CERTIK

Security Assessment

AstridDao

Apr 22nd, 2022

Table of Contents

Summary

This report has been prepared for AstridDao to discover issues and vulnerabilities in the source code of the AstridDao 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	AstridDao
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/AstridDao/contracts
Commit	9e0d1955eb2ad80924956543c90e811ae5fa3cfd

Audit Summary

Delivery Date	Apr 22, 2022 UTC
Audit Methodology	Static Analysis, Manual Review

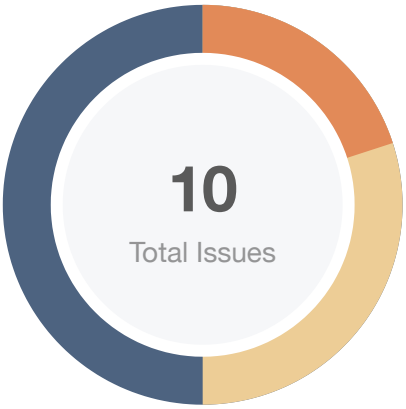
Vulnerability Summary

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

Audit Scope

ID	File	SHA256 Checksum
BAI	BAIToken.sol	31246191981aa6b67cf6ce066d386dcfdfa4cbd30e26dc016bc227b40b1439ab
ATD	ATID/ATIDStaking.sol	62a429af11734f033ecb0bb21d254e05bc460e70bca3549d763ac38b33f65e4c
GTA	ATID/GovToken.sol	a506a63e011144d9a65e70ec70b21b83e6c65127d9a9362e04ccab904043ba88
ATT	ATID/ATIDToken.sol	8671693546e56e47bd040bc70b013b4aeb9e01e7afa83f0e6cbbde00ecc0e7e8
CIA	ATID/CommunityIssuance.sol	0160ab3711009b3988a2d7464f7717e9722acb32102539a038c4d57d278037d5
LCA	ATID/LockupContract.sol	9953493081b55ebe3c7863157c36370d67a5b19a0d9172a4b9099cc9c47fe74a
ATI	ATID	
LCF	ATID/LockupContractFactory.sol	27d2b95333acd9167f0e5a5b3ebd6fb4f451e65cb6b8d5d6489dac4611d57e42

Findings



Critical	**0** (0.00%)
Major	**2** (20.00%)
Medium	**0** (0.00%)
Minor	**3** (30.00%)
Informational	**5** (50.00%)
Discussion	**0** (0.00%)

ID	Title	Category	Severity	Status
ATID-01	Centralization Related Risks	**Centralization / Privilege**	**Major**	Acknowledged
ADC-01	Missing Emit Events	Coding Style	Informational	Resolved
ADC-02	Unlocked Compiler Version	Language Specific	Informational	Resolved
ADC-03	Missing Error Messages	Coding Style	Informational	Acknowledged
ATD-01	Third Party Dependencies	Logical Issue	Minor	Acknowledged
ATD-02	Discussion On The `ATIDStaking` Contract	Logical Issue	Minor	Acknowledged
ATD-03	Redundant Code Components	Volatile Code	Informational	Acknowledged
ATD-04	Discussion On `unstakeLocked()`	Logical Issue	Informational	Resolved
ATI-01	Unchecked ERC-20 `transfer()` / `transferFrom()` Call	Volatile Code	Minor	Resolved
CIA-01	Tautology Or Contradiction	Logical Issue	Major	Resolved

