol#36) is not in mixedCase
Function validators.transfer_operatorship(uint256,uint256) (Validators.sol#41-43) is not in mixedCase
Parameter validators.transfer_operatorship(uint256,uint256).AxelarValidators (Validators.sol#41) is not in mixedCase
Function validators.add_approval(uint256,address,uint256,uint256,address) (Validators.sol#45-53) is not in mixedCase
Parameter validators.add_approval(uint256,address,uint256,uint256,address).AxelarValidators (Validators.sol#46) is not in mixedC
ase
Parameter validators.add_approval(uint256,address,uint256,uint256,address).cmd_id (Validators.sol#47) is not in mixedCase
Parameter validators.add_approval(uint256,address,uint256,uint256,address).source_chain (Validators.sol#48) is not in mixedCase
Parameter validators.add_approval(uint256,address,uint256,uint256,address).source_address (Validators.sol#49) is not in mixedCas
e
Parameter validators.add_approval(uint256,address,uint256,uint256,address).target_id (Validators.sol#50) is not in mixedCase
Function validators.take_approved_call(uint256,address,uint256,uint256,address) (Validators.sol#55-63) is not in mixedCase
Parameter validators.take_approved_call(uint256,address,uint256,uint256,address).AxelarValidators (Validators.sol#56) is not in
mixedCase
Parameter validators.take_approved_call(uint256,address,uint256,uint256,address).cmd_id (Validators.sol#57) is not in mixedCase
Parameter validators.take_approved_call(uint256,address,uint256,uint256,address).source_chain (Validators.sol#58) is not in mixe
dCase
Parameter validators.take_approved_call(uint256,address,uint256,uint256,address).source_address (Validators.sol#59) is not in mi
xedCase
Parameter validators.take_approved_call(uint256,address,uint256,uint256,address).target_id (Validators.sol#60) is not in mixedCa
se
Function validators.epoch_for_hash(address) (Validators.sol#66-68) is not in mixedCase
Parameter validators.epoch_for_hash(address).AxelarValidators (Validators.sol#66) is not in mixedCase
Function validators.epoch_for_hash_mut(address) (Validators.sol#70-72) is not in mixedCase
Parameter validators.epoch_for_hash_mut(address).AxelarValidators (Validators.sol#70) is not in mixedCase
Function validators.set_epoch(address) (Validators.sol#74-76) is not in mixedCase
Parameter validators.set_epoch(address).AxelarValidators (Validators.sol#74) is not in mixedCase
Parameter validators.epoch(address).AxelarValidators (Validators.sol#78) is not in mixedCase
Constant validators.EInvalidWeights (Validators.sol#7) is not in UPPER_CASE_WITH_UNDERSCORES
Constant validators.EInvalidThreshold (Validators.sol#8) is not in UPPER_CASE_WITH_UNDERSCORES
Constant validators.EInvalidOperators (Validators.sol#9) is not in UPPER_CASE_WITH_UNDERSCORES
Constant validators.EDuplicateOperators (Validators.sol#10) is not in UPPER_CASE_WITH_UNDERSCORES
Constant validators.ELowSignaturesWeight (Validators.sol#11) is not in UPPER_CASE_WITH_UNDERSCORES
Constant validators.EPayloadHashMismatch (Validators.sol#12) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

```
validators.EInvalidWeights (Validators.sol#7) is never used in validators (Validators.sol#5-82)
validators.EInvalidThreshold (Validators.sol#8) is never used in validators (Validators.sol#5-82)
validators.EInvalidOperators (Validators.sol#9) is never used in validators (Validators.sol#5-82)
validators.EDuplicateOperators (Validators.sol#10) is never used in validators (Validators.sol#5-82)
validators.ELowSignaturesWeight (Validators.sol#11) is never used in validators (Validators.sol#5-82)
validators.EPayloadHashMismatch (Validators.sol#12) is never used in validators (Validators.sol#5-82)
validators.OLD_KEY_RETENTION (Validators.sol#14) is never used in validators (Validators.sol#5-82)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
Validators.sol analyzed (1 contracts with 84 detectors), 38 result(s) found
```

