



Consilience

Smart Contract Security Audit

Prepared by ShellBoxes

September 19th, 2022 - October 3rd, 2022

[Shellboxes.com](https://shellboxes.com)

contact@shellboxes.com

Document Properties

Client	Consilience Group Limited
Version	1.0
Classification	Public

Scope

The Consilience Contract in the Consilience Repository

Repo	Commit Hash
https://github.com/tableturn/tt-white-contracts	c301eefeee4bc9d111fb8757bb4f3a3ecf2e9c90

Files	MD5 Hash
marketplace/MarketplaceAccessFacet.sol	13df788e55901547cf1dc801fae14414
marketplace/MarketplaceInitFacet.sol	19560c6942dd3dd0d728bccc35fd69b6
marketplace/MarketplaceTokenHoldersFacet.sol	e5aef4fad710d83533cfdded3aa67ed1
marketplace/MarketplaceTopFacet.sol	977c3322cf604fe5a131ee026572269a
marketplace/lib/AMarketplaceFacet.sol	75d88531057c84c71e22aea296202071
marketplace/lib/IMarketplaceEvents.sol	34df7c70e1301c3c5e3ae9eb1785f9b2
marketplace/lib/LibMarketplace.sol	f9b8cbc23be4c531f4495c3bacf4dcc3
marketplace/lib/LibMarketplaceAccess.sol	a15801117aa01b591b57b025bbcb2f56

marketplace/lib/LibMarketplaceTokenHolders.sol	1c12a13359c47da543bdb2a0af3fa119
lib/LibAddressSet.sol	a5b0270111e90e7b3a4ee279b1ac6d73
lib/LibConstants.sol	436d7fb9ab3806a92f76109dbc91e392
lib/LibDiamond.sol	ea6a285ed94100d4db0f3df60a047ca2
lib/LibHelpers.sol	b8c18d7e9a0fd3b89d0ab08ff5f9d60e
lib/LibPaginate.sol	f631521eaafb54dee9a08b62fa70ac74
issuer/IssuerAccessFacet.sol	a21dc4bc5a51598297f96145c669bc00
issuer/IssuerFrontendFacet.sol	acbb8d719ce4b596e3ef0e025fc4a80c
issuer/IssuerInitFacet.sol	901e8af2f23483454979f4c40ecfd651
issuer/IssuerTopFacet.sol	9f472084d1479b4f425cde2124a59aa1
issuer/lib/IssuerFacet.sol	936461a36b83eaccb40800a363540647
issuer/lib/IssuerEvents.sol	94b82c5cf72559dced1c66f78b99098d
issuer/lib/LibIssuer.sol	d14dd750659ca2a183017d418f85f183
issuer/lib/LibIssuerAccess.sol	50f05d3bbcdb7d73d16231eba312a412
fast/FastAccessFacet.sol	7aa1770fec4958017c6c38a1063167ad
fast/FastFrontendFacet.sol	5654dc69424cfd79f460b7f0df644528
fast/FastHistoryFacet.sol	ed89b9179abb31c9685ccfe37eb32ffe
fast/FastInitFacet.sol	dd965459b83379b7eebf3d36814699e4
fast/FastTokenFacet.sol	850b81bd88e1163fdb936f5daa75de56
fast/FastTopFacet.sol	95545917c8a724d8624e920a449c05e3

fast/lib/AFastFacet.sol	d71dc5c444174bf26f86faa17e87bf47
fast/lib/IFast.sol	7e0241bf39fb8c1e690813fde31aa40c
fast/lib/IFastEvents.sol	eb023c68fc5b95d07bb92b3c321f9568
fast/lib/LibFast.sol	a6d67440b97122bd9d10d23d75a8b94c
fast/lib/LibFastAccess.sol	adb2c1b1dd5e08c0741d7b0bc2897d94
fast/lib/LibFastHistory.sol	daaf04ccd91967b7e8c41a86eef6267f
fast/lib/LibFastToken.sol	4aacbdb8141741a14474d5f6acaa9798

Re-Audit Scope

Repo	Commit Hash
https://github.com/tableturn/tt-white-contracts	f01917e8955c62a12310300b8afeb7990ce3111c

Files	MD5 Hash
marketplace/MarketplaceAccessFacet.sol	13df788e55901547cf1dc801fae14414
marketplace/MarketplaceInitFacet.sol	19560c6942dd3dd0d728bccc35fd69b6
marketplace/MarketplaceTokenHoldersFacet.sol	e5aef4fad710d83533cfdded3aa67ed1
marketplace/MarketplaceTopFacet.sol	977c3322cf604fe5a131ee026572269a
marketplace/lib/AMarketplaceFacet.sol	75d88531057c84c71e22aea296202071

marketplace/lib/IMarketplaceEvents.sol	34df7c70e1301c3c5e3ae9eb1785f9b2
marketplace/lib/LibMarketplace.sol	f9b8cbc23be4c531f4495c3bacf4dcc3
marketplace/lib/LibMarketplaceAccess.sol	a15801117aa01b591b57b025bbcb2f56
marketplace/lib/LibMarketplaceTokenHolders.sol	1c12a13359c47da543bdb2a0af3fa119
lib/LibAddressSet.sol	a5b0270111e90e7b3a4ee279b1ac6d73
lib/LibConstants.sol	436d7fb9ab3806a92f76109dbc91e392
lib/LibDiamond.sol	ea6a285ed94100d4db0f3df60a047ca2
lib/LibHelpers.sol	b8c18d7e9a0fd3b89d0ab08ff5f9d60e
lib/LibPaginate.sol	f631521eaafb54dee9a08b62fa70ac74
issuer/IssuerAccessFacet.sol	b1d2236361a10452130023ab2ca2310b
issuer/IssuerFrontendFacet.sol	acbb8d719ce4b596e3ef0e025fc4a80c
issuer/IssuerInitFacet.sol	bd6ea0d9be20be4ffa2b32a9ba73c8a9
issuer/IssuerTopFacet.sol	9f472084d1479b4f425cde2124a59aa1
issuer/lib/AIssuerFacet.sol	da2dea79c5398768e77e13a0af6a28cf
issuer/lib/IssuerEvents.sol	94b82c5cf72559dced1c66f78b99098d
issuer/lib/LibIssuer.sol	d14dd750659ca2a183017d418f85f183
issuer/lib/LibIssuerAccess.sol	50f05d3bbcdb7d73d16231eba312a412
fast/FastAccessFacet.sol	7aa1770fec4958017c6c38a1063167ad
fast/FastFrontendFacet.sol	a841e15dc1dc24b2a5b0ced055fde0fe
fast/FastHistoryFacet.sol	ed89b9179abb31c9685ccfe37eb32ffe

fast/FastInitFacet.sol	6b385f7c8cb56ddecff869de85e9ce89
fast/FastTokenFacet.sol	d2bece36135fb014840eedf2c1482381
fast/FastTopFacet.sol	2db04d42eda896addcead77a1bf465db
fast/lib/AFastFacet.sol	690746367e84baa41287f6bb1bbbd5d0
fast/lib/IFast.sol	7e0241bf39fb8c1e690813fde31aa40c
fast/lib/IFastEvents.sol	6a9a8aaf827eeec5815ab08578cb4ed0
fast/lib/LibFast.sol	a101f9667384839ac112c1bff9e83831
fast/lib/LibFastAccess.sol	adb2c1b1dd5e08c0741d7b0bc2897d94
fast/lib/LibFastHistory.sol	daaf04ccd91967b7e8c41a86eef6267f
fast/lib/LibFastToken.sol	d1343335d99e7272a29941337fd4e9b4

Contacts

COMPANY	EMAIL
ShellBoxes	contact@shellboxes.com

Contents

- 1 Introduction 9
 - 1.1 About Consilience Group Limited 9
 - 1.2 Approach & Methodology 9
 - 1.2.1 Risk Methodology 10
- 2 Findings Overview 11
 - 2.1 Summary 11
 - 2.2 Key Findings 11
- 3 Finding Details 12
 - A FastTokenFacet.sol 12
 - A.1 Two Members Can Lock All The Fast's Funds [HIGH] 12
 - A.2 The Issuer Can Retrieve Anyone's Tokens [MEDIUM] 13
 - B LibDiamond.sol 15
 - B.1 All The Upgrades Should Be Performed Using DAO [MEDIUM] 15
 - B.2 The Diamond Can End Up Without An Owner [LOW] 16
 - C IssuerAccessFacet.sol 17
 - C.1 One Issuer Member May Remove Any Other Issuer Member [MEDIUM] 17
 - D FastTokenFacet.sol 19
 - D.1 The Issuer Member Can Burn The Fast's Tokens [MEDIUM] 19
- 4 Attack Scenarios 21
 - 4.1 Issuer 21
 - 4.2 Fast 21
 - 4.3 Marketplace 24
- 5 Best Practices 26
 - BP.1 Unnecessary Initializations 26
 - BP.2 Unnecessary Argument 26
- 6 Tests 28
- 7 Coverage 41
- 8 Static Analysis (Slither) 44

9	Conclusion	58
10	Disclaimer	59

1 Introduction

Consilience Group Limited engaged ShellBoxes to conduct a security assessment on the Consilience beginning on September 19th, 2022 and ending October 3rd, 2022. In this report, we detail our methodical approach to evaluate potential security issues associated with the implementation of smart contracts, by exposing possible semantic discrepancies between the smart contract code and design document, and by recommending additional ideas to optimize the existing code. Our findings indicate that the current version of smart contracts can still be enhanced further due to the presence of many security and performance concerns.

