CERTIK

Security Assessment

Tetu

Dec 19th, 2021

Table of Contents

Summary

This report has been prepared for Tetu - Audit 1 to discover issues and vulnerabilities in the source code of the Tetu project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

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

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

Overview

Project Summary

Project Name	Tetu
Platform	Polygon
Language	Solidity
Codebase	https://github.com/tetu-io/tetu-contracts/tree/certik-audit/
Commit	3b182085bff9991a7b3f9c3aaa730a5503d0df9a dfda2d064fa0a9b0e341d389c3ef1a6d6634cf18

Audit Summary

Delivery Date	Dec 19, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

Vulnerability Summary

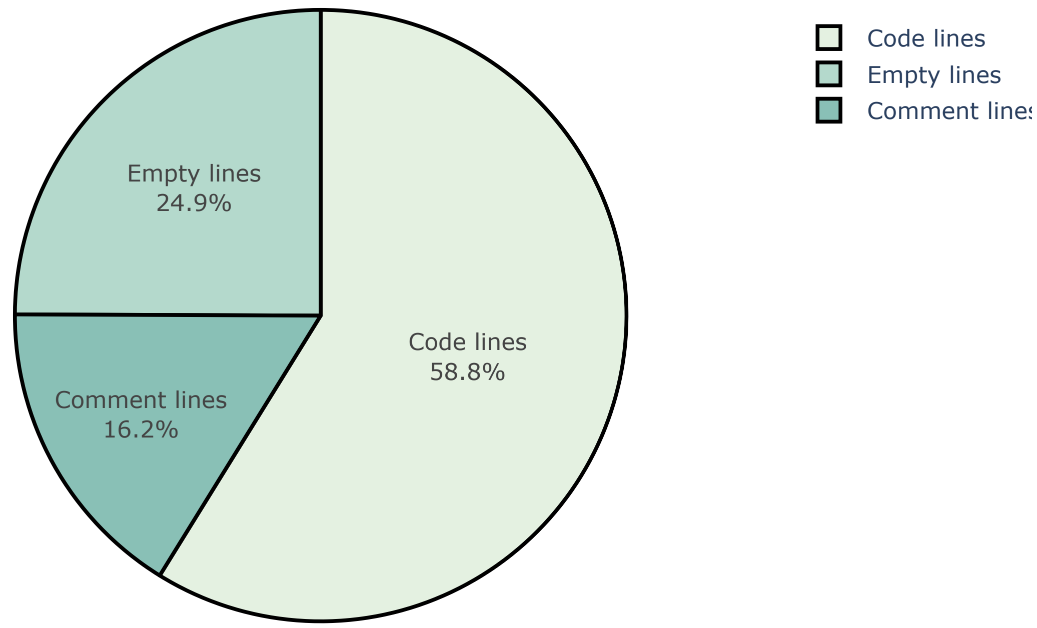
Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
● Critical	0	0	0	0	0	0
● Major	3	0	0	3	0	0
● Medium	1	0	0	0	0	1
● Minor	4	0	0	2	0	2
● Informational	5	0	0	0	0	5
● Discussion	0	0	0	0	0	0