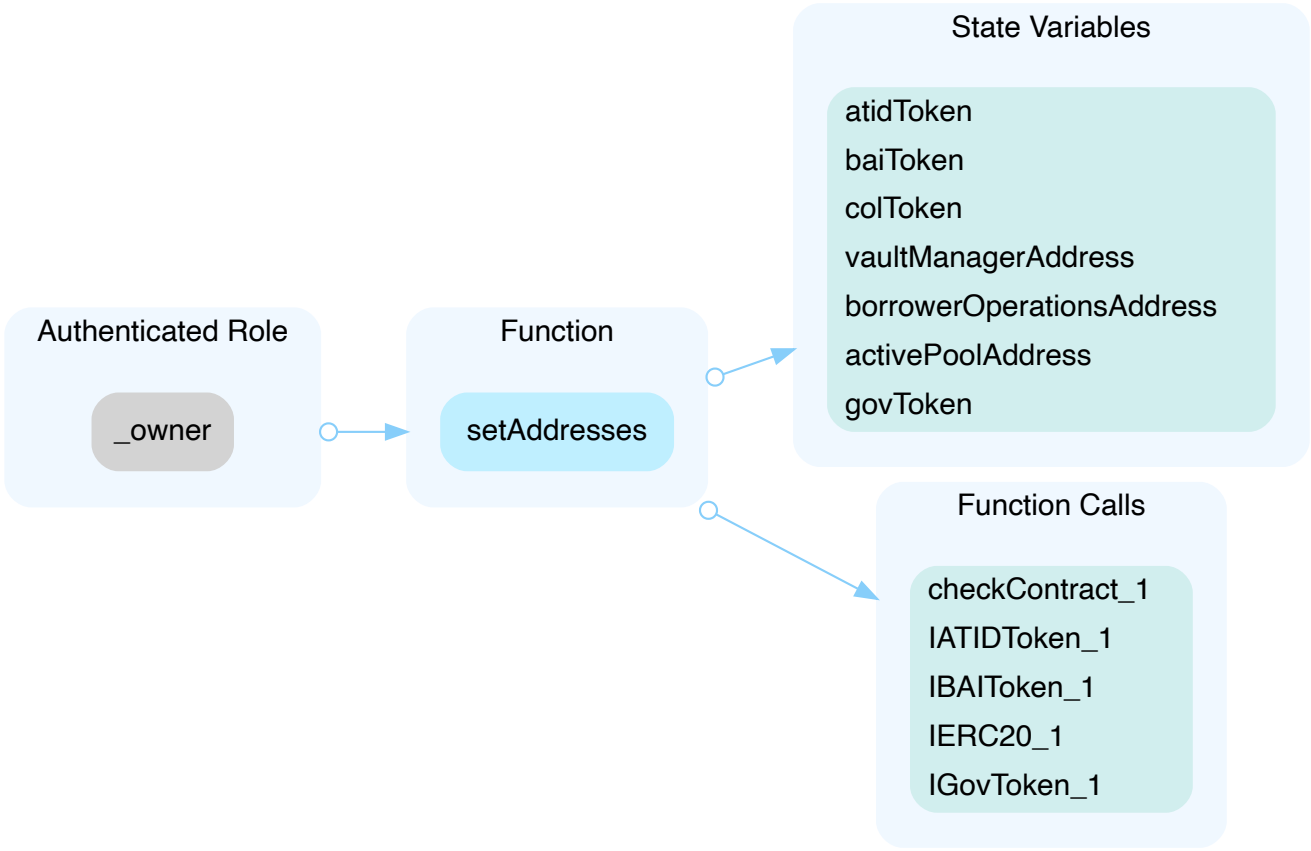
ATID-01 | Centralization Related Risks

Category	Severity	Location	Status
Centralization / Privilege	Major	Global	Acknowledged

Description

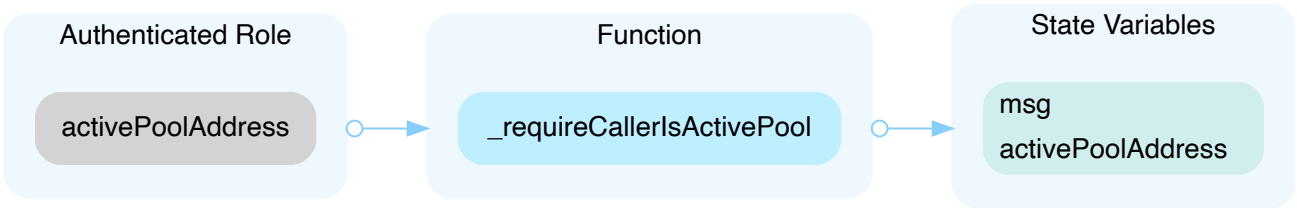
In the contract `ATIDStaking` the role `_owner` has authority over the functions shown in the diagram below.

Any compromise to the `_owner` account may allow the hacker to take advantage of this authority.



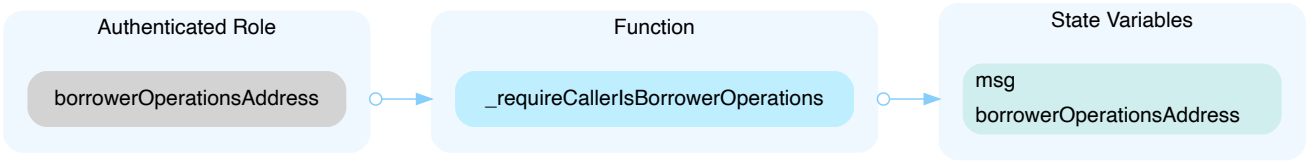
In the contract `ATIDStaking` the role `activePoolAddress` has authority over the functions shown in the diagram below.

Any compromise to the `activePoolAddress` account may allow the hacker to take advantage of this authority.



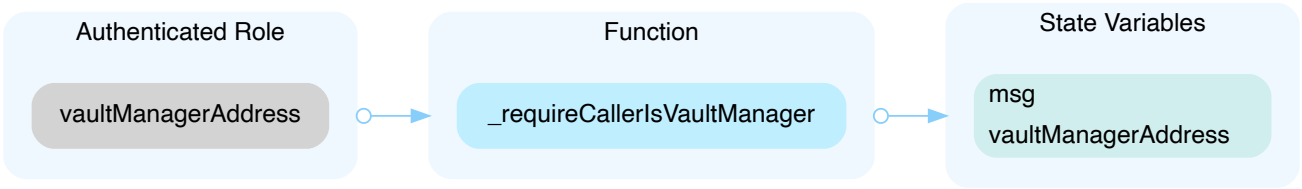
In the contract `ATIDStaking` the role `borrowerOperationsAddress` has authority over the functions shown in the diagram below.