This document summarizes the findings of our audit.

1.1 About Consilience Group Limited

Consilience Ventures is the first start-up market network to fully align the interests of start-ups, venture capital investors and experienced and talented experts to help turn innovative ideas into high-growth businesses.

Issuer	Consilience Group Limited
Website	https://consilience.vc/
Type	Solidity Smart Contract
Audit Method	Whitebox

1.2 Approach & Methodology

ShellBoxes used a combination of manual and automated security testing to achieve a balance between efficiency, timeliness, practicability, and correctness within the audit's scope. While manual testing is advised for identifying problems in logic, procedure, and implementation, automated testing techniques help to expand the coverage of smart contracts and can quickly detect code that does not comply with security best practices.

1.2.1 Risk Methodology

Vulnerabilities or bugs identified by ShellBoxes are ranked using a risk assessment technique that considers both the LIKELIHOOD and IMPACT of a security incident. This framework is effective at conveying the features and consequences of technological vulnerabilities.

Its quantitative paradigm enables repeatable and precise measurement, while also revealing the underlying susceptibility characteristics that were used to calculate the Risk scores. A risk level will be assigned to each vulnerability on a scale of 5 to 1, with 5 indicating the greatest possibility or impact.

- Likelihood quantifies the probability of a certain vulnerability being discovered and exploited in the untamed.
- Impact quantifies the technical and economic costs of a successful attack.
- Severity indicates the risk's overall criticality.

Probability and impact are classified into three categories: H, M, and L, which correspond to high, medium, and low, respectively. Severity is determined by probability and impact and is categorized into four levels, namely Critical, High, Medium, and Low.

Impact	High	Critical	High	Medium
	Medium	High	Medium	Low
	Low	Medium	Low	Low
		High	Medium	Low
		Likelihood		

2 Findings Overview

2.1 Summary

The following is a synopsis of our conclusions from our analysis of the Consilience implementation. During the first part of our audit, we examine the smart contract source code and run the codebase via a static code analyzer. The objective here is to find known coding problems statically and then manually check (reject or confirm) issues highlighted by the tool. Additionally, we check business logics, system processes, and DeFi-related components manually to identify potential hazards and/or defects.

2.2 Key Findings

In general, these smart contracts are well-designed and constructed, but their implementation might be improved by addressing the discovered flaws, which include , 1 high-severity, 4 medium-severity, 1 low-severity vulnerabilities.

Vulnerabilities	Severity	Status
A.1. Two Members Can Lock All The Fast's Funds	HIGH	Fixed
A.2. The Issuer Can Retrieve Anyone's Tokens	MEDIUM	Acknowledged
B.1. All The Upgrades Should Be Performed Using DAO	MEDIUM	Acknowledged
C.1. One Issuer Member May Remove Any Other Issuer Member	MEDIUM	Fixed
D.1. The Issuer Member Can Burn The Fast's Tokens	MEDIUM	Acknowledged
B.2. The Diamond Can End Up Without An Owner	LOW	Acknowledged

3 Finding Details

A FastTokenFacet.sol

A.1 Two Members Can Lock All The Fast's Funds [HIGH]

Description:

Each fast contains an attribute called `transferCredits`, the transfer credits are used to perform the transfer operation, whenever a fast is out of transfer credits, it becomes impossible for the member to transfer their tokens. However, this attribute depends on the fast itself and not the users, therefore the transfer credits are a shared resource between the fast members. Thus, two members can spend all the transfer credits of the fast resulting in a denial of service to all the members, locking their funds in the contract until the issuer member's intervention.

Code:

Listing 1: FastTokenFacet.sol

```
463 // If the funds are not moving from the zero address, decrease transfer
    ↪ credits.
464 if (p.from != address(0)) {
465     s.transferCredits -= p.amount;
466 }
```

Risk Level:

Likelihood - 5

Impact - 4

Recommendation:

It is recommended to attach the `transferCredits` attribute to the member instead of the fast itself, or to limit the transfer credits that can be spent by one member of the fast.

Status - Fixed

The Consilience team has fixed the issue by removing the transfer credits functionality.

A.2 The Issuer Can Retrieve Anyone's Tokens [MEDIUM]

Description:

The `retrieveDeadTokens` function allows the issuer to transfer all the tokens of any account to the reserve. This represents a significant centralization risk where the issuer has too much control over all the members' accounts.

Code:

Listing 2: FastTokenFacet.sol

```
117 function retrieveDeadTokens(address holder)
118     external
119     onlyIssuerMember {
120     // Cache how many tokens the holder has.
121     uint256 amount = balanceOf(holder);
122     // Note: The amount can be zero in this function.
123
124     // Grab a pointer to the token storage.
125     LibFastToken.Data storage s = LibFastToken.data();
126
127     // These should only run if the amount is zero, as they result in a
128     // ↔ no-op.
129     if (amount > 0) {
130     // Set the holder balance to zero.
131     s.balances[holder] = 0;
132     // Increment the reserve's balance.
133     s.balances[address(0)] += amount;
134     // The tokens aren't in circulation anymore - decrease total
135     // ↔ supply.
136     s.totalSupply -= amount;
```

```

135     }

137     // Since the holder's account is now empty, make sure to keep track
        ↳ of it both
138     // in this FAST and in the marketplace.
139     s.tokenHolders.remove(holder, true);
140     MarketplaceTokenHoldersFacet(LibFast.data().marketplace).
        ↳ fastBalanceChanged(holder, 0);

142     // This operation can be seen as a regular transfer between holder
        ↳ and reserve. Emit.
143     emit Transfer(holder, address(0), amount);

145     // If amount wasn't zero, total supply and reserve balance have
        ↳ changed - emit.
146     if (amount > 0) {
147         FastFrontendFacet(address(this)).emitDetailsChanged();
148     }
149 }

```

Risk Level:

Likelihood - 2

Impact - 5

Recommendation:

Consider using a multisig wallet or a DAO as the address of the issuer in order to avoid centralization risks and include multiple parties in the decision-making.

Status - Acknowledged

The Consilience team has acknowledged the issue, stating that the functionality cannot be changed due to regulatory restrictions under the [Financial Conduct Authority \(FCA\)](#) requirements in article [SUP 10A.10 Customer-dealing functions](#).

B LibDiamond.sol

B.1 All The Upgrades Should Be Performed Using DAO [MEDIUM]

Description:

The contracts implement Tokenization as a Service using the diamond pattern, this implementation offers the upgradability of functionalities along with other advantages. However, by using this approach the contract owner can upgrade the contract to implement any logic, this represents a centralization risk knowing that the contract owner can change the logic of the code anytime which can cause unexpected behaviors to the users.

Code:

Listing 3: LibDiamond.sol

```
65 function diamondCut(  
66     IDiamondCut.FacetCut[] memory _diamondCut,  
67     address _init,  
68     bytes memory _calldata  
69 ) internal {  
70     for (uint256 facetIndex; facetIndex < _diamondCut.length; facetIndex  
71         ↪ ++ ) {  
72         IDiamondCut.FacetCutAction action = _diamondCut[facetIndex].  
73             ↪ action;  
74         if (action == IDiamondCut.FacetCutAction.Add) {  
75             addFunctions(_diamondCut[facetIndex].facetAddress,  
76                 ↪ _diamondCut[facetIndex].functionSelectors);  
77         } else if (action == IDiamondCut.FacetCutAction.Replace) {  
78             replaceFunctions(_diamondCut[facetIndex].facetAddress,  
79                 ↪ _diamondCut[facetIndex].functionSelectors);  
80         } else if (action == IDiamondCut.FacetCutAction.Remove) {  
81             removeFunctions(_diamondCut[facetIndex].facetAddress,  
82                 ↪ _diamondCut[facetIndex].functionSelectors);  
83         }  
84     }  
85 }
```

```
78     } else {
79         revert("LibDiamondCut: Incorrect FacetCutAction");
80     }
81 }
82 emit DiamondCut(_diamondCut, _init, _calldata);
83 initializeDiamondCut(_init, _calldata);
84 }
```

Risk Level:

Likelihood - 1

Impact - 4

Recommendation:

Consider using a DAO as the contract owner in order to include the community in the decision of upgrades.

Status - Acknowledged

The Consilience team has acknowledged the issue, stating that the functionality cannot be changed due to regulatory restrictions under the [Financial Conduct Authority \(FCA\)](#) requirements in article [SUP 10A.10 Customer-dealing functions](#).

B.2 The Diamond Can End Up Without An Owner [LOW]

Description:

The `setContractOwner` allows the contract owner to delegate the ownership to another address. However, if the contract owner enters a wrong address or the `address(0)` as the `_newOwner`, the contract will have no owner, which results in a Denial Of Service in the privileged actions.