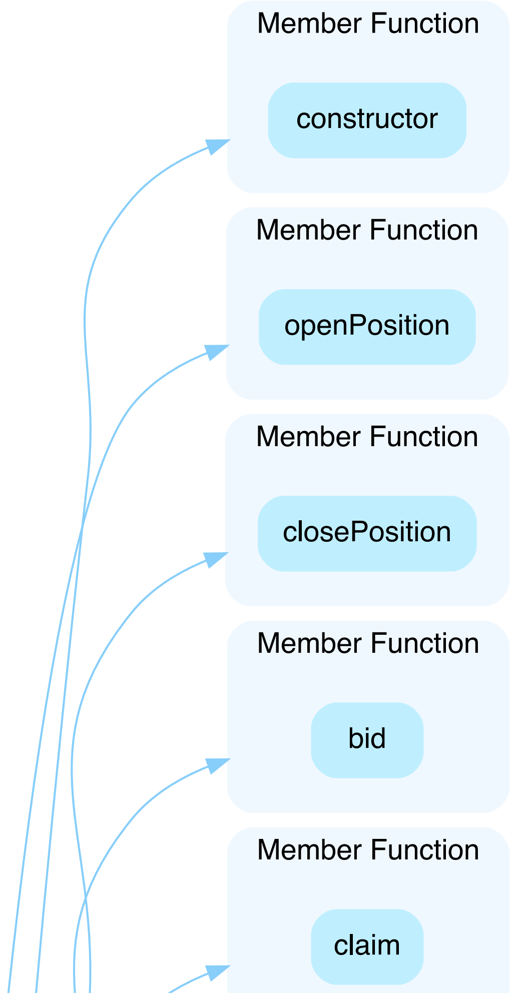
Audit Scope

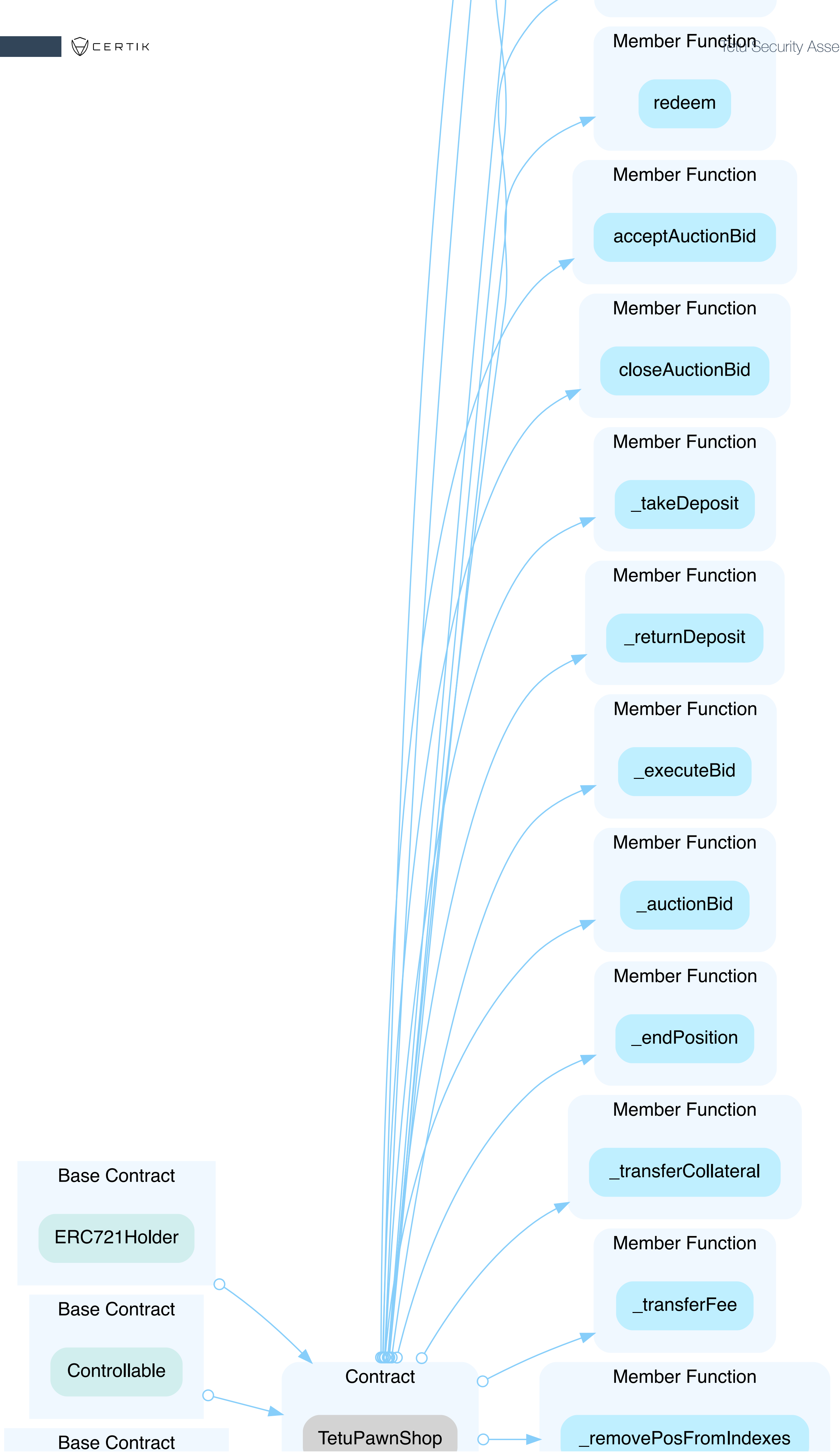
ID	File	SHA256 Checksum
ITP	loan/ITetuPawnShop.sol	c73c69ac7a610a661ed67b182619a74ce29ab2050eb22fa5065f797f0c874344
TPS	loan/TetuPawnShop.sol	dbcee01cfd6d2fce580a2187f386c968978a079a537f860e1cb8dc544244582b
ITS	swap/interfaces/ITetuSwapERC20.sol	1f084ec231fd4cadafb4b73f22c86066dfa5f714554e9ddb5a9d6ba404426527
ITF	swap/interfaces/ITetuSwapFactory.sol	bc2a0b6dd9938a9a6f6ceb6e4b57bd7d72f7db94e193735d7faf58665d79a8ec
ITT	swap/interfaces/ITetuSwapPair.sol	76898776e3c8af68ad5401bcbe41e997218280fd2afc63af19ea5f0f5b03eb63
ITR	swap/interfaces/ITetuSwapRouter.sol	a96ff883c5d609e5a4da6ec72eaea7c52eda3ab7383f26ea52293a4b9d0fd36e
IWE	swap/interfaces/IWETH.sol	032bfb4487d4301a78338f62746a653f3010b5f9f17798d34495edaacbb4cec3
MTC	swap/libraries/Math.sol	30a63107f23bd7756ab7d02fdd60c42a3fdbd6aca1034d69e77c423730845b19
TSL	swap/libraries/TetuSwapLibrary.sol	6cbb8b11805eb0ac7d375a2deb0e3f09c38e36950a0898cd8b6c729b4468b03a
THT	swap/libraries/TransferHelper.sol	d2e866a153de986e4c2171e6e621df2b6cb53d0244147e5c9283970aec85dd3b
UQT	swap/libraries/UQ112x112.sol	1815a971fd49b885df3abac1607d8b00413aa09da3963a09919b5e9c9e91cabf
FST	swap/FactoryStorage.sol	4d5451a3762d71f77f1e6fa76670c59eff342897125a1d9db3e2efc8625803c5
TSE	swap/TetuSwapERC20.sol	f74268087537ea1354011f5790f5f6ce0ae9fff00d5cc391a00c1db95f0c7500
TSF	swap/TetuSwapFactory.sol	e61d447dab84d0fd1864ac770b1d427cbc63f36c964ab2fc5cd57ffca559b88b
TSP	swap/TetuSwapPair.sol	25aa7e2fc11655a706a8513381108504d5dd01b590caf2f9add89ba40ee6ec8d
TSR	swap/TetuSwapRouter.sol	a96dcde2d6cb6c6af9ce6dbff4bacaac89332cbebe0e973794592b0ec9eb3d99