Any compromise to the `borrowerOperationsAddress` account may allow the hacker to take advantage of this authority.



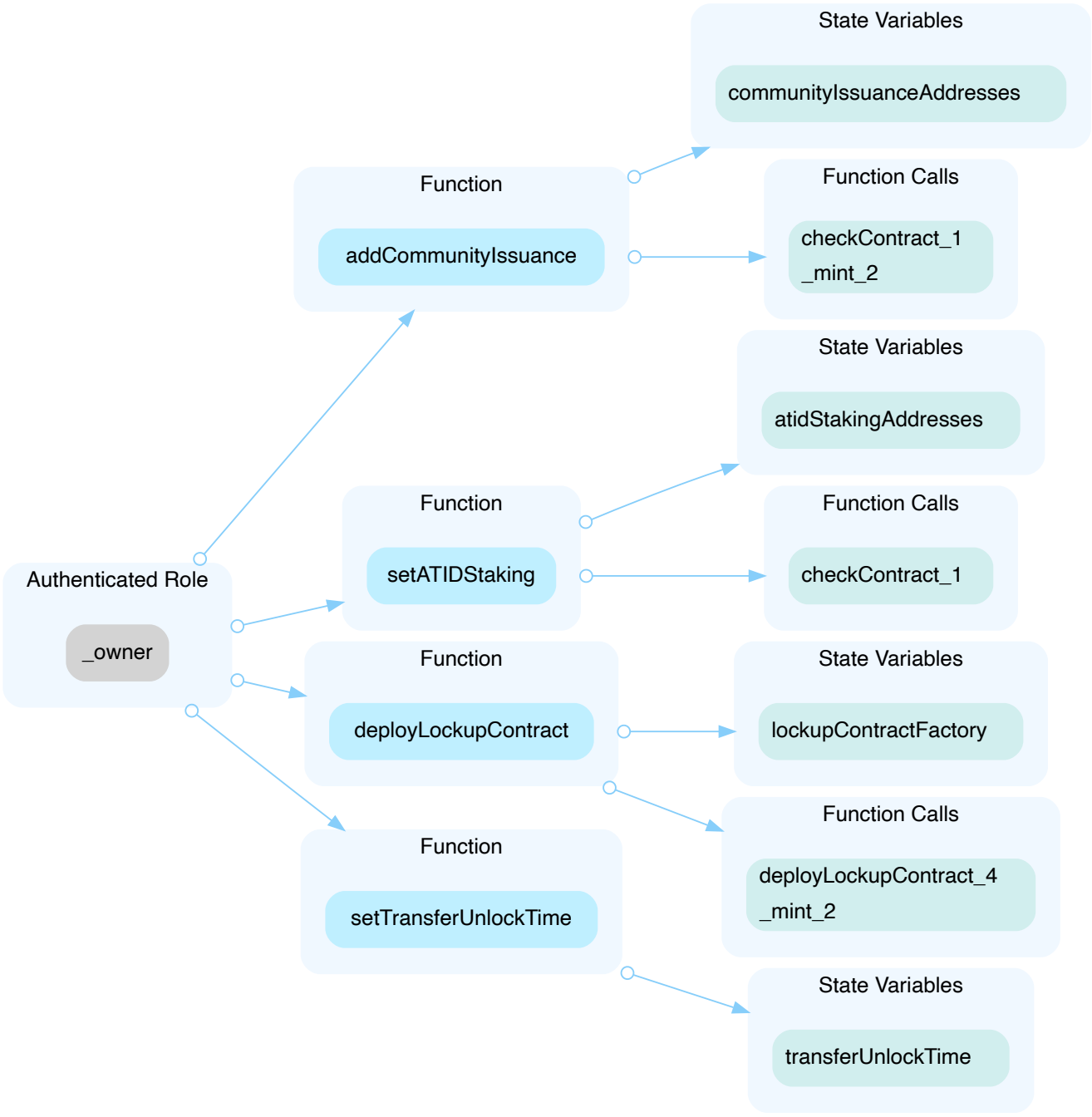
In the contract `ATIDStaking` the role `vaultManagerAddress` has authority over the functions shown in the diagram below.

Any compromise to the `vaultManagerAddress` account may allow the hacker to take advantage of this authority.