Code:

Listing 4: LibDiamond.sol

```
47 function setContractOwner(address _newOwner) internal {
48     DiamondStorage storage ds = diamondStorage();
49     address previousOwner = ds.contractOwner;
50     ds.contractOwner = _newOwner;
51     emit OwnershipTransferred(previousOwner, _newOwner);
52 }
```

Risk Level:

Likelihood - 1

Impact - 3

Recommendation:

Consider requiring the `_newOwner` to be different from the `address(0)`, and it is recommended to implement a process where the contract owner sets an address as a new owner candidate, then this address can only be the contract owner if it calls the contract to accept the ownership.

Status - Acknowledged

The Consilience team has acknowledged the issue, stating that they want ownership to be transferable to the `address(0)` as a proof of future immutability.

C IssuerAccessFacet.sol

C.1 One Issuer Member May Remove Any Other Issuer Member [MEDIUM]

Description:

The `removeMember` function can be called by any member. Allowing a member to remove another one with the same privilege can harm the logic of the contract. For in instance, a

new issuer member will be able to remove all the other members.

Code:

Listing 5: IssuerAccessFacet.sol

```
73 function removeMember(address member)
74     external override
75     onlyMember(msg.sender) {
76     // No suicide allowed.
77     if (msg.sender == member) {
78         revert ICustomErrors.CannotSelfRemove(msg.sender);
79     }
80     // Remove the member from the set.
81     LibIssuerAccess.data().memberSet.remove(member, false);
82     // Emit!
83     emit MemberRemoved(member);
84 }
```

Risk Level:

Likelihood - 2

Impact - 4

Recommendation:

Consider restricting the functionality to one master issuer member that will have a higher privilege over the other members.

Status - Fixed

The Consilience team has fixed the issue by only allowing the diamond owner to remove the issuer members.

D FastTokenFacet.sol

D.1 The Issuer Member Can Burn The Fast's Tokens [MEDIUM]

Description:

Anyone of the issuer members is able to burn tokens from the reserve, this action can represent a significant centralization risk where the issuer members have too much control over the fast's funds.

Code:

Listing 6: FastTokenFacet.sol

```
79 function burn(uint256 amount, string calldata ref)
80     external
81     onlyIssuerMember {
82     LibFastToken.Data storage s = LibFastToken.data();

84     if (FastTopFacet(address(this)).hasFixedSupply()) {
85         revert ICustomErrors.RequiresContinuousSupply();
86     }

88     // Remove the minted amount from the zero address.
89     s.balances[address(0)] -= amount;

91     // Keep track of the minting operation.
92     FastHistoryFacet(address(this)).burnt(amount, ref);

94     // Emit!
95     FastFrontendFacet(address(this)).emitDetailsChanged();
96     emit Burnt(amount, ref);
97 }
```

Risk Level:

Likelihood – 2

Impact – 4

Recommendation:

Consider adding some restrictions to the burn functionality to limit the power of the issuer members over the fasts.

Status – Acknowledged

The Consilience team has acknowledged the issue, stating that the functionality cannot be changed due to regulatory restrictions under the [Financial Conduct Authority \(FCA\)](#) requirements in article [SUP 10A.10 Customer-dealing functions](#).

4 Attack Scenarios

4.1 Issuer

🔒 Registering a fast without being an issuer member

❌ FAILED

💡 Protection by the `onlyMember` modifier

🔒 Adding/Removing Issuer members without being an issuer member

❌ FAILED

💡 Protection by the `onlyMember` modifier

🔒 Adding/Removing Issuer members being an issuer member

✅ PASSED

💡 The `onlyMember` modifier is not enough as all the members have the same privilege over each other

🔒 Adding/Removing a fast governor directly from the `fastGovernorships` mapping in the `Issuer` diamond without calling the `Fast`

❌ FAILED

💡 Having in place a check that makes sure `governorAddedToFast` and `governorRemovedFromFast` can only be called by a registered `Fast`

4.2 Fast

🔒 Minting an amount without being an issuer member

❌ FAILED

💡 Protection by the `onlyIssuerMember` modifier

🚫 Minting more than one time in a fast that has a fixed supply

✗ FAILED

💡 Having in place a check that makes sure the mint can only be performed one time in a fixed supply fast.

🚫 Burning an amount without being an issuer member

✗ FAILED

💡 Protection by the `onlyIssuerMember` modifier

🚫 Burning in a fast that has a fixed supply

✗ FAILED

💡 Having a check in place that makes sure the burn can only be performed in a continuous supply fast.

🚫 Causing an overflow/underflow and generating unexpected results

✗ FAILED

💡 Built in overflow protection in solidity 0.8.* versions

🚫 Approve Race Condition

✗ FAILED

💡 The allowance is incremented instead of being set to amount

🚫 Transferring tokens to a non-fast-member in a private fast

✗ FAILED

💡 The `onlyTokenHolder` modifier requires the `from` and `to` to be fast members when the fast is private

🚫 Transferring tokens to a non-marketplace-member in a semi public fast

✗ FAILED

💡 The `onlyTokenHolder` modifier requires the `to` to be a marketplace member when the fast is semi public

🚫 Transferring tokens from the reserve without being a fast governor

✗ FAILED

💡 The transfer function verifies the spender to be a fast governor whenever the `from` is the reserve address

🚫 Performing a transfer without having enough transfer credits

✗ FAILED

💡 The transfer call fails due to underflow protection

🚫 Changing a fast from semi public to private

✗ FAILED

💡 The `setIsSemiPublic` function prevents the change when a fast is semi public

🚫 Adding/Removing governors without being an issuer member

✗ FAILED

💡 Protection by the `onlyIssuerMember` modifier

🚫 Adding/Removing fast members without being a governor

✗ FAILED

💡 Protection by the `onlyGovernor` modifier

🚫 Adding a non-marketplace-member as a fast member

✗ FAILED

💡 Protection by `onlyMarketplaceMember` modifier

🚫 Spending all the transfer credits by one member

✗ FAILED

💡 The member is limited with his token balance

🚫 Spending all the transfer credits by two members

✓ PASSED

💡 Two members can spend all the transfer credits by performing multiple transfer calls between each other

4.3 Marketplace

🚫 Adding/Removing marketplace members without being an issuer member

✗ FAILED

💡 Protection by the `onlyIssuerMember` modifier

🚫 Adding/Removing a fast member directly from the `fastMemberships` mapping in the `Marketplace` diamond without calling the `Fast`

✗ FAILED

💡 Having in place a check that makes sure `memberAddedToFast` and `memberRemovedFromFast` can only be called by a registered `Fast`

✖ Manipulating the `fastHoldings` mapping in the `Marketplace` diamond without calling the `Fast`

✖ FAILED

💡 Having in place a check that makes sure `fastBalanceChanged` can only be called by a registered `Fast`

Conclusion:

The results of the attack scenarios are mentioned in the Finding Details section, the contracts were also tested for reentrancy attacks, front running attacks, Block Timestamp manipulation... , and no issues were found.

5 Best Practices

BP.1 Unnecessary Initializations

Description:

When a variable is declared in solidity, it gets initialized with its type's default value. Thus, there is no need to initialize a variable with the default value.

Code:

Listing 7: FastInitFacet.sol

```
87 tokenData.totalSupply = 0;
88 // Initialize other internal stuff.
89 tokenData.transferCredits = 0;
```

BP.2 Unnecessary Argument

Description:

The `setIsSemiPublic` function is used to change the fast's type. However, the fast's type can only be changed from private to semi-public, therefore the `flag` argument is unnecessary since the `isSemiPublic` attribute can be set directly to `true` in order to change the fast's type from private to semi-public.