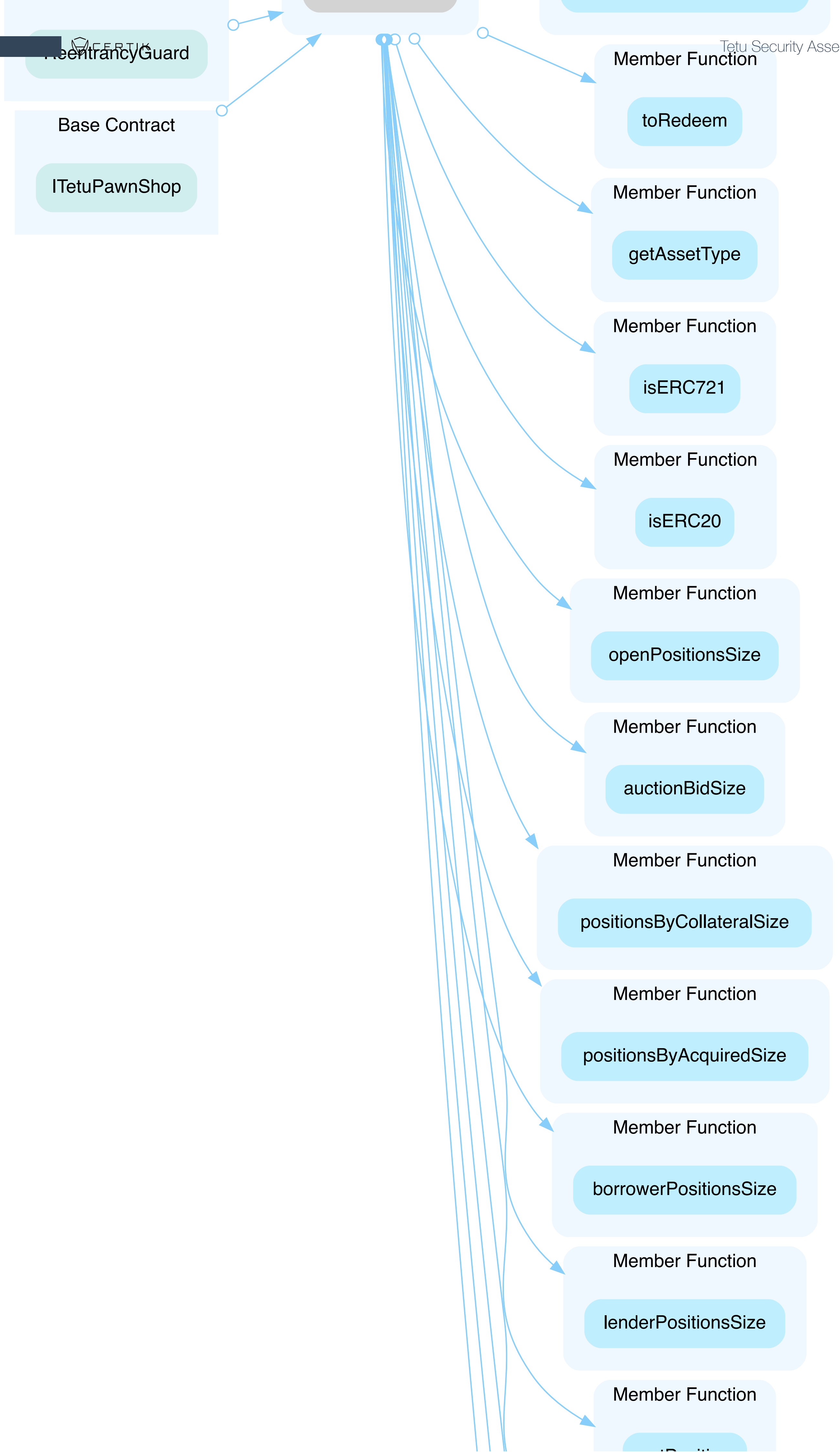
Diagrams

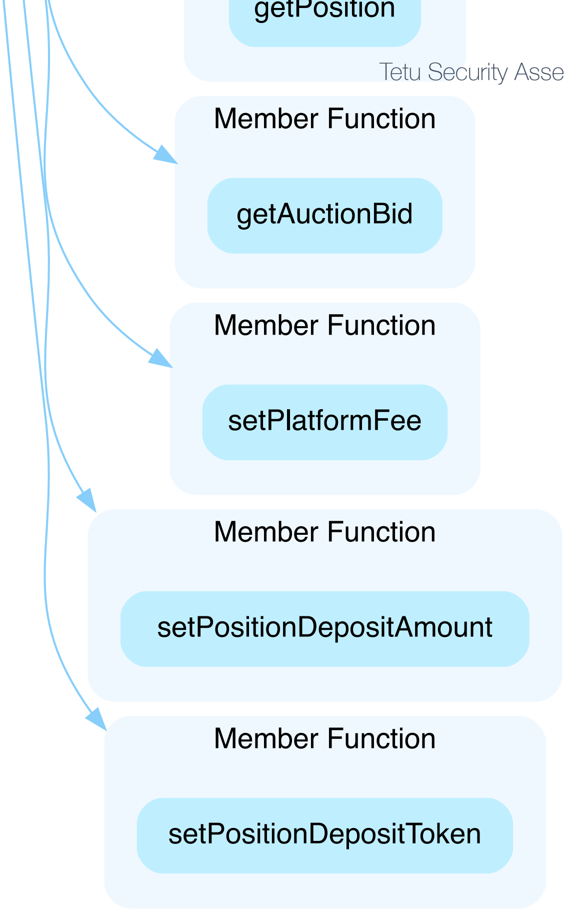
Source Line Chart

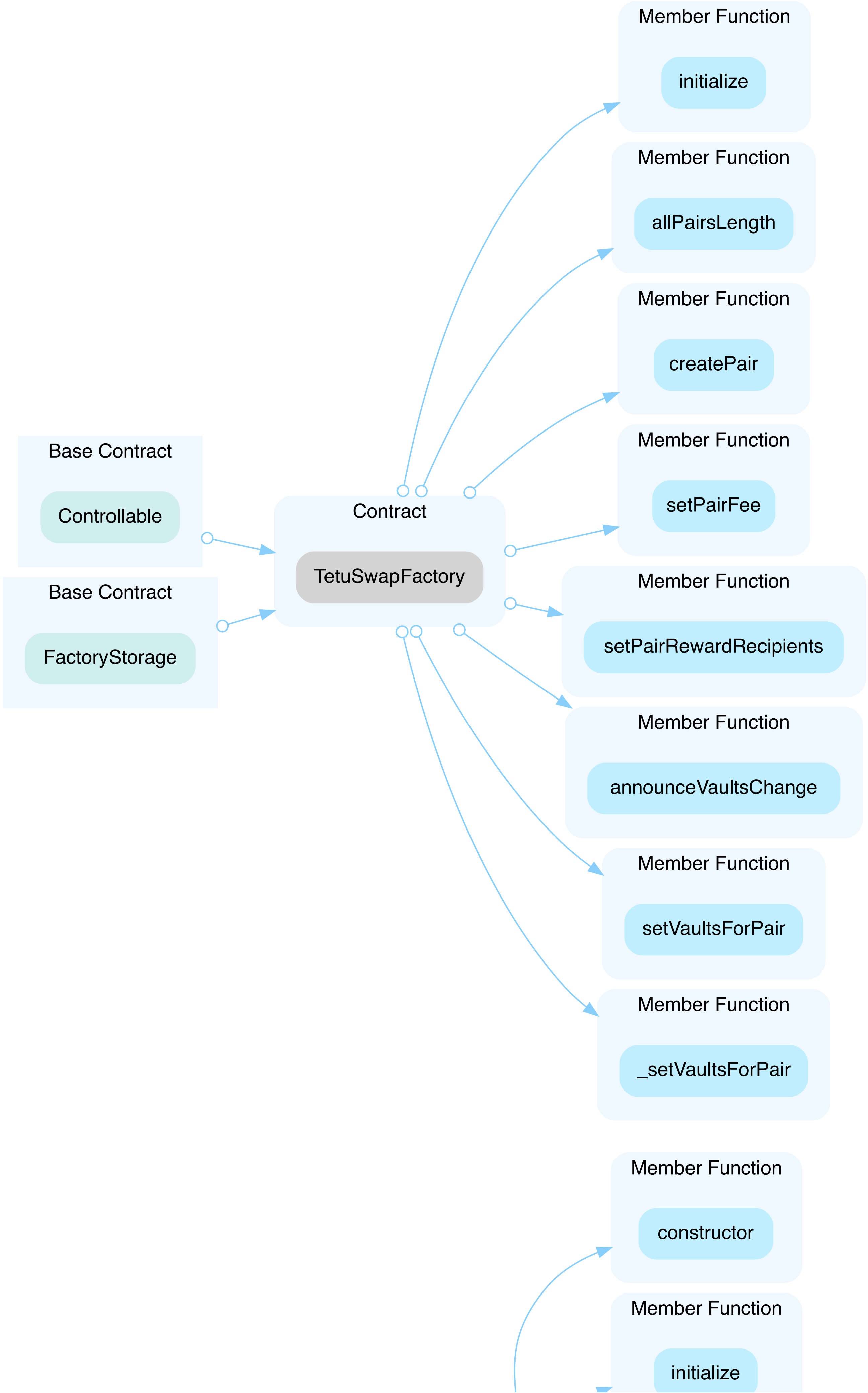


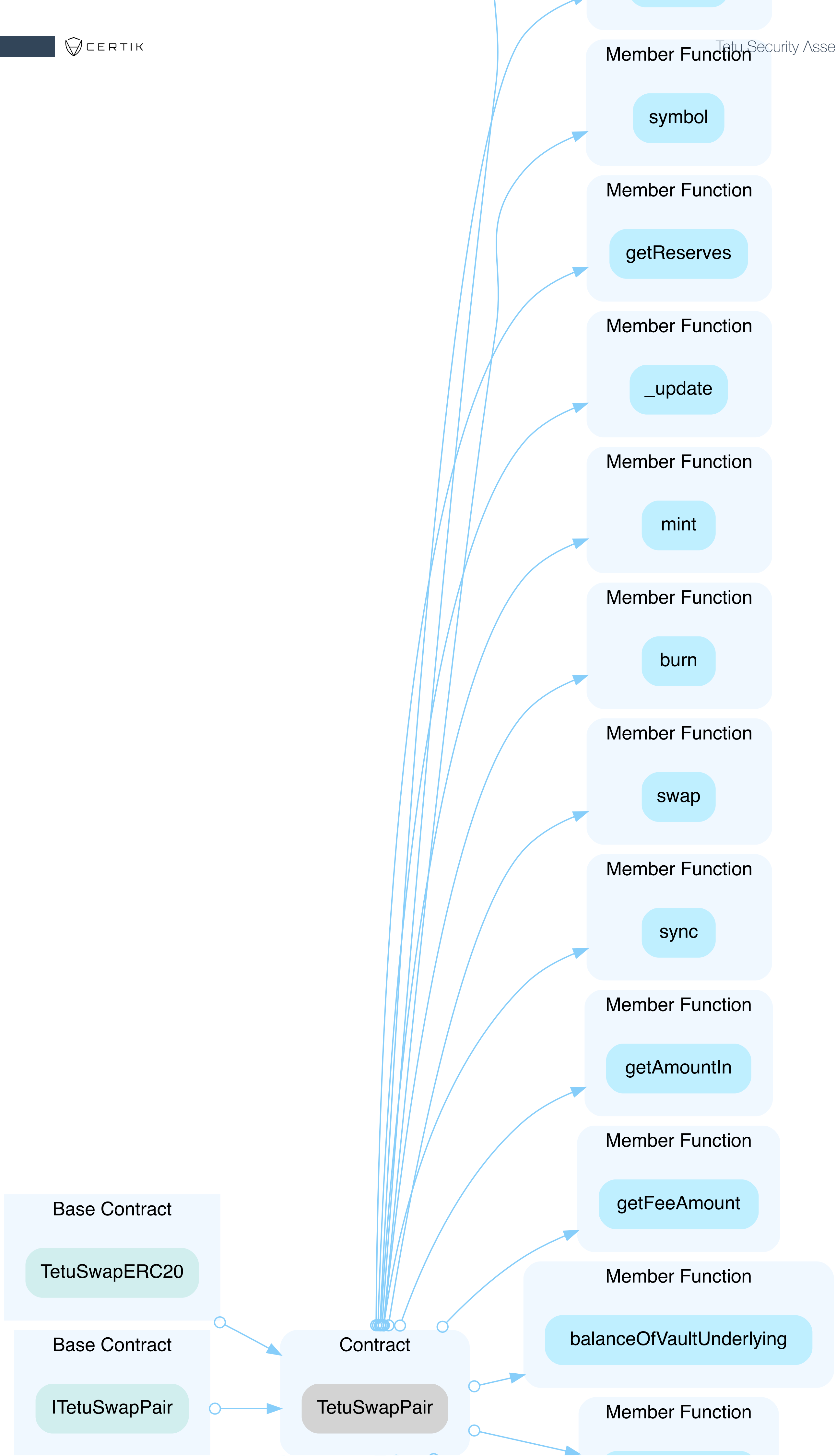


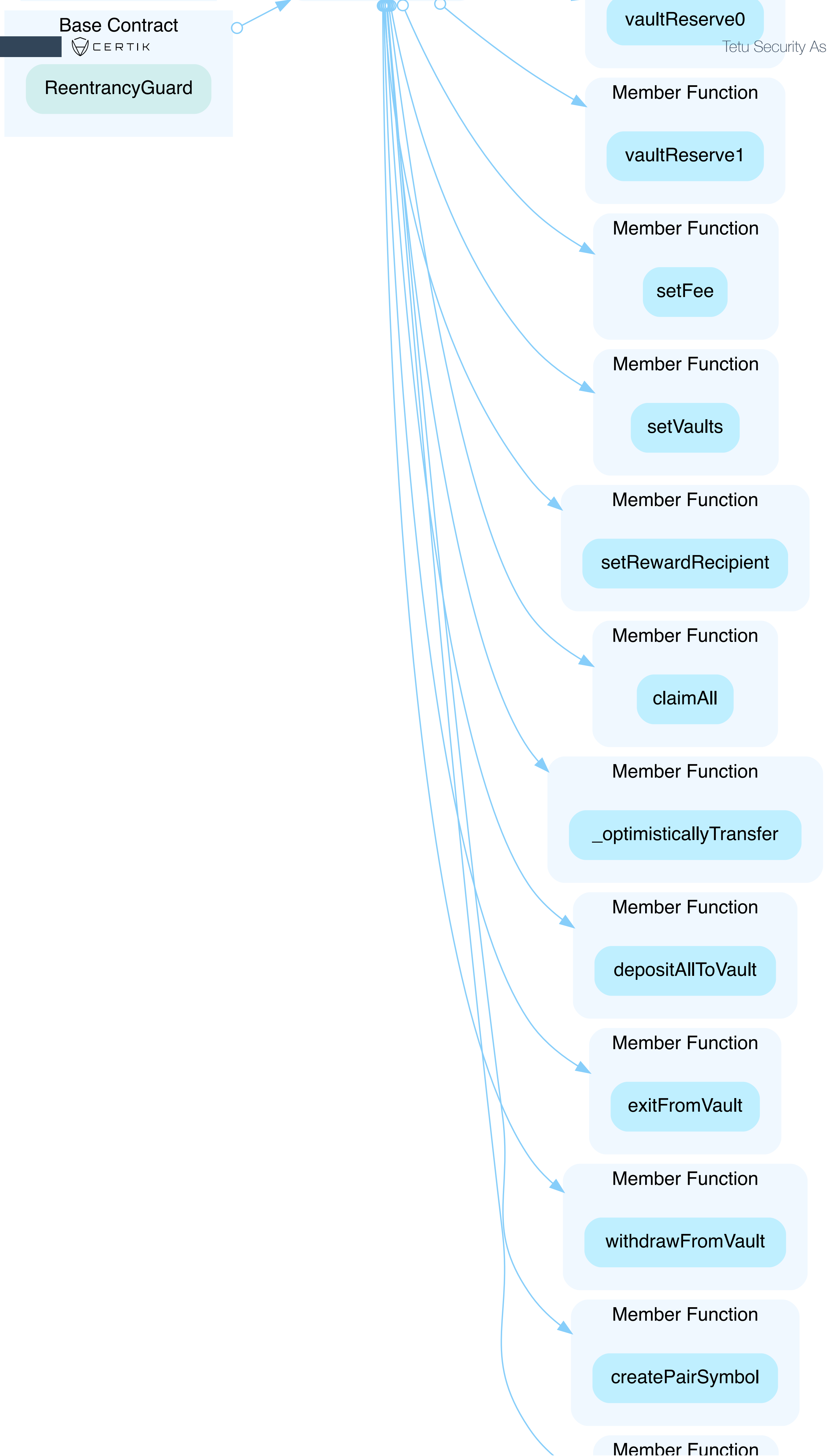

Member Function
redeem
Member Function
acceptAuctionBid
Member Function
closeAuctionBid
Member Function
_takeDeposit
Member Function
_returnDeposit
Member Function
_executeBid
Member Function
