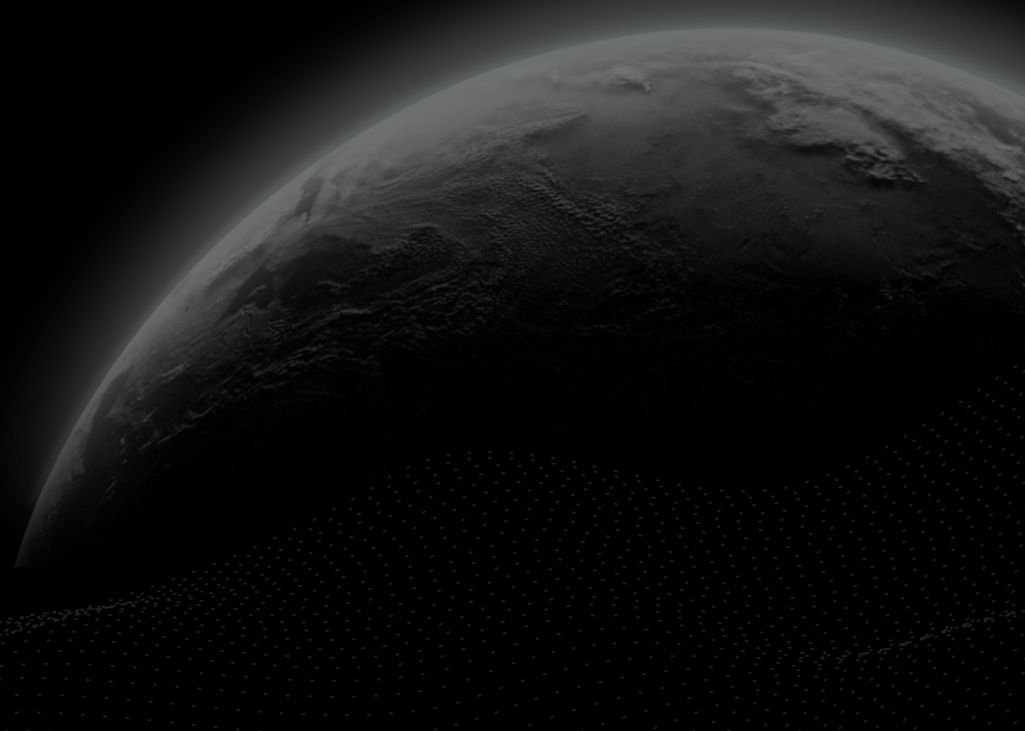
CERTIK

Security Assessment

Arcana Network

CertiK Verified on Dec 28th, 2022



CertiK Verified on Dec 28th, 2022

Arcana Network

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

Executive Summary

TYPES	ECOSYSTEM	METHODS
Other-Contract	Polygon	Manual Review, Static Analysis
LANGUAGE	**TIMELINE**	**KEY COMPONENTS**
Solidity	Delivered on 12/28/2022	N/A

CODEBASE
https://github.com/arcana-network/arcana-smart-contract/tree/audit-only
...View All

COMMITS
69fc4e3ee96488a3fa6edb8a749fcd7ad2f859e5
...View All

Vulnerability Summary

13	11	0	0	2	0	0
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined	Unresolved

0	Critical		Critical risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
2	Major	2 Acknowledged	Major risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
1	Medium	1 Resolved	Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
6	Minor	6 Resolved	Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.
4	Informational	4 Resolved	Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

TABLE OF CONTENTS | ARCANA NETWORK

CODEBASE | ARCANA NETWORK

Repository

https://github.com/arcana-network/arcana-smart-contract/tree/audit-only

Commit

69fc4e3ee96488a3fa6edb8a749fcd7ad2f859e5

AUDIT SCOPE | ARCANA NETWORK

17 files audited ● 7 files with Acknowledged findings ● 10 files with Resolved findings