Code:

Listing 8: FastTopFacet.sol

```
57 function setIsSemiPublic(bool flag)
58     external
59     onlyIssuerMember { // can be set to true
60 // Someone is trying to toggle back to private?... No can do!
61 if (this.isSemiPublic()) {
```

```
62     revert ICustomErrors.UnsupportedOperation();
63 }
64 LibFast.data().isSemiPublic = flag;
65 // Emit!
66 FastFrontendFacet(address(this)).emitDetailsChanged();
67 }
```

6 Tests

Results:

FastAccessFacet

IHasGovernors implementation

isGovernor

returns true when the address is a governor

returns false when the address is not a governor

governorCount

returns the current count of governors

paginateGovernors

returns the cursor to the next page

does not crash when overflowing and returns the correct cursor

returns the governors in the order they were added

addGovernor

requires Issuer membership (anonymous) (42ms)

requires Issuer membership (governor)

delegates to the Issuer for permission checking

requires that the address is an Marketplace member

requires that the address is not a governor yet (52ms)

adds the given address as a governor (49ms)

calls FastFrontendFacet.emitDetailsChanged

emits a GovernorAdded event

removeGovernor

requires Issuer membership (anonymous)

requires Issuer membership (governor)

delegates to the Issuer for permission checking

requires that the address is an existing governor

removes the given address as a governor (51ms)

calls FastFrontendFacet.emitDetailsChanged

emits a GovernorRemoved event

IHasMembers

isMember

returns true when the `address` is a member
returns false when the `address` is not a member

`memberCount`
returns the current count of members

`paginateMembers`
returns the cursor to the next page
does not crash when overflowing and returns the correct cursor
returns the members in the order they were added

`addMember`
requires governance (anonymous)
requires governance (Issuer governor)
requires that the `address` is an Marketplace member
requires that the `address` is not a member yet (57ms)
adds the given `address` as a member (43ms)
delegates to the Marketplace contract to signal the membership
↳ addition
calls `FastFrontendFacet.emitDetailsChanged`
emits a `MemberAdded` event

`removeMember`
requires governance (anonymous)
requires governance (Issuer governor)
requires that the `address` is an existing member
removes the given `address` as a member
delegates to the token contract (40ms)
delegates to the Marketplace contract to signal the membership
↳ addition
calls `FastFrontendFacet.emitDetailsChanged` (41ms)
emits a `MemberRemoved` event

`flags`
is accurate when all flags set (83ms)
is accurate when only `isGovernor` is set (39ms)
is accurate when only `isMember` is set (64ms)
is accurate when no flags are set (42ms)

FastFrontendFacet

emitDetailsChanged

requires that the caller is the diamond

emits a DetailsChanged event with all the correct information details

returns a populated details struct

detailedMember

returns a MemberDetails struct with the correct information

detailedGovernor

returns a GovernorDetails struct with the correct information

paginateDetailedMembers

returns member details with next cursor

handles an offset index cursor

paginateDetailedGovernors

returns governor details with next cursor

FastHistoryFacet

minted

requires that the caller is the token (anonymous)

requires that the caller is the token (governor)

as the diamond

adds an entry to the supply proof list

burnt

requires that the caller is the diamond (anonymous)

requires that the caller is the diamond (governor)

as the diamond

adds an entry to the supply proof list

supplyProofCount

counts how many supply proofs have been stored

paginateSupplyProofs

returns the cursor to the next page

does not crash when overflowing and returns the correct cursor

returns the supply proofs in the order they were added

transferred

requires that the caller is the token (anonymous)
requires that the caller is the token (governor)
adds an entry to the transfer proof list (76ms)

transferProofCount
counts how many transfer proofs have been stored (88ms)

paginateTransferProofs
returns the cursor to the next page
does not crash when overflowing and returns the correct cursor
returns the transfer proofs in the order they were added

paginateTransferProofsByInvolvee
returns the cursor to the next page
does not crash when overflowing and returns the correct cursor (
↔ bob)
does not crash when overflowing and returns the correct cursor (
↔ john)
- counts the proofs regardless of the involvement (sender and
↔ recipient)
categorizes the proofs for the senders

paginateTransferProofIndicesByInvolvee
returns a paginated list of addresses and cursor

transferProofByInvolveeCount
returns the count of the transfer proofs for a given address

FastTokenFacet

initialize
keeps track of the ERC20 parameters and extra ones (40ms)

name
returns the name

symbol
returns the symbol

decimals
returns the decimals

totalSupply
returns the total supply

transferCredits
 returns the remaining transfer credits

mint
 requires Issuer membership (anonymous)
 requires Issuer membership (member)
 requires Issuer membership (governor)
 delegates to the history contract (48ms)
 adds the minted tokens to the zero address (59ms)
 does not impact total supply (61ms)
 emits a Minted event (53ms)
- delegates to the frontend facet

with fixed supply
 is allowed only once (73ms)

with continuous supply
 is allowed more than once (99ms)

burn
 requires Issuer membership (anonymous)
 requires Issuer membership (member)
 requires Issuer membership (governor)
 requires that the supply is continuous
 requires that the zero address has enough funds
 removes tokens from the zero address (56ms)
 does not impact total supply (48ms)
 delegates to the history contract
 emits a Burnt event (38ms)
- delegates to the frontend facet

retrieveDeadTokens
 requires Issuer membership
 still emits a Transfer event if the balance was already zero
 sets the holder balance to zero while increasing the reserve
 ↔ balance (66ms)
 decreases the total supply by the amount (47ms)
 removes the holder from the FAST token holder list (43ms)
 calls the marketplace to stop tracking this token holder for this

↳ FAST

emits a `Transfer` event between the holder and the reserve
delegates to the Frontend facet for a global event emission (40ms)

`addTransferCredits`

requires Issuer membership (`anonymous`)
requires Issuer membership (`member`)
requires Issuer membership (`governor`)
accumulates the credits to the existing `transfer` credits (103ms)
emits a `TransferCreditsAdded` event
- delegates to the frontend facet

`drainTransferCredits`

requires Issuer membership (`anonymous`)
requires Issuer membership (`member`)
requires Issuer membership (`governor`)
sets the credit amount to zero (61ms)
emits a `TransferCreditsDrained` event (38ms)

ERC20

`balanceOf`

returns the amount of tokens at a given `address`

`transfer`

delegates to the `internal` `performTransfer` method (115ms)

`transferWithRef`

delegates to the `internal` `performTransfer` method (78ms)

`allowance`

returns the allowance for a given member (44ms)

follows `value` at zero `address` for governors (76ms)

`approve`

delegates to the `internal` `performApproval` method

requires FAST membership

adds an allowance with the correct parameters (117ms)

functions properly when given a zero amount (97ms)

stacks up `new` allowances (105ms)

keeps track of given allowances (47ms)

keeps track of received allowances (48ms)

- emits an Approval event
- disapprove
 - delegates to the internal Disapproval method
 - subtracts from the existing allowance
 - emits a Disapproval event
 - when the allowance remains positive after the operation
 - removes the spender received allowance when it reaches zero
 - removes the original given allowance when it reaches zero
 - when the allowance reaches zero
 - removes the spender received allowance when it reaches zero
 - removes the original given allowance when it reaches zero
- transferFrom
 - delegates to the internal performTransfer method (79ms)
- transferFromWithRef
 - delegates to the internal performTransfer method (85ms)
 - decreases the transfer credits when not transacting from the
 - ↪ zero address (128ms)
 - requires that the sender and recipient are different
 - requires sufficient funds (83ms)
 - requires sufficient transfer credits (129ms)
 - transfers from / to the given wallet address (132ms)
 - delegates to the history contract (76ms)
 - delegates to the MarketplaceTokenHoldersFacet contract (76ms)
 - updates who holds this token (87ms)
 - decreases total supply when transferring to the zero address
 - ↪ (112ms)
 - emits a IERC20.Transfer event (90ms)
 - requires that there is enough allowance (52ms)
 - allows non-members to transact on behalf of members (189ms)
 - increases total supply when transferring from the zero address
 - ↪ (96ms)
 - requires that zero address can only be spent from as a governor
 - ↪ (Issuer member) (48ms)
 - requires that zero address can only be spent from as a governor

↪ (member) (43ms)
 requires that zero **address** can only be spent **from** as a governor
 ↪ (anonymous) (120ms)
 allows governors to **transfer from** the zero **address** (151ms)
 does not **require transfer** credits when drawing **from** the zero
 ↪ **address** (126ms)
 does not impact **transfer** credits when drawing **from** the zero
 ↪ **address** (165ms)
 when member deactivated
 requires active member when transferring **from address** (at the
 ↪ Marketplace level) (55ms)
 allows **transfer** to a deactivated member (at the Marketplace
 ↪ level) (155ms)
 when semi-public
 requires **sender** membership (Marketplace membership)
 requires recipient membership (Marketplace membership) (40ms)
 allows marketplace members to transact (169ms)
 when private
 requires **sender** membership (FAST member)
 requires recipient membership (FAST member) (54ms)
 givenAllowanceCount
 returns the count of allowancesByOwner
 paginateAllowancesByOwner
 returns the list of addresses to which the caller gave allowances
 ↪ to
 does not list addresses **from** which the caller has received
 ↪ allowances
 receivedAllowanceCount
 returns the count of allowancesBySpender
 paginateAllowancesBySpender
 returns the list of addresses to which the caller gave allowances
 ↪ to
 does not list addresses to which the caller has given allowances
 beforeRemovingMember

cannot be called directly
when successful
reverts if the member to remove still has a positive balance
↪ (125ms)
sets allowances to / from the removed members to zero (108ms)
removes given and received allowances (74ms)
emits a Disapproval event as many times as it removed allowance
↪ (68ms)

FastTopFacet

issuerAddress
returns the Issuer address
marketplaceAddress
returns the marketplace address
isSemiPublic
returns the FAST semi-public parameter
hasFixedSupply
returns the FAST fixed supply parameter
setIsSemiPublic
requires Issuer membership for the sender
delegates to the Issuer for permission check (39ms)
prevents changing from semi-public to closed (73ms)
sets the required flag on the FAST (47ms)
delegates to FastFrontendFacet.emitDetailsChanged (41ms)