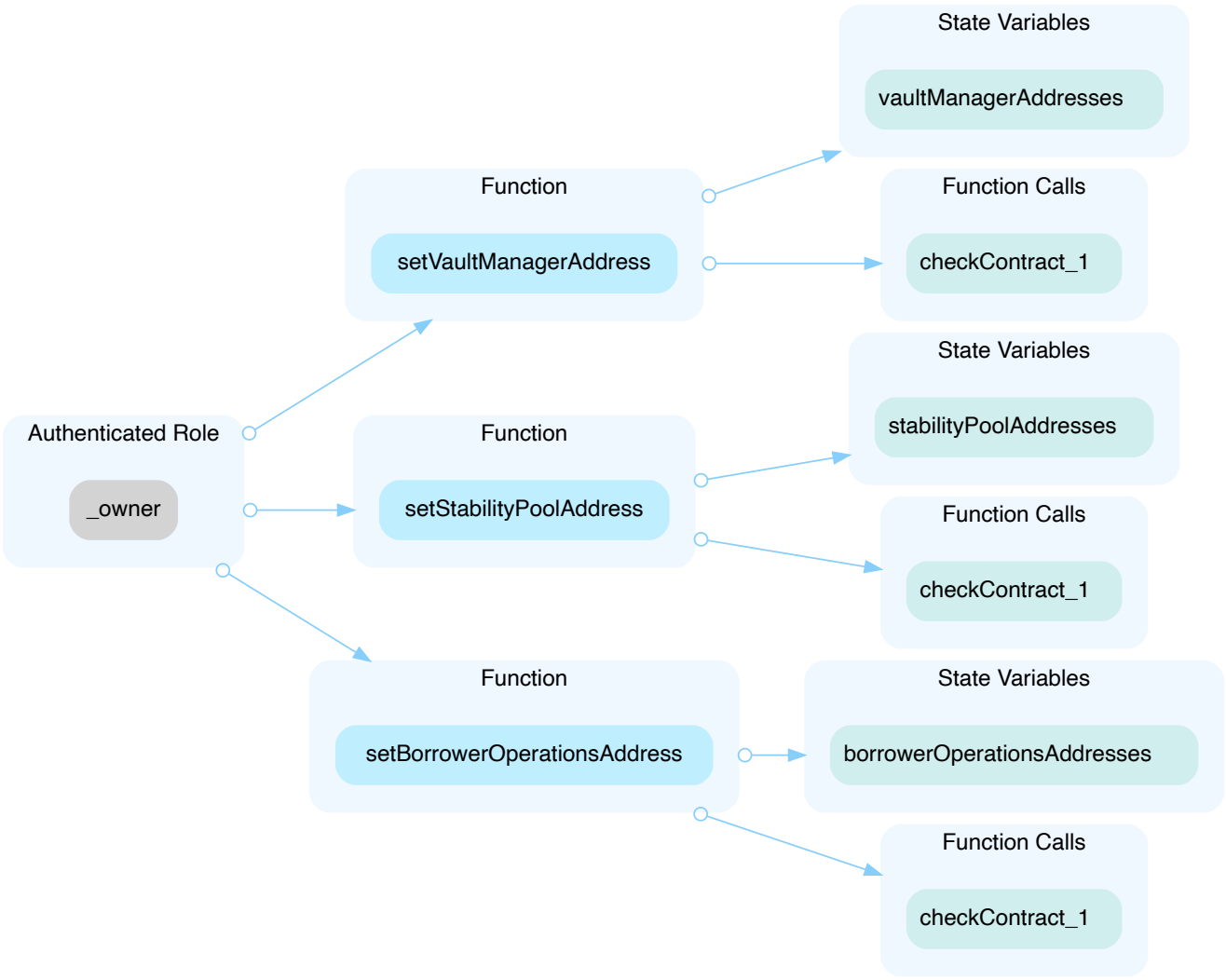
In the contract `ATIDToken` the role `_owner` has authority over the functions shown in the diagram below.

Any compromise to the `_owner` account may allow the hacker to take advantage of this authority.



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

Any compromise to the `_owner` account may allow the hacker to take advantage of this authority.



In the contract `BAIToken`, the role `borrowerOperationsAddresses` has authority over the following functions:

- mint()

In the contract `[BAIToken]`, the role `vaultManagerAddresses` has authority over the following functions:

- burn()
- returnFromPool()