_auctionBid
Member Function
_endPosition
Member Function
_transferCollateral
Member Function
_transferFee
Member Function
_removePosFromIndexes
Base Contract
ERC721Holder
Base Contract
Controllable
Base Contract
Contract
TetuPawnShop

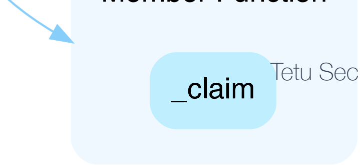

ReentrancyGuard
Base Contract
ITetuPawnShop
Member Function
toRedeem
Member Function
getAssetType
Member Function
isERC721
Member Function
isERC20
Member Function
openPositionsSize
Member Function
auctionBidSize
Member Function
positionsByCollateralSize
Member Function
positionsByAcquiredSize
Member Function
borrowerPositionsSize
Member Function
lenderPositionsSize
Member Function

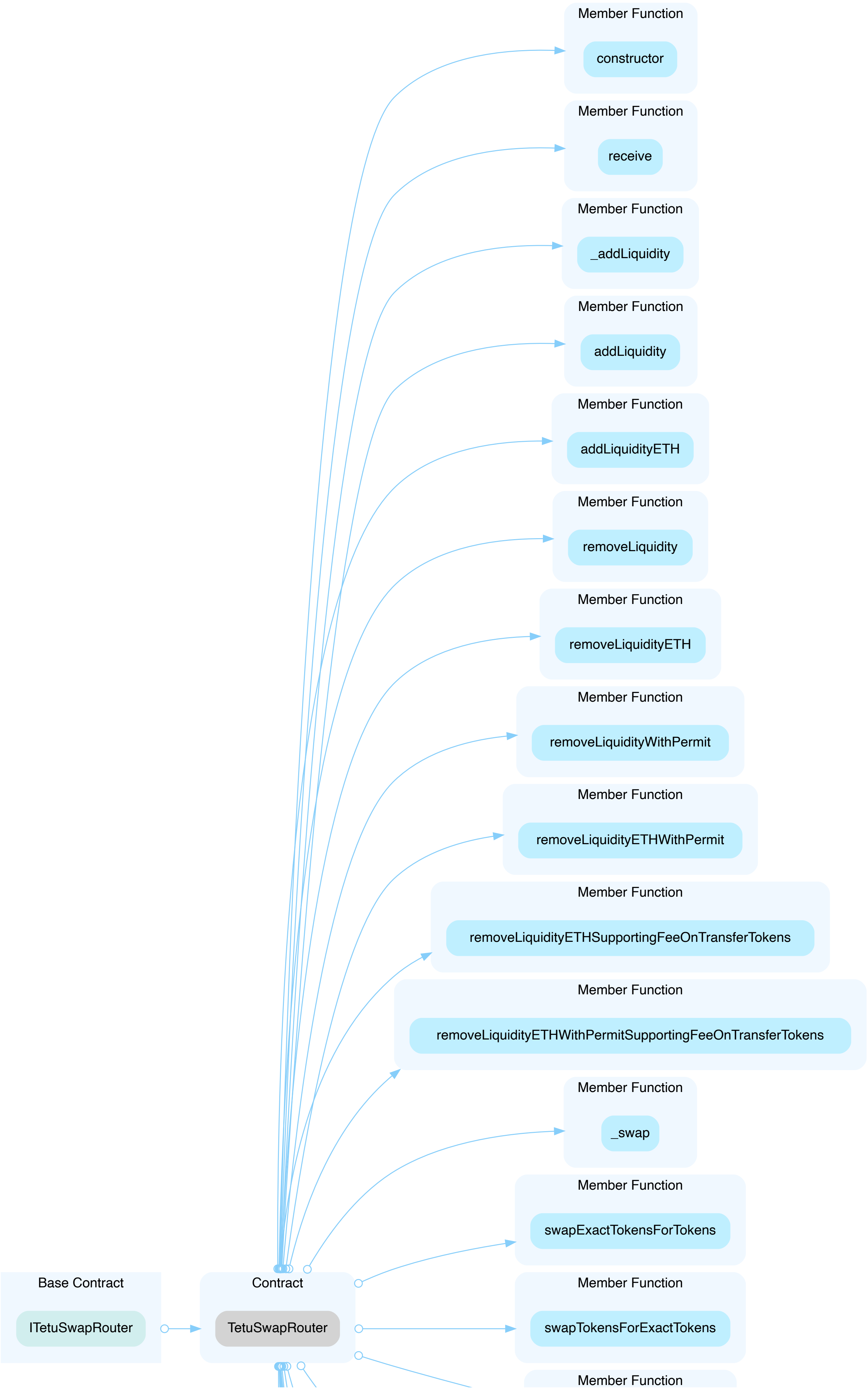

getPosition
Member Function
getAuctionBid
Member Function
setPlatformFee
Member Function
setPositionDepositAmount
Member Function
setPositionDepositToken