IssuerAccessFacet

IHasMembers
isMember
returns true when the candidate is a member
returns false when the candidate is not a member
memberCount
correctly counts members
paginateMembers
returns pages of members (54ms)

addMember

- requires that the `sender` is a member
- adds the member to the list
- does not add the same member twice
- emits a `MemberAdded` event

removeMember

- requires that the `sender` is a member
- requires that the user is not removing themselves
- removes the member `from` the list
- reverts `if` the member is not in the list
- emits a `MemberRemoved` event

governorAddedToFast

- requires the caller to be a registered FAST
- adds the given member to the FAST governorship tracking `data`
 - ↳ structure (66ms)
- emits `GovernorshipAdded` event (55ms)

governorRemovedFromFast

- requires the caller to be a registered FAST
- adds the given member to the FAST governorship tracking `data`
 - ↳ structure (92ms)
- emits `GovernorshipRemoved` event (77ms)

paginateGovernorships

- given an `address`, returns the list of FASTs that it is a governor
 - ↳ of

IssuerFrontendFacet

paginateDetailedFasts

- returns a paginated list of detailed FAST details (45ms)

IssuerInitFacet

initialize

- requires that it is not initialized
- set various `storage` versions
- registers supported interfaces (50ms)

adds the given `address` to the member list
emits a `MemberAdded` event

IssuerTopFacet

FAST management

`isFastRegistered`

returns `false` when the FAST symbol is unknown
returns `true` when the FAST symbol is registered

`fastBySymbol`

returns the zero `address` when the FAST symbol is unknown
returns the FAST `address` when the FAST symbol is registered

`registerFast`

requires Issuer membership
reverts if trying to add a FAST with an already existing symbol
↔ (50ms)
adds the registry `address` to the list of registries
keeps track of the symbol
emits a `FastRegistered` event

`fastCount`

returns the FAST count

`paginateFasts`

returns pages of FASTs

MarketplaceAccessFacet

IHasMembers

`isMember`

returns `true` when the `address` is a member
returns `false` when the `address` is not a member

`memberCount`

returns the current count of members

`paginateMembers`

returns the cursor to the next page
does not crash when overflowing and returns the correct cursor
returns the governors in the order they were added

addMember
requires Issuer membership (anonymous)
delegates to the Issuer for permission (42ms)
requires that the address is not a member yet (76ms)
adds the given address as a member (59ms)
emits a MemberAdded event

removeMember
requires Issuer membership (anonymous)
delegates to the Issuer for permission
requires that the address is an existing member - calls
↳ LibAddressSet
requires that the given member has no FAST memberships (85ms)
removes the given address as a member (59ms)
emits a MemberRemoved event (40ms)

fastMemberships
returns an array of FASTs a given user belongs to along with a
↳ cursor
does not return FASTs the given user does not belong to

memberAddedToFast
requires the caller to be a registered FAST
adds the given member to the FAST membership tracking data
↳ structure (62ms)

memberRemovedFromFast
requires the caller to be a registered FAST
removes the FAST contract from the list of Fast members (95ms)

isMemberActive
returns true when a member is active
returns false when a member is deactivated

deactivateMember
requires the caller to be an Issuer member
requires the member to deactivate is an Marketplace member
adds the FAST member to the list of deactivated members (65ms)
emits a MemberDeactivated event (49ms)
requires that a given member is not already deactivated (94ms)

activateMember

requires the caller to be an Issuer member

requires the member to activate is an Marketplace member

removes the FAST member from the list of deactivated members (70ms

↪)

emits a MemberActivated event (50ms)

requires that a given member is currently deactivated (79ms)

MarketplaceInitFacet

initialize

requires that it is not initialized (147ms)

set various storage versions

registers supported interfaces (92ms)

stores the given Issuer address

MarketplaceTokenHoldersFacet

holdingsUpdated

reverts if not called by a FAST contract

returns a list of FASTs that an account holds (64ms)

removes the FAST holding if account balance drops to 0 (93ms)

does not track the zero address (46ms)

holdings

returns a list of FASTs a account holds (60ms)

MarketplaceTopFacet

issuerAddress

returns the Issuer address

270 passing (2m)

4 pending

7 Coverage

Results:

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
fast/	100	91.07	100	98.68	
FastAccessFacet.sol	100	100	100	100	
FastFrontendFacet.sol	100	100	100	100	
FastHistoryFacet.sol	100	100	100	100	
FastInitFacet.sol	100	50	100	96	47
FastTokenFacet.sol	100	90.24	100	98.43	283,428
FastTopFacet.sol	100	100	100	100	
fast/lib/	88.89	77.78	92.31	83.87	
AFastFacet.sol	88.89	77.78	88.89	81.48	36,37,39,45,99
IFast.sol	100	100	100	100	
IFastEvents.sol	100	100	100	100	
LibFast.sol	100	100	100	100	
LibFastAccess.sol	100	100	100	100	
LibFastHistory.sol	100	100	100	100	
LibFastToken.sol	100	100	100	100	
interfaces/	100	100	100	100	
ICustomErrors.sol	100	100	100	100	
IDiamondCut.sol	100	100	100	100	
IDiamondLoupe.sol	100	100	100	100	
IERC1404.sol	100	100	100	100	
IERC165.sol	100	100	100	100	
IERC173.sol	100	100	100	100	
IERC20.sol	100	100	100	100	
IHasActiveMembers.sol	100	100	100	100	
IHasGovernors.sol	100	100	100	100	
IHasMembers.sol	100	100	100	100	
issuer/	100	88.89	100	97.96	

```

IssuerAccessFacet.sol | 100 | 90 | 100 | 94.44 | 108 |
IssuerFrontendFacet.sol | 100 | 100 | 100 | 100 | |
IssuerInitFacet.sol | 100 | 75 | 100 | 100 | |
IssuerTopFacet.sol | 100 | 100 | 100 | 100 | |
issuer/lib/ | 66.67 | 50 | 80 | 63.64 | |
AIssuerFacet.sol | 66.67 | 50 | 66.67 | 55.56 | 24,25,27,33 |
IIssuerEvents.sol | 100 | 100 | 100 | 100 | |
LibIssuer.sol | 100 | 100 | 100 | 100 | |
LibIssuerAccess.sol | 100 | 100 | 100 | 100 | |
lib/ | 29.13 | 24.29 | 40 | 32.54 | |
LibAddressSet.sol | 100 | 100 | 100 | 100 | |
LibConstants.sol | 100 | 100 | 100 | 100 | |
LibDiamond.sol | 1.35 | 0 | 7.69 | 2.3 | ... 201,202,205 |
LibHelpers.sol | 100 | 100 | 100 | 100 | |
LibPaginate.sol | 100 | 87.5 | 100 | 100 | |
marketplace/ | 100 | 94.12 | 100 | 100 | |
MarketplaceAccessFacet.sol | 100 | 100 | 100 | 100 | |
MarketplaceInitFacet.sol | 100 | 75 | 100 | 100 | |
MarketplaceTokenHoldersFacet.sol | 100 | 87.5 | 100 | 100 | |
MarketplaceTopFacet.sol | 100 | 100 | 100 | 100 | |
marketplace/lib/ | 100 | 83.33 | 100 | 91.67 | |
AMarketplaceFacet.sol | 100 | 83.33 | 100 | 88.89 | 27 |
IMarketplaceEvents.sol | 100 | 100 | 100 | 100 | |
LibMarketplace.sol | 100 | 100 | 100 | 100 | |
LibMarketplaceAccess.sol | 100 | 100 | 100 | 100 | |
LibMarketplaceTokenHolders.sol | 100 | 100 | 100 | 100 | |

```

Conclusion:

The code coverage results were obtained by running `npx hardhat coverage`. We found the :

- Statements Coverage : **79.67%**
- Branches Coverage : **80.47%**
- Functions Coverage : **71.59%**
- Lines Coverage : **89.71%**

File	Statements	Branches	Functions	Lines
fast/	100%	182/182	91.07%	102/112
fast/lib/	88.89%	8/9	77.78%	14/18
interfaces/	100%	0/0	100%	0/0
issuer/	100%	34/34	88.89%	16/18
issuer/lib/	66.67%	2/3	50%	3/6
lib/	29.13%	30/103	24.29%	17/70
marketplace/	100%	35/35	94.12%	32/34
marketplace/lib/	100%	3/3	83.33%	5/6

8 Static Analysis (Slither)

Description:

ShellBoxes expanded the coverage of the specific contract areas using automated testing methodologies. Slither, a Solidity static analysis framework, was one of the tools used. Slither was run on all-scoped contracts in both text and binary formats. This tool can be used to test mathematical relationships between Solidity instances statically and variables that allow for the detection of errors or inconsistent usage of the contracts' APIs throughout the entire codebase.