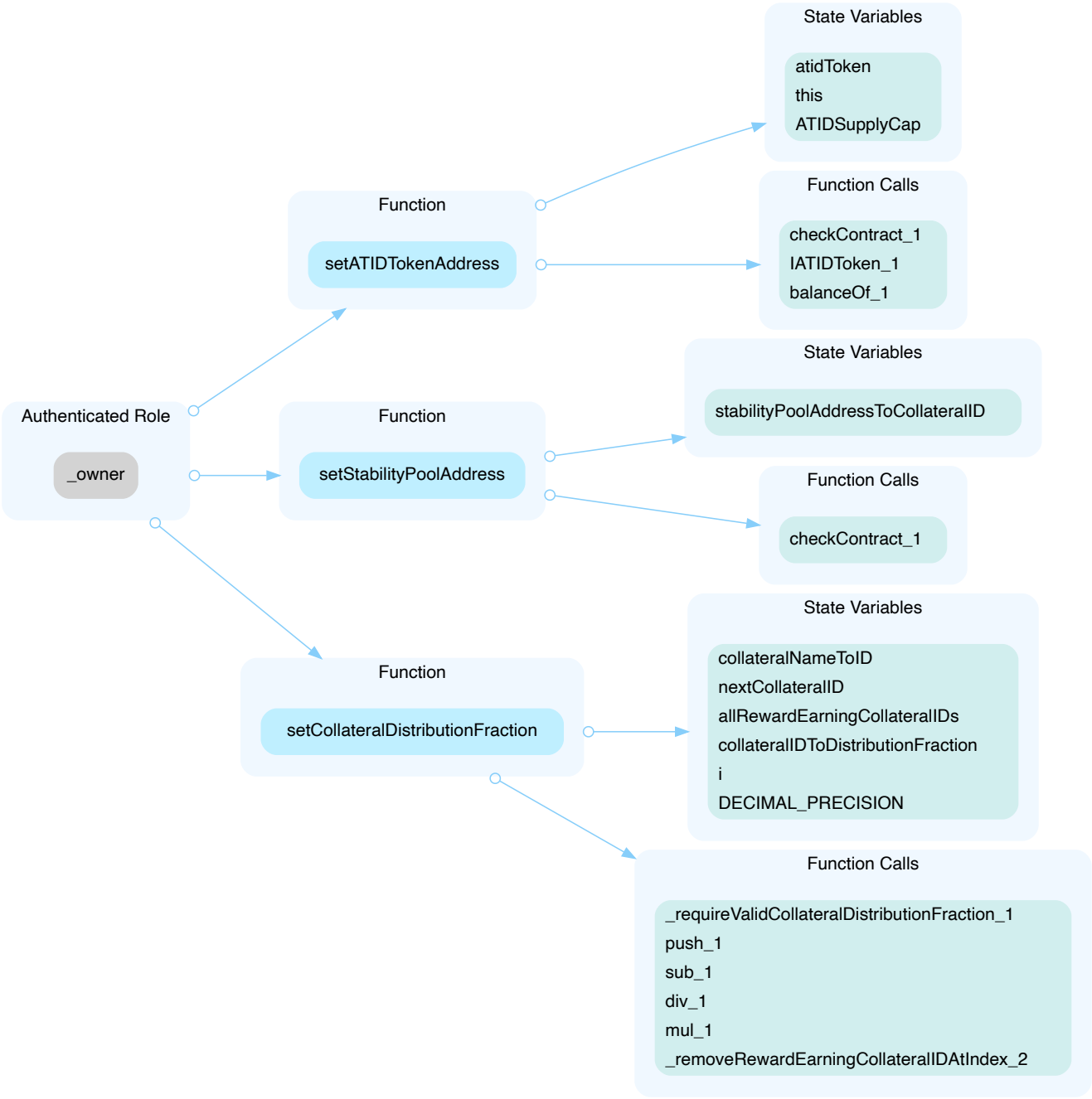
In the contract `[BAIToken]`, the role `stabilityPoolAddresses` has authority over the following functions:

- burn()
- sendToPool()
- returnFromPool()

Any compromise to the `borrowerOperationsAddresses`, `vaultManagerAddresses`, and `stabilityPoolAddresses` accounts may allow the hacker to take advantage of this authority.

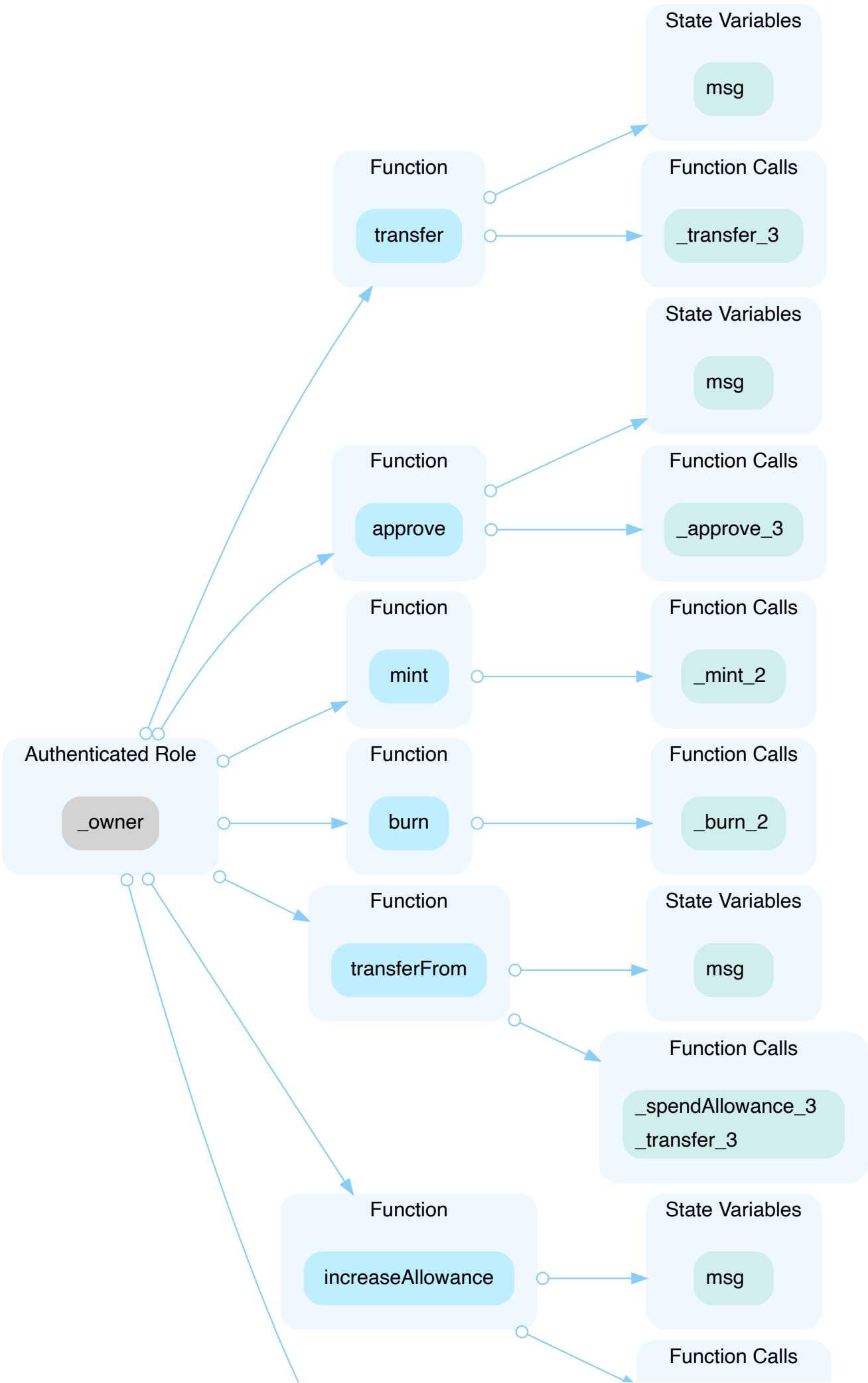
In the contract `CommunityIssuance` the role `_owner` has authority over the functions shown in the diagram below.