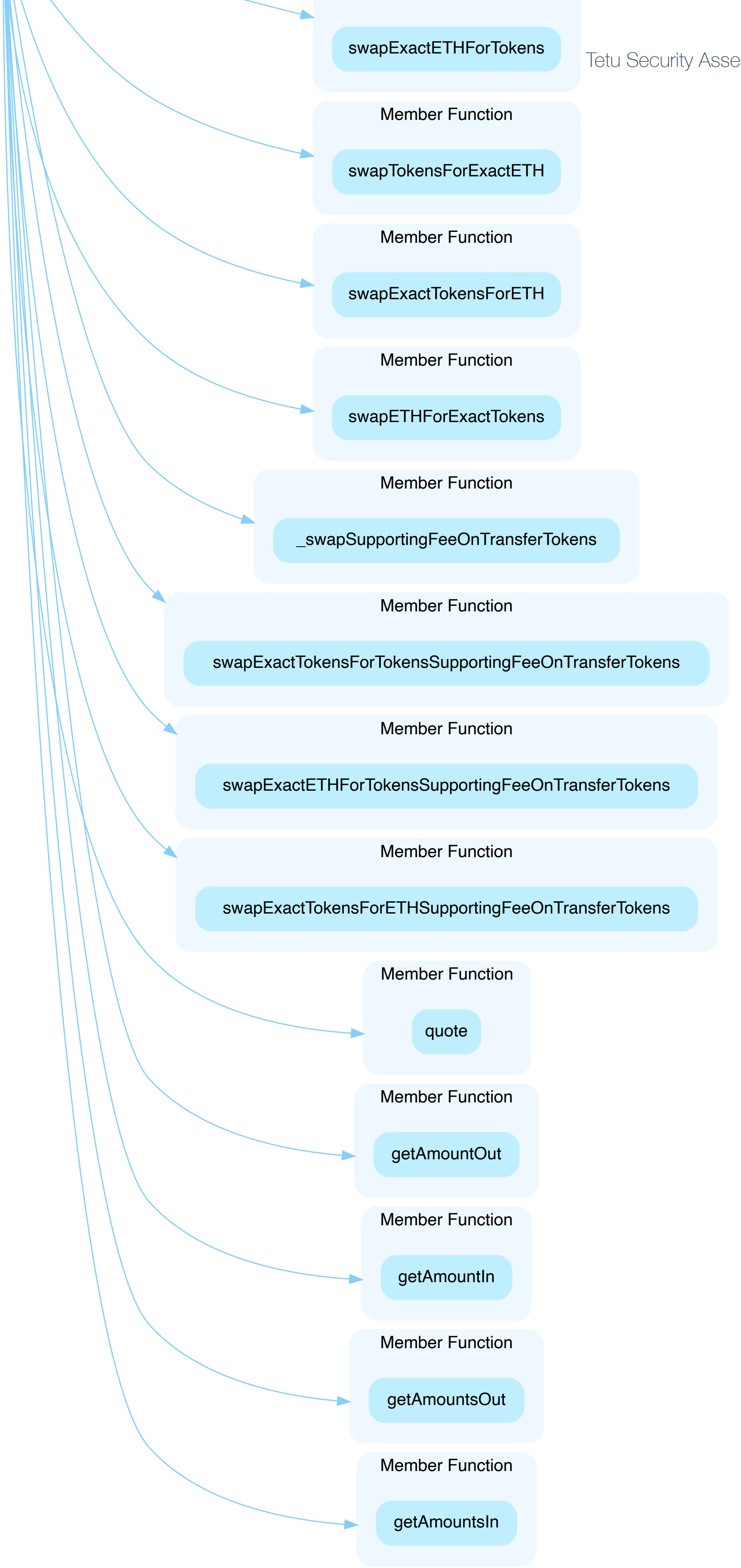

Member Function
initialize
Member Function
allPairsLength
Member Function
createPair
Member Function
setPairFee
Member Function
setPairRewardRecipients
Member Function
announceVaultsChange
Member Function
setVaultsForPair
Member Function
_setVaultsForPair
Base Contract
Controllable
Base Contract
FactoryStorage
Contract
TetuSwapFactory
Member Function
constructor
Member Function
initialize

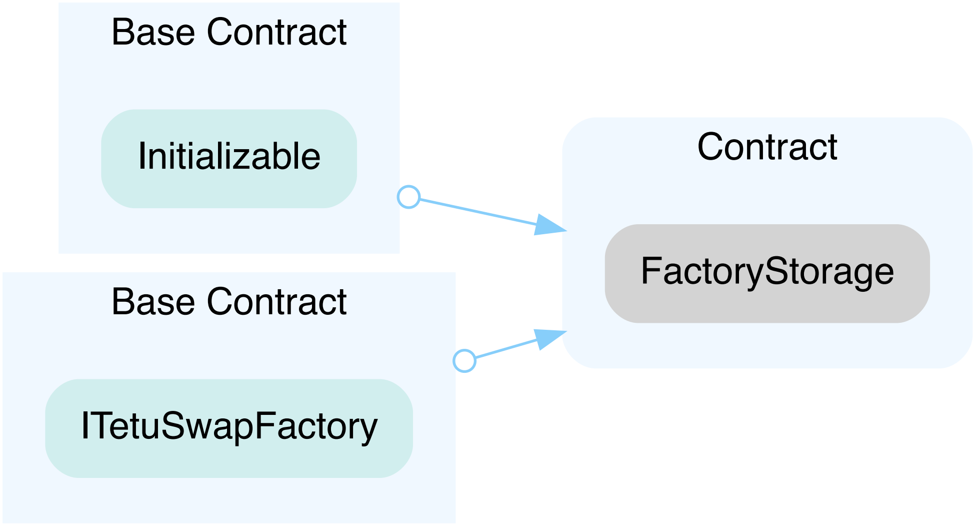

Member Function
symbol
Member Function
getReserves
Member Function
_update
Member Function
mint
Member Function
burn
Member Function
swap
Member Function
sync
Member Function
getAmountIn
Member Function
getFeeAmount
Member Function
balanceOfVaultUnderlying
Member Function
Base Contract
TetuSwapERC20
Base Contract
ITetuSwapPair
Contract
TetuSwapPair


Base Contract
ReentrancyGuard
vaultReserve0
Member Function
vaultReserve1
Member Function
setFee
Member Function
setVaults
Member Function
setRewardRecipient
Member Function
claimAll
Member Function
_optimisticallyTransfer
Member Function
depositAllToVault
Member Function
exitFromVault
Member Function
withdrawFromVault
Member Function
createPairSymbol
Member Function