Results:

```
'npx hardhat compile --force' running
Generating typings for: 45 artifacts in dir: typechain for target:
  ↳ ethers-v5
Successfully generated 90 typings!
Compiled 45 Solidity files successfully

LibDiamond.diamondCut(IDiamondCut.FacetCut [],address,bytes).facetIndex (
  ↳ contracts/lib/LibDiamond.sol#70) is a local variable never
  ↳ initialized
LibDiamond.addFunctions(address,bytes4[]).selectorIndex (contracts/lib/
  ↳ LibDiamond.sol#95) is a local variable never initialized
LibDiamond.removeFunctions(address,bytes4[]).selectorIndex (contracts/
  ↳ lib/LibDiamond.sol#128) is a local variable never initialized
LibDiamond.replaceFunctions(address,bytes4[]).selectorIndex (contracts/
  ↳ lib/LibDiamond.sol#113) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation
  ↳ #uninitialized-local-variables
FastInitFacet.initialize(FastInitFacet.InitializerParams) (contracts/
  ↳ fast/FastInitFacet.sol#42-90) ignores return value by
  ↳ ICustomErrors.AlreadyInitialized() (contracts/fast/FastInitFacet.
  ↳ sol#47)
```

FastTokenFacet.mint(uint256,string) (contracts/fast/FastTokenFacet.sol
↳ #41-61) ignores return value by ICustomErrors.
↳ RequiresContinuousSupply() (contracts/fast/FastTokenFacet.sol#49)

FastTokenFacet.burn(uint256,string) (contracts/fast/FastTokenFacet.sol
↳ #79-97) ignores return value by ICustomErrors.
↳ RequiresContinuousSupply() (contracts/fast/FastTokenFacet.sol#85)

FastTokenFacet.allowance(address,address) (contracts/fast/FastTokenFacet
↳ .sol#276-288) ignores return value by ICustomErrors.
↳ RequiresFastGovernorship(spender) (contracts/fast/FastTokenFacet.
↳ sol#283)

FastTokenFacet.performTransfer(FastTokenFacet.TransferArgs) (contracts/
↳ fast/FastTokenFacet.sol#416-486) ignores return value by
↳ ICustomErrors.UnsupportedOperation() (contracts/fast/
↳ FastTokenFacet.sol#428)

FastTokenFacet.performTransfer(FastTokenFacet.TransferArgs) (contracts/
↳ fast/FastTokenFacet.sol#416-486) ignores return value by
↳ ICustomErrors.RequiresFastGovernorship(p.spender) (contracts/fast
↳ /FastTokenFacet.sol#432)

FastTokenFacet.beforeRemovingMember(address) (contracts/fast/
↳ FastTokenFacet.sol#560-587) ignores return value by ICustomErrors
↳ .RequiresPositiveBalance(member) (contracts/fast/FastTokenFacet.
↳ sol#563)

FastTokenFacet.onlyTokenHolder(address) (contracts/fast/FastTokenFacet.
↳ sol#628-645) ignores return value by ICustomErrors.
↳ RequiresMarketplaceMembership(candidate) (contracts/fast/
↳ FastTokenFacet.sol#634)

FastTokenFacet.onlyTokenHolder(address) (contracts/fast/FastTokenFacet.
↳ sol#628-645) ignores return value by ICustomErrors.
↳ RequiresFastMembership(candidate) (contracts/fast/FastTokenFacet.
↳ sol#640)

FastTopFacet.setIsSemiPublic(bool) (contracts/fast/FastTopFacet.sol
↳ #57-67) ignores return value by ICustomErrors.
↳ UnsupportedOperation() (contracts/fast/FastTopFacet.sol#62)

`AFastFacet.onlyDiamondFacet()` (`contracts/fast/lib/AFastFacet.sol#27-32`)
 ↪ ignores return value by `ICustomErrors.InternalMethod()` (`contracts`
 ↪ `/fast/lib/AFastFacet.sol#29`)

`AFastFacet.onlyDiamondOwner()` (`contracts/fast/lib/AFastFacet.sol#35-40`)
 ↪ ignores return value by `ICustomErrors.RequiresDiamondOwnership()`
 ↪ (`contracts/fast/lib/AFastFacet.sol#37`)

`AFastFacet.onlyDeployer()` (`contracts/fast/lib/AFastFacet.sol#43-48`)
 ↪ ignores return value by `ICustomErrors.InternalMethod()` (`contracts`
 ↪ `/fast/lib/AFastFacet.sol#45`)

`AFastFacet.onlyMarketplaceMember(address)` (`contracts/fast/lib/AFastFacet`
 ↪ `.sol#54-59`) ignores return value by `ICustomErrors.`
 ↪ `RequiresMarketplaceMembership(candidate)` (`contracts/fast/lib/`
 ↪ `AFastFacet.sol#56`)

`AFastFacet.onlyMarketplaceActiveMember(address)` (`contracts/fast/lib/`
 ↪ `AFastFacet.sol#65-70`) ignores return value by `ICustomErrors.`
 ↪ `RequiresMarketplaceActiveMember(candidate)` (`contracts/fast/lib/`
 ↪ `AFastFacet.sol#67`)

`AFastFacet.onlyIssuerMember()` (`contracts/fast/lib/AFastFacet.sol#75-80`)
 ↪ ignores return value by `ICustomErrors.RequiresIssuerMembership(
 ↪ msg.sender)` (`contracts/fast/lib/AFastFacet.sol#77`)

`AFastFacet.onlyGovernor(address)` (`contracts/fast/lib/AFastFacet.sol`
 ↪ `#86-91`) ignores return value by `ICustomErrors.`
 ↪ `RequiresFastGovernorship(candidate)` (`contracts/fast/lib/`
 ↪ `AFastFacet.sol#88`)

`AFastFacet.onlyMember(address)` (`contracts/fast/lib/AFastFacet.sol`
 ↪ `#97-102`) ignores return value by `ICustomErrors.`
 ↪ `RequiresFastMembership(candidate)` (`contracts/fast/lib/AFastFacet.`
 ↪ `sol#99`)

`AFastFacet.differentAddresses(address,address)` (`contracts/fast/lib/`
 ↪ `AFastFacet.sol#109-114`) ignores return value by `ICustomErrors.`
 ↪ `RequiresDifferentSenderAndRecipient(a)` (`contracts/fast/lib/`
 ↪ `AFastFacet.sol#111`)

`IssuerAccessFacet.removeMember(address)` (`contracts/issuer/`
 ↪ `IssuerAccessFacet.sol#73-84`) ignores return value by

↪ `ICustomErrors.CannotSelfRemove(msg.sender)` (`contracts/issuer/IssuerAccessFacet.sol#78`)

`IssuerAccessFacet.governorAddedToFast(address)` (`contracts/issuer/IssuerAccessFacet.sol#89-99`) ignores return value by

↪ `ICustomErrors.RequiresFastContractCaller()` (`contracts/issuer/IssuerAccessFacet.sol#93`)

`IssuerAccessFacet.governorRemovedFromFast(address)` (`contracts/issuer/IssuerAccessFacet.sol#104-114`) ignores return value by

↪ `ICustomErrors.RequiresFastContractCaller()` (`contracts/issuer/IssuerAccessFacet.sol#108`)

`IssuerInitFacet.initialize(IssuerInitFacet.InitializerParams)` (`contracts/issuer/IssuerInitFacet.sol#33-63`) ignores return value by

↪ `ICustomErrors.AlreadyInitialized()` (`contracts/issuer/IssuerInitFacet.sol#39`)

`IssuerTopFacet.registerFast(address)` (`contracts/issuer/IssuerTopFacet.sol#44-60`) ignores return value by `ICustomErrors.DuplicateEntry()`

↪ (`contracts/issuer/IssuerTopFacet.sol#50`)

`AIssuerFacet.onlyDiamondFacet()` (`contracts/issuer/lib/AIssuerFacet.sol#23-28`) ignores return value by `ICustomErrors.InternalMethod()`

↪ (`contracts/issuer/lib/AIssuerFacet.sol#25`)

`AIssuerFacet.onlyDiamondOwner()` (`contracts/issuer/lib/AIssuerFacet.sol#31-36`) ignores return value by `ICustomErrors.RequiresDiamondOwnership()`

↪ (`contracts/issuer/lib/AIssuerFacet.sol#33`)

`AIssuerFacet.onlyMember(address)` (`contracts/issuer/lib/AIssuerFacet.sol#39-44`) ignores return value by `ICustomErrors.RequiresIssuerMembership(candidate)`

↪ (`contracts/issuer/lib/AIssuerFacet.sol#41`)

`MarketplaceAccessFacet.removeMember(address)` (`contracts/marketplace/MarketplaceAccessFacet.sol#75-87`) ignores return value by

↪ `ICustomErrors.RequiresNoFastMemberships(member)` (`contracts/marketplace/MarketplaceAccessFacet.sol#81`)

`MarketplaceAccessFacet.memberAddedToFast(address)` (`contracts/marketplace/MarketplaceAccessFacet.sol#104-112`) ignores return value by

↪ `ICustomErrors.RequiresFastContractCaller()` (`contracts/marketplace`
 ↪ `/MarketplaceAccessFacet.sol#108`)

`MarketplaceAccessFacet.memberRemovedFromFast(address)` (`contracts/`
 ↪ `marketplace/MarketplaceAccessFacet.sol#118-125`) ignores `return`
 ↪ `value` by `ICustomErrors.RequiresFastContractCaller()` (`contracts/`
 ↪ `marketplace/MarketplaceAccessFacet.sol#121`)