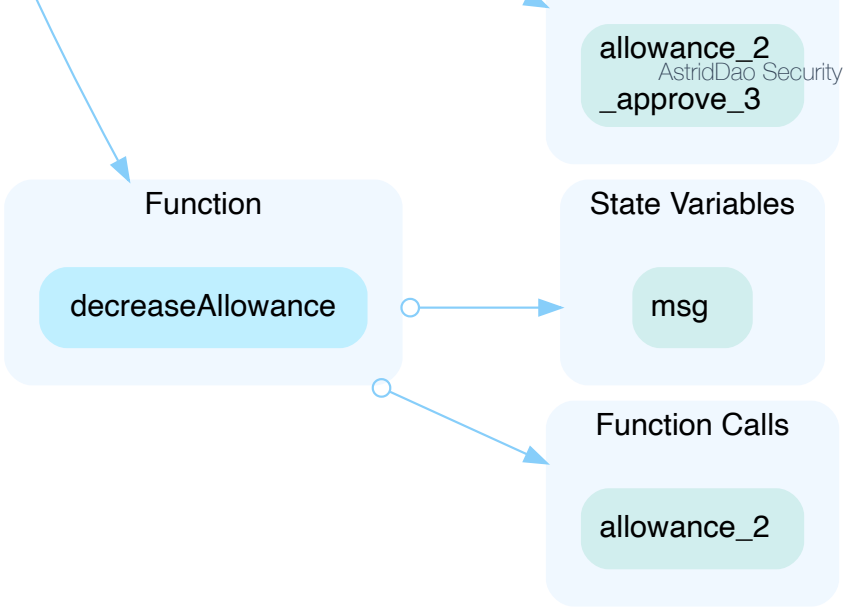
Any compromise to the `_owner` account may allow the hacker to take advantage of this authority.