ID	File	SHA256 Checksum
● ARA	Arcana.sol	c6040ed0762796037377685a11f8dbeff779330b05acbab61b39019621ea150b
● ABU	ArcanaBeacon.sol	0959630f607f6a6795392c7aba1f84dcf8a550f0a7bef68e211d4464d821a426
● DIC	DID.sol	9d65e107d12bf6fd6d7e1266243142eb2ada499f9f3904b47ccd002ad8420d69
● ERU	ERC2771ContextUpgradeable.sol	19bde90ccedd1fad670fce73aa96780b85e8edb483526c5b60fabbffcea083b6
● FAT	Factory.sol	9220774a9f309302db711a5dd08b74c7c20137937f6e687fd21334f47ffb5601
● FOW	Forwarder.sol	d2f62f144dfde3c0d4ab5b75071533733b3ad2b7340f1f8bb1b5917035c0f555
● NLU	NodeList.sol	eecd705b54761789859956129cf793cab0184f10df7d007c7116ed999d8d30c6
● IAU	interfaces/IArcana.sol	61bf1b45ed92005477956af614f7b9ef20646c71a2569cc614a18f9754fe9f2c
● IAD	interfaces/IArcanaDID.sol	44a2858406534b673451c1aeed3b1d073a7a96eee24220c938154be7da1c21d3
● IAF	interfaces/IArcanaFactory.sol	bfa8a3b61ae12ed453944b45d52dce2907912ae5c8e1566a18cbc6b8e23d42b1
● IDD	interfaces/IDID.sol	d89897c5fc2e9762e6684eb3d81d9b14b554286b5cdd2d40f8bce561c1802a3a
● IFH	interfaces/IFactory.sol	c5f90651edc6e6762f8d7ebda8cdcd4e3f0b077fbe1c8d5c05b0013e6278378b
● IFA	interfaces/IFactoryArcana.sol	c1e7b0f910154dcfa690b03c3eaf74017074fe9682b1d6569f2848cfb9ce1852
● IFD	interfaces/IFactoryDID.sol	faf06cf66b284b40a6506a6a9476758071041d964cdc6f492372146f65e74324

ID	File	SHA256 Checksum
IFF	interfaces/IFactoryForwarder.sol	488558dda9592150482899d4e28ca93eb237c663c112585ae6bb7626b77832e3
IFT	interfaces/IForwarder.sol	31cdedb272f9263a571f14e01b81ddbf9be135c241fbdd093a83e82ac61d0628
RLU	RoleLib.sol	566cc9c77e6b9c1da4fca033b682f356648b9517106e458318be7a16d931be63

APPROACH & METHODS | ARCANA NETWORK

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

The auditing process pays special attention to the following considerations:

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

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

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

FINDINGS | ARCANA NETWORK

13	0	2	1	6	4
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Arcana Network. Through this audit, we have uncovered 13 issues ranging from different severity levels. Utilizing the techniques of Manual Review & Static Analysis to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
CON-01	**Centralization Risks In Arcana.Sol**	**Centralization / Privilege**	**Major**	**Acknowledged**
CON-02	**Centralized Control Of Contract Upgrade**	**Centralization / Privilege**	**Major**	**Acknowledged**
ARA-01	Logical Issue Of The `download()`	Logical Issue	Medium	Resolved
ARA-02	The `bandwidth` Limit Not Checked In `download()`	Logical Issue	Minor	Resolved
ARA-03	The `ephemeralWallet` And `ephemeralAddress` Not Used	Logical Issue	Minor	Resolved
DIC-01	The Use Of `controlRules[1]`	Logical Issue	Minor	Resolved
DIC-02	The `files[did].version` Not Used	Logical Issue	Minor	Resolved
NLU-01	Unprotected Initializer	Coding Style	Minor	Resolved
NLU-02	Logical Issue Of `updateEpoch()`	Logical Issue	Minor	Resolved
CON-03	Missing Emit Events	Coding Style	Informational	Resolved