`MarketplaceAccessFacet.activateMember(address)` (`contracts/marketplace/`
 ↪ `MarketplaceAccessFacet.sol#139-154`) ignores `return value` by
 ↪ `ICustomErrors.RequiresMarketplaceDeactivatedMember(member)` (
 ↪ `contracts/marketplace/MarketplaceAccessFacet.sol#146`)

`MarketplaceAccessFacet.deactivateMember(address)` (`contracts/marketplace/`
 ↪ `MarketplaceAccessFacet.sol#160-175`) ignores `return value` by
 ↪ `ICustomErrors.RequiresMarketplaceActiveMember(member)` (`contracts/`
 ↪ `marketplace/MarketplaceAccessFacet.sol#167`)

`MarketplaceInitFacet.initialize(MarketplaceInitFacet.InitializerParams)`
 ↪ (`contracts/marketplace/MarketplaceInitFacet.sol#20-52`) ignores
 ↪ `return value` by `ICustomErrors.AlreadyInitialized()` (`contracts/`
 ↪ `marketplace/MarketplaceInitFacet.sol#25`)

`MarketplaceTokenHoldersFacet.fastBalanceChanged(address,uint256)` (
 ↪ `contracts/marketplace/MarketplaceTokenHoldersFacet.sol#16-39`)
 ↪ ignores `return value` by `ICustomErrors.RequiresFastContractCaller`
 ↪ `()` (`contracts/marketplace/MarketplaceTokenHoldersFacet.sol#25`)

`AMarketplaceFacet.onlyDeployer()` (`contracts/marketplace/lib/`
 ↪ `AMarketplaceFacet.sol#25-30`) ignores `return value` by
 ↪ `ICustomErrors.InternalMethod()` (`contracts/marketplace/lib/`
 ↪ `AMarketplaceFacet.sol#27`)

`AMarketplaceFacet.onlyIssuerMember()` (`contracts/marketplace/lib/`
 ↪ `AMarketplaceFacet.sol#35-40`) ignores `return value` by
 ↪ `ICustomErrors.RequiresIssuerMembership(msg.sender)` (`contracts/`
 ↪ `marketplace/lib/AMarketplaceFacet.sol#37`)

`AMarketplaceFacet.onlyMember(address)` (`contracts/marketplace/lib/`
 ↪ `AMarketplaceFacet.sol#46-51`) ignores `return value` by
 ↪ `ICustomErrors.RequiresMarketplaceMembership(candidate)` (`contracts`
 ↪ `/marketplace/lib/AMarketplaceFacet.sol#48`)

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation>

↪ #unused-return

FastTokenFacet.beforeRemovingMember(address) (contracts/fast/

↪ FastTokenFacet.sol#560-587) has external calls inside a loop:

↪ this.performDisapproval(member,spender,s.allowances[member][

↪ spender]) (contracts/fast/FastTokenFacet.sol#574)

FastTokenFacet.beforeRemovingMember(address) (contracts/fast/

↪ FastTokenFacet.sol#560-587) has external calls inside a loop:

↪ this.performDisapproval(owner,member,s.allowances[owner][member])

↪ (contracts/fast/FastTokenFacet.sol#584)

IssuerFrontendFacet.paginateDetailedFasts(uint256,uint256) (contracts/

↪ issuer/IssuerFrontendFacet.sol#22-33) has external calls inside a

↪ loop: fastDetails[i] = FastFrontendFacet(addresses[i]).details()

↪ (contracts/issuer/IssuerFrontendFacet.sol#29)

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation>

↪ /#calls-inside-a-loop

LibFast.data() (contracts/fast/lib/LibFast.sol#40-43) uses assembly

- INLINE ASM (contracts/fast/lib/LibFast.sol#42)

LibFastAccess.data() (contracts/fast/lib/LibFastAccess.sol#35-38) uses

↪ assembly

- INLINE ASM (contracts/fast/lib/LibFastAccess.sol#37)

LibFastHistory.data() (contracts/fast/lib/LibFastHistory.sol#72-75) uses

↪ assembly

- INLINE ASM (contracts/fast/lib/LibFastHistory.sol#74)

LibFastToken.data() (contracts/fast/lib/LibFastToken.sol#63-66) uses

↪ assembly

- INLINE ASM (contracts/fast/lib/LibFastToken.sol#65)

LibIssuer.data() (contracts/issuer/lib/LibIssuer.sol#24-27) uses

↪ assembly

- INLINE ASM (contracts/issuer/lib/LibIssuer.sol#26)

LibIssuerAccess.data() (contracts/issuer/lib/LibIssuerAccess.sol#22-25)

↪ uses assembly

- INLINE ASM (contracts/issuer/lib/LibIssuerAccess.sol#24)

LibDiamond.diamondStorage() ([contracts/lib/LibDiamond.sol#38-43](#)) uses
↳ assembly

- INLINE ASM ([contracts/lib/LibDiamond.sol#40-42](#))

LibDiamond.enforceHasContractCode(address,string) ([contracts/lib/](#)
↳ LibDiamond.sol#200-206) uses assembly

- INLINE ASM ([contracts/lib/LibDiamond.sol#202-204](#))

LibMarketplace.data() ([contracts/marketplace/lib/LibMarketplace.sol](#)
↳ #20-23) uses assembly

- INLINE ASM ([contracts/marketplace/lib/LibMarketplace.sol#22](#))

LibMarketplaceAccess.data() ([contracts/marketplace/lib/](#)
↳ LibMarketplaceAccess.sol#24-27) uses assembly

- INLINE ASM ([contracts/marketplace/lib/LibMarketplaceAccess.sol](#)
↳ #26)

LibMarketplaceTokenHolders.data() ([contracts/marketplace/lib/](#)
↳ LibMarketplaceTokenHolders.sol#20-23) uses assembly

- INLINE ASM ([contracts/marketplace/lib/](#)

↳ LibMarketplaceTokenHolders.sol#22)

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation>
↳ #assembly-usage

Pragma version0.8.10 ([contracts/fast/FastAccessFacet.sol#2](#)) necessitates
↳ a version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/fast/FastFrontendFacet.sol#2](#))
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/fast/FastHistoryFacet.sol#2](#))
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/fast/FastInitFacet.sol#2](#)) necessitates a
↳ version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/fast/FastTokenFacet.sol#2](#)) necessitates
↳ a version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/fast/FastTopFacet.sol#2`) necessitates a
↳ version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/fast/lib/AFastFacet.sol#2`) necessitates
↳ a version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/fast/lib/IFast.sol#2`) necessitates a
↳ version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/fast/lib/IFastEvents.sol#2`) necessitates
↳ a version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/fast/lib/LibFast.sol#2`) necessitates a
↳ version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/fast/lib/LibFastAccess.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/fast/lib/LibFastHistory.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/fast/lib/LibFastToken.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/interfaces/ICustomErrors.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/interfaces/IDiamondCut.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/interfaces/IDiamondLoupe.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/interfaces/IERC1404.sol#2`) necessitates
↳ a version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/interfaces/IERC165.sol#2`) necessitates a
↳ version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/interfaces/IERC173.sol#2`) necessitates a
↳ version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/interfaces/IERC20.sol#2`) necessitates a
↳ version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/interfaces/IHasActiveMembers.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/interfaces/IHasGovernors.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/interfaces/IHasMembers.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/issuer/IssuerAccessFacet.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/issuer/IssuerFrontendFacet.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/issuer/IssuerInitFacet.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/issuer/IssuerTopFacet.sol#2`)
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/issuer/lib/AIssuerFacet.sol#2](#))
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/issuer/lib/IIssuerEvents.sol#2](#))
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/issuer/lib/LibIssuer.sol#2](#)) necessitates
↳ a version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/issuer/lib/LibIssuerAccess.sol#2](#))
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/lib/LibAddressSet.sol#2](#)) necessitates a
↳ version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/lib/LibConstants.sol#2](#)) necessitates a
↳ version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/lib/LibDiamond.sol#2](#)) necessitates a
↳ version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/lib/LibHelpers.sol#2](#)) necessitates a
↳ version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/lib/LibPaginate.sol#2](#)) necessitates a
↳ version too recent to be trusted. Consider deploying with
↳ 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/marketplace/MarketplaceAccessFacet.sol](#)
↳ #2) necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 ([contracts/marketplace/MarketplaceInitFacet.sol#2](#))
↳ necessitates a version too recent to be trusted. Consider
↳ deploying with 0.6.12/0.7.6/0.8.7

Pragma version0.8.10 (`contracts/marketplace/MarketplaceTokenHoldersFacet.sol#2`) necessitates a version too recent to be trusted. Consider
↳ `deploying with 0.6.12/0.7.6/0.8.7`

Pragma version0.8.10 (`contracts/marketplace/MarketplaceTopFacet.sol#2`) necessitates a version too recent to be trusted. Consider
↳ `deploying with 0.6.12/0.7.6/0.8.7`

Pragma version0.8.10 (`contracts/marketplace/lib/AMarketplaceFacet.sol#2`) necessitates a version too recent to be trusted. Consider
↳ `deploying with 0.6.12/0.7.6/0.8.7`