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

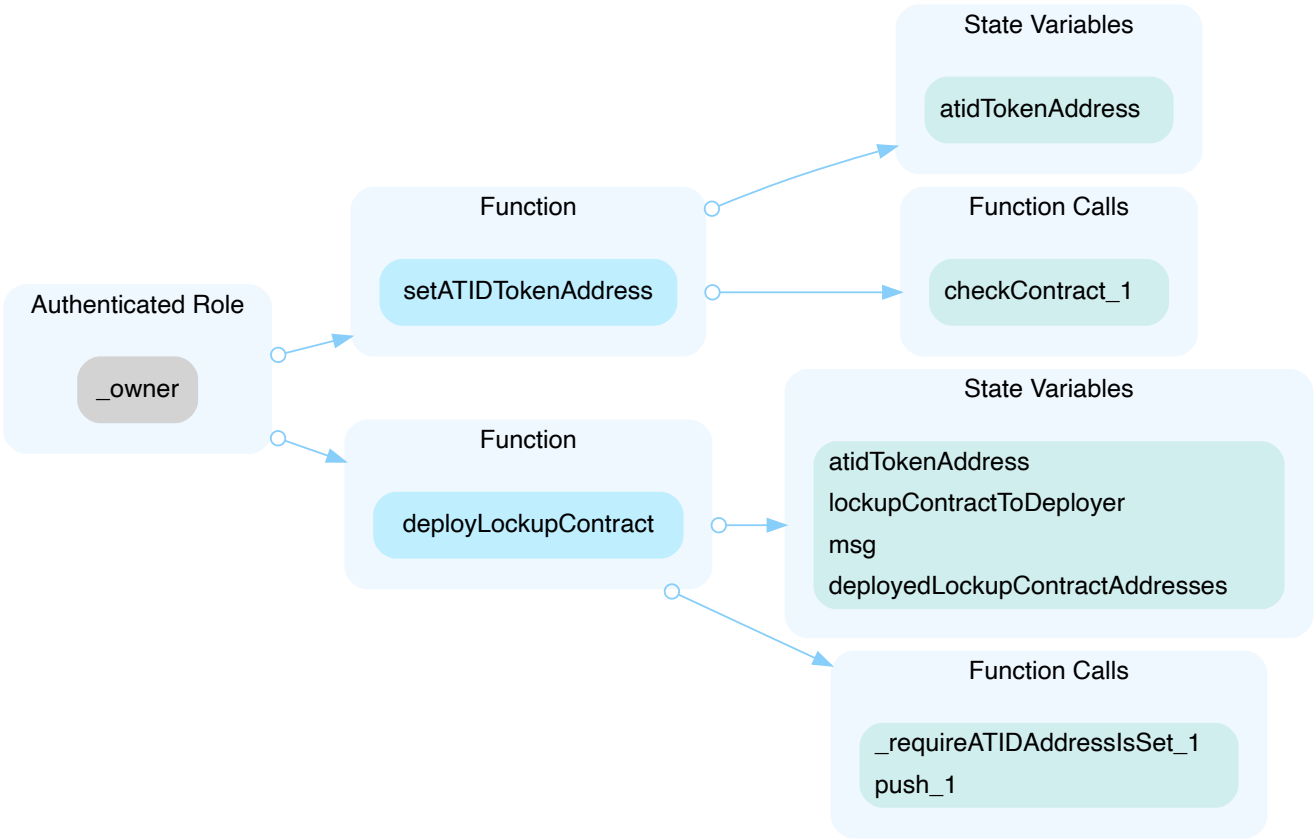
Any compromise to the `_owner` account may allow the hacker to take advantage of this authority.


State Variables
msg
Function
transfer
Function Calls
_transfer_3
State Variables
msg
Function
approve
Function Calls
_approve_3
Function
mint
Function Calls
_mint_2
Authenticated Role
_owner
Function
burn
Function Calls
_burn_2
Function
transferFrom
State Variables
msg
Function Calls
_spendAllowance_3
_transfer_3
Function
increaseAllowance
State Variables
msg
Function Calls



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

Any compromise to the `_owner` account may allow the hacker to take advantage of this authority.



Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential

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

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

Short Term:

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

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

Long Term:

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

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

Permanent:

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

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

Alleviation

[AstridDAO Team]: Ownership is necessary so that (while for now managed by our team and necessary for enabling multi-sig support) we can incorporate a contract in the future to support on-chain governance (by having only a well defined set of calls that can be automatically triggered to adjust certain contract parameters and perform operations).

ADC-01 | Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	ATID/ATIDToken.sol: 160, 170, 179, 195 ATID/ATIDStaking.sol: 357, 361, 365 ATID/CommunityIssuance.sol: 139 Dependencies/AstridBase.sol: 100, 104, 108, 111, 114, 117, 120, 130 BAIToken.sol: 97, 101, 105	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

The AstridDao team has modified the code in commit `6f3569cf53ea037ec1f1bb83a10cf3bfe10120cb`.

ADC-02 | Unlocked Compiler Version

Category	Severity	Location	Status
Language Specific	Informational	ATID/LockupContract.sol: 3 ATID/ATIDToken.sol: 3 ATID/GovToken.sol: 4 ATID/LockupContractFactory.sol: 3 ATID/ATIDStaking.sol: 3 ATID/CommunityIssuance.sol: 3 BAIToken.sol: 3	Resolved

Description

The contract has unlocked compiler version. An unlocked compiler version in the source code of the contract permits the user to compile it at or above a particular version. This, in turn, leads to differences in the generated bytecode between compilations due to different compiler versions. This can lead to an ambiguity when debugging as compiler specific bugs may occur in the codebase that would be hard to identify over a span of multiple compiler versions rather than a specific one.

Recommendation

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

```
pragma solidity 0.8.0;
```

Alleviation

The AstridDao team used the compiler version `pragma solidity 0.8.13;` in commit `6f3569cf53ea037ec1f1bb83a10cf3bfe10120cb`.

ADC-03 | Missing Error Messages

Category	Severity	Location	Status
Coding Style	Informational	ATID/CommunityIssuance.sol: 115, 300 BAIToken.sol: 227, 228, 236, 244, 252, 253	Acknowledged

Description

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

Recommendation

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

Alleviation

The AstridDao team acknowledged this finding.

ATD-01 | Third Party Dependencies

Category	Severity	Location	Status
Logical Issue	Minor	ATID/ATIDStaking.sol: 69, 353	Acknowledged

Description

The contract is serving as the underlying entity to interact with third-party `colToken` protocols. The scope of the audit treats 3rd party entities as black boxes and assumes their functional correctness. However, in the real world, 3rd parties can be compromised and this may lead to lost or stolen assets. In addition, upgrades of 3rd 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 `ATIDStaking` requires interaction with `colToken`. We encourage the team to constantly monitor the statuses of 3rd parties to mitigate the side effects when unexpected activities are observed.