Slither Log >> Validators.move

```
Pragma version^0.8.1 (Gateway.sol#4) allows old versions
solc-0.8.1 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Contract gateway (Gateway.sol#5-45) is not in CapWords
Function gateway.process_commands(address) (Gateway.sol#20-24) is not in mixedCase
Parameter gateway.process_commands(address).AxelarValidators (Gateway.sol#21) is not in mixedCase
Function gateway.take_approved_call(address,address,uint256,uint256,address) (Gateway.sol#26-34) is not in mixedCase
Parameter gateway.take_approved_call(address,address,uint256,uint256,address).AxelarValidators (Gateway.sol#27) is not in mixedC
ase
Parameter gateway.take_approved_call(address,address,uint256,uint256,address).cmd_id (Gateway.sol#28) is not in mixedCase
Parameter gateway.take_approved_call(address,address,uint256,uint256,address).source_chain (Gateway.sol#29) is not in mixedCase
Parameter gateway.take_approved_call(address,address,uint256,uint256,address).source_address (Gateway.sol#30) is not in mixedCas
e
Parameter gateway.take_approved_call(address,address,uint256,uint256,address).target_id (Gateway.sol#31) is not in mixedCase
Function gateway.call_contract(address,address,address) (Gateway.sol#36-42) is not in mixedCase
Parameter gateway.call_contract(address,address,address).Channel (Gateway.sol#37) is not in mixedCase
Parameter gateway.call_contract(address,address,address).destination_address (Gateway.sol#39) is not in mixedCase
Constant gateway.ESignatureInvalid (Gateway.sol#6) is not in UPPER_CASE_WITH_UNDERSCORES
Constant gateway.EInvalidCommands (Gateway.sol#8) is not in UPPER_CASE_WITH_UNDERSCORES
Constant gateway.EInvalidChain (Gateway.sol#10) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

gateway.ESignatureInvalid (Gateway.sol#6) is never used in gateway (Gateway.sol#5-45)
gateway.EInvalidCommands (Gateway.sol#8) is never used in gateway (Gateway.sol#5-45)
gateway.EInvalidChain (Gateway.sol#10) is never used in gateway (Gateway.sol#5-45)
gateway.SELECTOR_APPROVE_CONTRACT_CALL (Gateway.sol#12) is never used in gateway (Gateway.sol#5-45)
gateway.SELECTOR_TRANSFER_OPERATORSHIP (Gateway.sol#13) is never used in gateway (Gateway.sol#5-45)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable
Gateway.sol analyzed (1 contracts with 84 detectors), 22 result(s) found
```

LOVELY INSPECTOR

SOLIDITY STATIC ANALYSIS

Static code analysis is used to identify many common coding problems before a program is released. It involves examining the code manually or using tools to automate the process. Static code analysis tools can automatically scan the code without executing it.

Channel.move

Gas costs:

Gas requirement of function channel.create_approved_call is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)
Pos: 33:5:

Constant/View/Pure functions:

channel.create_channel(uint256,uint256) : Potentially should be constant/view/pure but is not.
more
Pos: 26:5:

Constant/View/Pure functions:

channel.source_id(address) : Potentially should be constant/view/pure but is not.
more
Pos: 49:5:

LOVELY INSPECTOR

Utils.move

Constant/View/Pure functions:

Utils.to_sui_signed(address) : Potentially should be constant/view/pure but is not.

more

Pos: 13:5:

Constant/View/Pure functions:

Utils.operators_hash(address,uint256,uint256) : Potentially should be constant/view/pure but is not.

more

Pos: 17:5:

Gas costs:

Gas requirement of function Utils.operators_hash is infinite: If the gas requirement of a function is higher than the block gas limit, it cannot be executed. Please avoid loops in your functions or actions that modify large areas of storage (this includes clearing or copying arrays in storage)

Pos: 17:5:

LOVELY INSPECTOR

Validators.move

Constant/View/Pure functions:

validators.set_epoch(address) : Potentially should be constant/view/pure but is not.
more
Pos: 74:4:

Constant/View/Pure functions:

validators.epoch(address) : Potentially should be constant/view/pure but is not.
more
Pos: 78:5:

Gateway.move

Constant/View/Pure functions:

gateway.take_approved_call(address,address,uint256,uint256,address) : Potentially should be constant/view/pure but is not.
more
Pos: 26:5:

LOVELY INSPECTOR

COMPLIANCE ANALYSIS

Linters are the utility tools that analyze the given source code and report programming errors, bugs, and stylistic errors. For the Solidity language, there are some linter tools available that a developer can use to improve the quality of their Solidity contracts.

Channel.move

```
Compiler version ^0.8.1 does not satisfy the ^0.5.8 semver requirement
Pos: 1:3
Contract name must be in CamelCase
Pos: 1:4
Constant name must be in capitalized SNAKE_CASE
Pos: 6:6
Constant name must be in capitalized SNAKE_CASE
Pos: 6:7
Contract name must be in CamelCase
Pos: 5:11
Variable name must be in mixedCase
Pos: 8:12
Variable name must be in mixedCase
Pos: 8:13
Variable name must be in mixedCase
Pos: 8:18
Variable name must be in mixedCase
Pos: 8:19
Variable name must be in mixedCase
Pos: 8:20
Variable name must be in mixedCase
Pos: 8:21
Function name must be in mixedCase
Pos: 6:25
Variable name must be in mixedCase
Pos: 30:25
Variable name must be in mixedCase
Pos: 41:25
Code contains empty blocks
Pos: 67:25
Function name must be in mixedCase
Pos: 6:28
Variable name must be in mixedCase
Pos: 31:28
Code contains empty blocks
Pos: 55:28
Function name must be in mixedCase
Pos: 6:32
Variable name must be in mixedCase
Pos: 8:33
Variable name must be in mixedCase
```

Pos: 7:34
Variable name must be in mixedCase
Pos: 6:35
Variable name must be in mixedCase
Pos: 8:36
Code contains empty blocks
Pos: 14:38
Function name must be in mixedCase
Pos: 6:42
Variable name must be in mixedCase
Pos: 9:43
Code contains empty blocks
Pos: 14:45
Function name must be in mixedCase
Pos: 6:48
Variable name must be in mixedCase
Pos: 26:48
Code contains empty blocks
Pos: 50:48

Utils.move

Compiler version ^0.8.1 does not satisfy the ^0.5.8 semver requirement
Pos: 1:3
Constant name must be in capitalized SNAKE_CASE
Pos: 7:5
Constant name must be in capitalized SNAKE_CASE
Pos: 5:7
Function name must be in mixedCase
Pos: 6:8
Code contains empty blocks
Pos: 58:8
Function name must be in mixedCase
Pos: 6:12
Code contains empty blocks
Pos: 53:12
Function name must be in mixedCase
Pos: 6:16
Code contains empty blocks
Pos: 87:16

Validators.move

Compiler version ^0.8.1 does not satisfy the ^0.5.8 semver requirement
Pos: 1:3
Contract name must be in CamelCase
Pos: 1:4