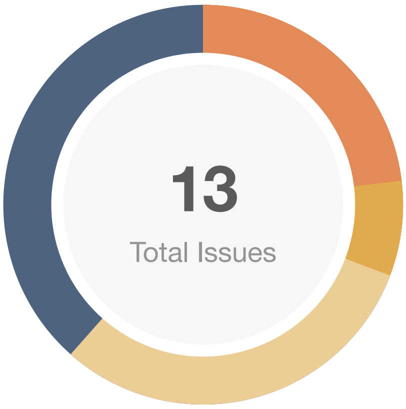

Member Function
_claim


Member Function
constructor
Member Function
receive
Member Function
_addLiquidity
Member Function
addLiquidity
Member Function
addLiquidityETH
Member Function
removeLiquidity
Member Function
removeLiquidityETH
Member Function
removeLiquidityWithPermit
Member Function
removeLiquidityETHWithPermit
Member Function
removeLiquidityETHSupportingFeeOnTransferTokens
Member Function
removeLiquidityETHWithPermitSupportingFeeOnTransferTokens
Member Function
_swap
Member Function
swapExactTokensForTokens
Base Contract
ITetuSwapRouter
Contract
TetuSwapRouter
Member Function
swapTokensForExactTokens
Member Function


swapExactETHForTokens
Member Function
swapTokensForExactETH
Member Function
swapExactTokensForETH
Member Function
swapETHForExactTokens
Member Function
_swapSupportingFeeOnTransferTokens
Member Function
swapExactTokensForTokensSupportingFeeOnTransferTokens
Member Function
swapExactETHForTokensSupportingFeeOnTransferTokens
Member Function
swapExactTokensForETHSupportingFeeOnTransferTokens
Member Function
quote
Member Function
getAmountOut
Member Function
getAmountIn
Member Function
getAmountsOut
Member Function
getAmountsIn


Base Contract
Initializable
Base Contract
ITetuSwapFactory
Contract
FactoryStorage

Findings



🟥 **Critical**	**0** (0.00%)
🟧 **Major**	**3** (23.08%)
🟨 **Medium**	**1** (7.69%)
🟨 **Minor**	**4** (30.77%)
🟦 **Informational**	**5** (38.46%)
🟩 **Discussion**	**0** (0.00%)

ID	Title	Category	Severity	Status
TPS-01	Centralization Risk in TetuPawnShop.sol	**Centralization / Privilege**	● **Major**	ⓘ Acknowledged
TPS-02	Hardcoded Gas Stipend	Logical Issue	● Medium	✓ Resolved
TPS-03	Missing Emit Events	Coding Style	● Informational	✓ Resolved
TPS-04	Undocumented Functionality	Logical Issue	● Informational	✓ Resolved
TPS-05	Third Party Dependencies	Volatile Code	● Minor	ⓘ Acknowledged
TSF-01	Centralization Risk in TetuSwapFactory.sol	**Centralization / Privilege**	● **Major**	ⓘ Acknowledged
TSP-01	Centralization Risk in TetuSwapPair.sol	**Centralization / Privilege**	● **Major**	ⓘ Acknowledged
TSP-02	Missing Emit Events	Coding Style	● Informational	✓ Resolved
TSP-03	Divide by Zero	Logical Issue	● Minor	✓ Resolved
TSR-01	Missing Emit Events	Coding Style	● Informational	✓ Resolved
TSR-02	Incompatibility With Deflationary Tokens	Logical Issue	● Minor	ⓘ Acknowledged
TSR-03	Missing Input Validation	Volatile Code	● Minor	✓ Resolved
TSR-04	Proper Usage of `require` And `assert` Functions	Coding Style	● Informational	✓ Resolved

TPS-01 | Centralization Risk in TetuPawnShop.sol

Category	Severity	Location	Status
Centralization / Privilege	● Major	projects/Tetu/loan/TetuPawnShop.sol (b662827): 499, 505, 510	ⓘ Acknowledged

Description

In the contract `TetuPawnShop`, the role validated in `onlyControllerOrGovernance` modifier has the authority over the following function:

- `setPlatformFee()`
- `setPositionDepositAmount()`
- `setPositionDepositToken()`

Any compromise to the privileged account may allow the hacker to take advantage of this and update the sensitive settings of the system.

Recommendation

We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

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

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

Alleviation

`[CertiK]`: In the commit dfda2d064fa0a9b0e341d389c3ef1a6d6634cf18, additional privileged functions added

- `announceGovernanceAction()` authorized by privileged role `owner`
- `setOwner()` authorized by privileged role `owner`

- `setFeeRecipient()` authorized by privileged role `owner`

TPS-02 | Hardcoded Gas Stipend

Category	Severity	Location	Status
Logical Issue	● Medium	projects/Tetu/loan/TetuPawnShop.sol (b662827): 404	✓ Resolved

Description

The outward `forwarder.liquidate` performed in `_transferFee()` may cease to function in a future version of Ethereum given that it has a hardcoded gas allowance.

Recommendation

We advise a upgradable gas limitation to be set for the call to avoid new EIPs such as EIP-2929 from rendering the code inexecutable.

Alleviation

`[CertiK]`: the client heeded the advice and removed the gas limitation in the commit dfda2d064fa0a9b0e341d389c3ef1a6d6634cf18

TPS-03 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	● Informational	projects/Tetu/loan/TetuPawnShop.sol (b662827): 499, 505, 510	✓ Resolved

Description

The function that affects the status of sensitive variables should be able to emit events as notifications.

- `setPlatformFee()`
- `setPositionDepositAmount()`
- `setPositionDepositToken()`

Recommendation

We advise the client to consider adding events for sensitive actions, and emit them in the function.

Alleviation

`[CertiK]`: the client heeded the advice and added the events in the commit dfda2d064fa0a9b0e341d389c3ef1a6d6634cf18

TPS-04 | Undocumented Functionality

Category	Severity	Location	Status
Logical Issue	● Informational	projects/Tetu/loan/TetuPawnShop.sol (b662827): 204, 219	✓ Resolved

Description

In the functions `claim()` and `redeem()`, lender and borrower can claim and redeem on the specific position respectively. As the `pos.open` is the critical status of the specific position, there's no documentation to elaborate if the `claim` and `redeem` can be done when `pos.open` is false.