Alleviation

The AstridDao team acknowledged this finding.

ATD-02 | Discussion On The `ATIDStaking` Contract

Category	Severity	Location	Status
Logical Issue	Minor	ATID/ATIDStaking.sol: 214	Acknowledged

Description

The parameter variable `_lockedUntil` is unrestricted and can be the history time, is this in accordance with the design intent?

What is the rule for calling the `increaseF_COL()` and `increaseF_BAI()` functions? And this rule should be transparent to the community.

Recommendation

Review the code logic to ensure it meets design intent.

Alleviation

[AstridDao Team]:

1. A historical timestamp means the locked stake can be unlocked right away.
2. These two calls are for accounting for the overall fees received by the ATIDStaking contract from associated contracts (we take fees during borrowing and send them to `ATIDStaking` contract for `ATID` stakers to claim). They should have been only callable by the parties mentioned in the first line of their implementation (`_require*` calls)

ATD-03 | Redundant Code Components

Category	Severity	Location	Status
Volatile Code	Informational	ATID/ATIDStaking.sol: 365~367	Acknowledged

Description

The linked statements do not affect the functionality of the codebase and appear to be either leftovers from test code or older functionality.

Recommendation

We advise to remove the redundant statements for production environments.

Alleviation

The AstridDao team acknowledged this finding.

ATD-04 | Discussion On `unstakeLocked()`

Category	Severity	Location	Status
Logical Issue	Informational	ATID/ATIDStaking.sol: 254	Resolved

Description

In the `unstakeLocked()` function, if an invalid locked stake ID is provided, such as 0, the user still gets their accumulated BAI and COL earnings.

But the comment on line 266 is "If this condition is false, the user is basically doing a fee top-up".

Is this consistent with the design intent?

Recommendation

Financial models of blockchain protocols need to be resilient to attacks. They need to pass simulations and verifications to guarantee the security of the overall protocol.

The financial model of this protocol is not in the scope of this audit.

Alleviation

The AstridDao team has modified the code comments in commit `6f3569cf53ea037ec1f1bb83a10cf3bfe10120cb`.

ATI-01 | Unchecked ERC-20 `transfer()` / `transferFrom()` Call

Category	Severity	Location	Status
Volatile Code	Minor	ATID/ATIDStaking.sol: 244, 271, 280 ATID/CommunityIssuance.sol: 308 ATID/LockupContract.sol: 101	Resolved

Description

The return value of the transfer()/transferFrom() call is not checked.

File: projects/AstridDao/contracts/ATID/ATIDStaking.sol (Line 244, Function `ATIDStaking.stakeLocked`)

```
            baiToken.transfer(msg.sender, BAIGain);
```

File: projects/AstridDao/contracts/ATID/ATIDStaking.sol (Line 271, Function `ATIDStaking.unstakeLocked`)

```
            atidToken.transfer(msg.sender, ATIDToWithdraw);
```

File: projects/AstridDao/contracts/ATID/ATIDStaking.sol (Line 280, Function `ATIDStaking.unstakeLocked`)

```
        baiToken.transfer(msg.sender, BAIGain);
```

File: projects/AstridDao/contracts/ATID/CommunityIssuance.sol (Line 308, Function `CommunityIssuance.sendATID`)

```
        atidToken.transfer(_account, _ATIDamount);
```

File: projects/AstridDao/contracts/ATID/LockupContract.sol (Line 101, Function `LockupContract.withdrawATID`)

```
        atidTokenCached.transfer(beneficiary, amount);
```

Recommendation

Since some ERC-20 tokens return no values and others return a `bool` value, they should be handled with care. We advise using the OpenZeppelin's `SafeERC20.sol` implementation to interact with the `transfer()` and `transferFrom()` functions of external ERC-20 tokens. The OpenZeppelin implementation checks for the existence of a return value and reverts if `false` is returned, making it compatible with all ERC-20 token implementations.

Alleviation

The AstridDao team has modified the code in commit `6f3569cf53ea037ec1f1bb83a10cf3bfe10120cb`.

CIA-01 | Tautology Or Contradiction

Category	Severity	Location	Status
Logical Issue	Major	ATID/CommunityIssuance.sol: 314	Resolved

Description

Comparisons that are always true or always false may be incorrect. This means any user can call functions `sendATID()` and `issueATID()`.

File: projects/AstridDao/contracts/ATID/CommunityIssuance.sol (Line 314, Function `CommunityIssuance._requireCallerIsStabilityPool`)

```
require(stabilityPoolAddressToCollateralID[msg.sender] >= 0, "CommunityIssuance: caller is not an SP");
```

Recommendation

We recommend fixing the incorrect comparison by changing the value type or the comparison operator.

Alleviation

The AstridDao team has modified the code in commit `6f3569cf53ea037ec1f1bb83a10cf3bfe10120cb`.

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.

Language Specific

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

Coding Style

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

Checksum Calculation Method

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

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

Disclaimer

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

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

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

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

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

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

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

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

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

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

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

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

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

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

About

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