Constant name must be in capitalized SNAKE_CASE
Pos: 6:6
Constant name must be in capitalized SNAKE_CASE
Pos: 6:7
Use double quotes for string literals
Pos: 51:7
Constant name must be in capitalized SNAKE_CASE
Pos: 6:8
Use double quotes for string literals
Pos: 51:8
Constant name must be in capitalized SNAKE_CASE
Pos: 6:9
Use double quotes for string literals
Pos: 53:9
Constant name must be in capitalized SNAKE_CASE
Pos: 6:10
Use double quotes for string literals
Pos: 53:10
Constant name must be in capitalized SNAKE_CASE
Pos: 6:11
Use double quotes for string literals
Pos: 54:11
Use double quotes for string literals
Pos: 51:13
Variable name must be in mixedCase
Pos: 9:16
Variable name must be in mixedCase
Pos: 9:22
Variable name must be in mixedCase
Pos: 9:31
Variable name must be in mixedCase
Pos: 19:35
Code contains empty blocks
Pos: 44:35
Function name must be in mixedCase
Pos: 6:40
Variable name must be in mixedCase
Pos: 37:40
Code contains empty blocks
Pos: 86:40
Function name must be in mixedCase
Pos: 6:44
Variable name must be in mixedCase
Pos: 9:45
Variable name must be in mixedCase
Pos: 8:46
Variable name must be in mixedCase
Pos: 9:47
Variable name must be in mixedCase
Pos: 8:48
Variable name must be in mixedCase
Pos: 9:49
Code contains empty blocks

LOVELY INSPECTOR

Pos: 13:50
Function name must be in mixedCase
Pos: 2:54
Variable name must be in mixedCase
Pos: 9:55
Variable name must be in mixedCase
Pos: 9:56
Variable name must be in mixedCase
Pos: 9:57
Variable name must be in mixedCase
Pos: 9:58
Variable name must be in mixedCase
Pos: 8:59
Code contains empty blocks
Pos: 15:60
Function name must be in mixedCase
Pos: 5:65
Variable name must be in mixedCase
Pos: 29:65
Code contains empty blocks
Pos: 62:65
Function name must be in mixedCase
Pos: 5:69
Variable name must be in mixedCase
Pos: 33:69
Code contains empty blocks
Pos: 66:69
Function name must be in mixedCase
Pos: 5:73
Variable name must be in mixedCase
Pos: 24:73
Code contains empty blocks
Pos: 57:73
Variable name must be in mixedCase
Pos: 21:77
Code contains empty blocks
Pos: 54:77

LOVELY INSPECTOR

Gateway.move

Compiler version ^0.8.1 does not satisfy the ^0.5.8 semver requirement
Pos: 1:3
Contract name must be in CamelCase
Pos: 1:4
Constant name must be in capitalized SNAKE_CASE
Pos: 6:5
Use double quotes for string literals
Pos: 51:5
Constant name must be in capitalized SNAKE_CASE
Pos: 5:7

Use double quotes for string literals
Pos: 49:7
Constant name must be in capitalized SNAKE_CASE
Pos: 5:9
Use double quotes for string literals
Pos: 46:9
Variable name must be in mixedCase
Pos: 9:16
Function name must be in mixedCase
Pos: 6:19
Variable name must be in mixedCase
Pos: 9:20
Code contains empty blocks
Pos: 14:21
Function name must be in mixedCase
Pos: 6:25
Variable name must be in mixedCase
Pos: 9:26
Variable name must be in mixedCase
Pos: 9:27
Variable name must be in mixedCase
Pos: 9:28
Variable name must be in mixedCase
Pos: 9:29
Variable name must be in mixedCase
Pos: 8:30
Code contains empty blocks
Pos: 13:31
Function name must be in mixedCase
Pos: 6:35
Variable name must be in mixedCase
Pos: 8:36
Variable name must be in mixedCase
Pos: 8:38
Code contains empty blocks
Pos: 14:39

LOVELY INSPECTOR

SOFTWARE ANALYSIS RESULT

This software reported many false positive results and some are informational issues. So, those issues can be safely ignored.

LOVELY INSPECTOR

INSPECTOR LOVELY

INFO

Website: Inspector.lovely.finance

Telegram community: t.me/inspectorlovely

Twitter: twitter.com/InspectorLovely

inspector.lovely.finance