Recommendation

We would like to confirm with the client if the `claim` and `redeem` can be done when `pos.open` is false. If no, then the following check should be considered to be added in both functions

```
require(pos.open, "position closed");
```

Alleviation

`[CertiK]`: the client heeded the advice and added the validations in the commit dfda2d064fa0a9b0e341d389c3ef1a6d6634cf18

TPS-05 | Third Party Dependencies

Category	Severity	Location	Status
Volatile Code	● Minor	projects/Tetu/loan/TetuPawnShop.sol (b662827): 20~23	ⓘ Acknowledged

Description

The contract is serving as the underlying entity to interact with third-party protocols and dependencies that are out of the scope. The scope of the audit treats these entities as black boxes and assumes their functional correctness. However, in the real world, third parties can be compromised and unknown out-of-scope dependencies may lead to lost or stolen assets. In addition, upgrades of third parties can possibly create severe impacts, such as increasing fees of 3rd parties, migrating to new LP pools, etc.

Recommendation

We understand that the business logic of TetuPawnShop requires interaction with openzeppelin, Controllable/ArrayLib dependencies, etc. We encourage the team to constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

TSF-01 | Centralization Risk in TetuSwapFactory.sol

Category	Severity	Location	Status
Centralization / Privilege	● Major	projects/Tetu/swap/TetuSwapFactory.sol (b662827): 73, 78, 86, 97	ⓘ Acknowledged

Description

In the contract `TetuSwapFactory`, the role validated in `onlyControllerOrGovernance` modifier has the authority over the following function:

- `setPairFee()`
- `setPairRewardRecipients()`
- `announceVaultsChange()`
- `setVaultsForPair()`

Any compromise to the privileged account may allow the hacker to take advantage of this and update the sensitive settings and execute the sensitive functions of the system.

Recommendation

We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

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

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

Alleviation

`[CertiK]`: In the commit dfda2d064fa0a9b0e341d389c3ef1a6d6634cf18, additional privileged functions added

- `announcePairsFeeChange()` authorized by privileged role `onlyControllerOrGovernance`

- `setPairsFee()` authorized by privileged role `onlyControllerOrGovernance`
- `setFeeRecipient()` authorized by privileged role `onlyControllerOrGovernance`

TSP-01 | Centralization Risk in TetuSwapPair.sol

Category	Severity	Location	Status
Centralization / Privilege	● Major	projects/Tetu/swap/TetuSwapPair.sol (b662827): 92~106, 305~309 , 312~328, 331~334, 338~343	ⓘ Acknowledged

Description

In the contract, `TetuSwapPair`, the role, `factory`, has the authority over the below functions:

- `initialize()`
- `setFee()`
- `setVaults()`
- `setRewardRecipient()`

Moreover, the role, `rewardRecipient`, has the authority over the below function

- `claimAll()`

Any compromise to the privileged account may allow the hacker to take advantage of this and update the sensitive settings and execute the sensitive functions of the system.

Recommendation

We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked.

In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

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

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

TSP-02 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	● Informational	projects/Tetu/swap/TetuSwapPair.sol (b662827): 92~106	✓ Resolved

Description

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

Recommendation

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

Alleviation

`[CertiK]`: the client heeded the advice and added the events in the commit dfda2d064fa0a9b0e341d389c3ef1a6d6634cf18

TSP-03 | Divide by Zero

Category	Severity	Location	Status
Logical Issue	● Minor	projects/Tetu/swap/TetuSwapPair.sol (b662827): 176~179	✓ Resolved

Description

If the value of `totalSupply` is 0, the following two division operations will fail due to the divide by 0 error, which ultimately makes the invocation to `burn()` function fail.

```
178  uint shareToWithdraw0 = liquidity * shareAmount0 / totalSupply;
179  uint shareToWithdraw1 = liquidity * shareAmount1 / totalSupply;
```

Recommendation

We advise the client to add the following validation in the function `burn()`

```
function burn(address to) external nonReentrant override returns (uint amount0, uint amount1) {
   require(totalSupply != 0, "The value of totalSupply must not be 0");
    ...
}
```

Alleviation

`[CertiK]`: the client heeded the advice and fixed the issue in the commit dfda2d064fa0a9b0e341d389c3ef1a6d6634cf18

TSR-01 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	● Informational	projects/Tetu/swap/TetuSwapRouter.sol (b662827): 44~47	✓ Resolved

Description

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

Recommendation

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

Alleviation

`[CertiK]`: the client heeded the advice and added the events in the commit dfda2d064fa0a9b0e341d389c3ef1a6d6634cf18

TSR-02 | Incompatibility With Deflationary Tokens

Category	Severity	Location	Status
Logical Issue	● Minor	projects/Tetu/swap/TetuSwapRouter.sol (b662827): 88, 88~89, 110, 129	ⓘ Acknowledged

Description

When users add or remove LP tokens into the router, and the `mint` and `burn` operations are performed. When transferring standard ERC20 deflationary tokens, the input amount may not be equal to the received amount due to the charged transaction fee. As a result, the amount inconsistency will occur and the transaction may fail due to the validation checks.

Recommendation

We advise the client to regulate the set of LP tokens supported and add necessary mitigation mechanisms to keep track of accurate balances if there is a need to support deflationary tokens.

TSR-03 | Missing Input Validation

Category	Severity	Location	Status
Volatile Code	● Minor	projects/Tetu/swap/TetuSwapRouter.sol (b662827): 39	✓ Resolved

Description

The given input is missing the check for the non-zero address.

Recommendation

We advise adding the check for the passed-in values to prevent unexpected error as below:

```
39 constructor(address _factory, address _WETH) {
40      require(_factory != address(0),"_factory should not be address(0)");
41      require(_WETH != address(0),"_WETH should not be address(0)");
42      factory = _factory;
43      WETH = _WETH;
44 }
```

Alleviation

`[CertiK]`: the client heeded the advice and added the validations in the commit dfda2d064fa0a9b0e341d389c3ef1a6d6634cf18

TSR-04 | Proper Usage of `require` And `assert` Functions

Category	Severity	Location	Status
Coding Style	● Informational	projects/Tetu/swap/TetuSwapRouter.sol (b662827): 45	✓ Resolved

Description

The `assert()` function should only be used to test for internal errors, and to check invariants. The `require()` function should be used to ensure valid conditions, such as inputs, or contract state variables are met, or to validate return values from calls to external contracts.

Recommendation

We advise the client using the `require()` function, along with a custom error message when the condition fails, instead of the `assert()` function.

Alleviation

`[CertiK]`: the client heeded the advice and fixed the issue in the commit dfda2d064fa0a9b0e341d389c3ef1a6d6634cf18

Appendix

Finding Categories

Centralization / Privilege

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

Logical Issue

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

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

Coding Style

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

Checksum Calculation Method

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

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

Disclaimer

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

This report is not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK’s position is that each company and individual are responsible for their own due diligence and continuous security. CertiK’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

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

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

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

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

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

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

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

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

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

About

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