Pragma version0.8.10 (`contracts/marketplace/lib/IMarketplaceEvents.sol#2`) necessitates a version too recent to be trusted. Consider
↳ `deploying with 0.6.12/0.7.6/0.8.7`

Pragma version0.8.10 (`contracts/marketplace/lib/LibMarketplace.sol#2`) necessitates a version too recent to be trusted. Consider
↳ `deploying with 0.6.12/0.7.6/0.8.7`

Pragma version0.8.10 (`contracts/marketplace/lib/LibMarketplaceAccess.sol#2`) necessitates a version too recent to be trusted. Consider
↳ `deploying with 0.6.12/0.7.6/0.8.7`

Pragma version0.8.10 (`contracts/marketplace/lib/LibMarketplaceTokenHolders.sol#2`) necessitates a version too recent to be trusted. Consider `deploying with 0.6.12/0.7.6/0.8.7`
solc-0.8.10 is not recommended for deployment

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation>
↳ `#incorrect-versions-of-solidity`

Low level call in `LibDiamond.initializeDiamondCut(address,bytes)` (`contracts/lib/LibDiamond.sol#180-198`):
- `(success,error) = _init.delegatecall(_calldata)` (`contracts/lib/LibDiamond.sol#188`)

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation>
↳ `#low-level-calls`

`FastTopFacet` (`contracts/fast/FastTopFacet.sol#11-68`) should inherit `IFast` (`contracts/fast/lib/IFast.sol#6-18`)

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation>
↳ `#missing-inheritance`

Parameter LibDiamond.setContractOwner(address)._newOwner (contracts/lib/
↳ LibDiamond.sol#47) is not in mixedCase

Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).
↳ _diamondCut (contracts/lib/LibDiamond.sol#66) is not in mixedCase

Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).
↳ _init (contracts/lib/LibDiamond.sol#67) is not in mixedCase

Parameter LibDiamond.diamondCut(IDiamondCut.FacetCut[],address,bytes).
↳ _calldata (contracts/lib/LibDiamond.sol#68) is not in mixedCase

Parameter LibDiamond.addFunctions(address,bytes4[])._facetAddress (
↳ contracts/lib/LibDiamond.sol#86) is not in mixedCase

Parameter LibDiamond.addFunctions(address,bytes4[])._functionSelectors (
↳ contracts/lib/LibDiamond.sol#86) is not in mixedCase

Parameter LibDiamond.replaceFunctions(address,bytes4[])._facetAddress (
↳ contracts/lib/LibDiamond.sol#104) is not in mixedCase

Parameter LibDiamond.replaceFunctions(address,bytes4[]).
↳ _functionSelectors (contracts/lib/LibDiamond.sol#104) is not in
↳ mixedCase

Parameter LibDiamond.removeFunctions(address,bytes4[])._facetAddress (
↳ contracts/lib/LibDiamond.sol#123) is not in mixedCase

Parameter LibDiamond.removeFunctions(address,bytes4[]).
↳ _functionSelectors (contracts/lib/LibDiamond.sol#123) is not in
↳ mixedCase

Parameter LibDiamond.addFacet(LibDiamond.DiamondStorage,address).
↳ _facetAddress (contracts/lib/LibDiamond.sol#135) is not in
↳ mixedCase

Parameter LibDiamond.addFunction(LibDiamond.DiamondStorage,bytes4,uint96
↳ ,address)._selector (contracts/lib/LibDiamond.sol#142) is not in
↳ mixedCase

Parameter LibDiamond.addFunction(LibDiamond.DiamondStorage,bytes4,uint96
↳ ,address)._selectorPosition (contracts/lib/LibDiamond.sol#142) is
↳ not in mixedCase

Parameter LibDiamond.addFunction(LibDiamond.DiamondStorage,bytes4,uint96
↳ ,address)._facetAddress (contracts/lib/LibDiamond.sol#142) is not
↳ in mixedCase

Parameter LibDiamond.removeFunction(LibDiamond.DiamondStorage,address,
↳ bytes4)._facetAddress (contracts/lib/LibDiamond.sol#148) is not
↳ in mixedCase

Parameter LibDiamond.removeFunction(LibDiamond.DiamondStorage,address,
↳ bytes4)._selector (contracts/lib/LibDiamond.sol#148) is not in
↳ mixedCase

Parameter LibDiamond.initializeDiamondCut(address,bytes)._init (
↳ contracts/lib/LibDiamond.sol#180) is not in mixedCase

Parameter LibDiamond.initializeDiamondCut(address,bytes)._calldata (
↳ contracts/lib/LibDiamond.sol#180) is not in mixedCase

Parameter LibDiamond.enforceHasContractCode(address,string)._contract (
↳ contracts/lib/LibDiamond.sol#200) is not in mixedCase

Parameter LibDiamond.enforceHasContractCode(address,string).
↳ _errorMessage (contracts/lib/LibDiamond.sol#200) is not in
↳ mixedCase

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation>
↳ #conformance-to-solidity-naming-conventions

LibFast.STORAGE_VERSION (contracts/fast/lib/LibFast.sol#13) is never
↳ used in LibFast (contracts/fast/lib/LibFast.sol#11-44)

LibFastAccess.STORAGE_VERSION (contracts/fast/lib/LibFastAccess.sol#14)
↳ is never used in LibFastAccess (contracts/fast/lib/LibFastAccess.
↳ sol#12-39)

LibFastHistory.STORAGE_VERSION (contracts/fast/lib/LibFastHistory.sol
↳ #14) is never used in LibFastHistory (contracts/fast/lib/
↳ LibFastHistory.sol#12-76)

LibFastToken.STORAGE_VERSION (contracts/fast/lib/LibFastToken.sol#15) is
↳ never used in LibFastToken (contracts/fast/lib/LibFastToken.sol
↳ #13-67)

LibFastToken.DEFAULT_TRANSFER_REFERENCE (contracts/fast/lib/LibFastToken
↳ .sol#21) is never used in LibFastToken (contracts/fast/lib/
↳ LibFastToken.sol#13-67)

LibIssuer.STORAGE_VERSION (contracts/issuer/lib/LibIssuer.sol#9) is
↳ never used in LibIssuer (contracts/issuer/lib/LibIssuer.sol#7-28)

LibIssuerAccess.STORAGE_VERSION ([contracts/issuer/lib/LibIssuerAccess.sol#9](#)) is never used in LibIssuerAccess ([contracts/issuer/lib/LibIssuerAccess.sol#7-26](#))

LibMarketplace.STORAGE_VERSION ([contracts/marketplace/lib/LibMarketplace.sol#9](#)) is never used in LibMarketplace ([contracts/marketplace/lib/LibMarketplace.sol#7-24](#))

LibMarketplaceAccess.STORAGE_VERSION ([contracts/marketplace/lib/LibMarketplaceAccess.sol#9](#)) is never used in LibMarketplaceAccess ([contracts/marketplace/lib/LibMarketplaceAccess.sol#7-28](#))

LibMarketplaceTokenHolders.STORAGE_VERSION ([contracts/marketplace/lib/LibMarketplaceTokenHolders.sol#9](#)) is never used in LibMarketplaceTokenHolders ([contracts/marketplace/lib/LibMarketplaceTokenHolders.sol#7-24](#))

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation>
↳ #unused-state-variable

details() should be declared external:

- FastFrontendFacet.details() ([contracts/fast/FastFrontendFacet.sol#107-126](#))

allowance(address,address) should be declared external:

- FastTokenFacet.allowance(address,address) ([contracts/fast/FastTokenFacet.sol#276-288](#))

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation>
↳ #public-function-that-could-be-declared-external

. analyzed (45 [contracts](#) with 72 detectors), 135 result(s) found

Conclusion:

Most of the vulnerabilities found by the analysis have already been addressed by the smart contract code review.

9 Conclusion

In this audit, we examined the design and implementation of Consilience contract and discovered several issues of varying severity. Consilience Group Limited team addressed 2 issues raised in the initial report and implemented the necessary fixes, while acknowledging the rest due to legal and business logic requirements.

10 Disclaimer

Shellboxes reports should not be construed as "endorsements" or "disapprovals" of particular teams or projects. These reports do not reflect the economics or value of any "product" or "asset" produced by any team or project that engages Shellboxes to do a security evaluation, nor should they be regarded as such. Shellboxes Reports do not provide any warranty or guarantee regarding the absolute bug-free nature of the examined technology, nor do they provide any indication of the technology's proprietors, business model, business or legal compliance. Shellboxes Reports should not be used in any way to decide whether to invest in or take part in a certain project. These reports don't offer any kind of investing advice and shouldn't be used that way. Shellboxes Reports are the result of a thorough auditing process designed to assist our clients in improving the quality of their code while lowering the significant risk posed by blockchain technology. According to Shellboxes, each business and person is in charge of their own due diligence and ongoing security. Shellboxes does not guarantee the security or functionality of the technology we agree to research; instead, our purpose is to assist in limiting the attack vectors and the high degree of variation associated with using new and evolving technologies.



For a Contract Audit, contact us at contact@shellboxes.com