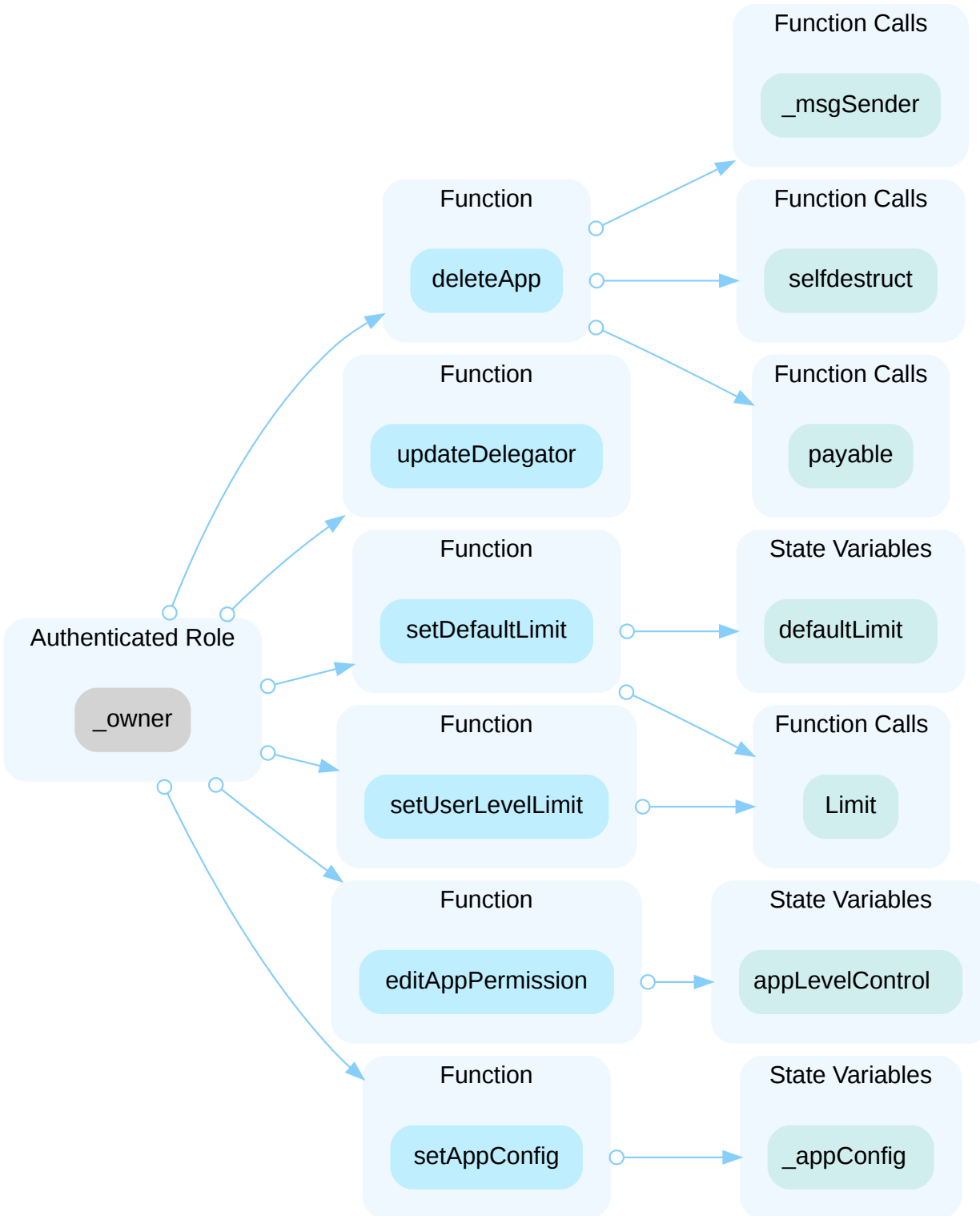
ID	Title	Category	Severity	Status
CON-04	Declaration Naming Convention	Coding Style	Informational	Resolved
CON-05	Solidity Version Not Recommended	Language Specific	Informational	Resolved
CON-06	Different Solidity Versions	Language Specific	Informational	Resolved

CON-01 | CENTRALIZATION RISKS IN ARCANA.SOL

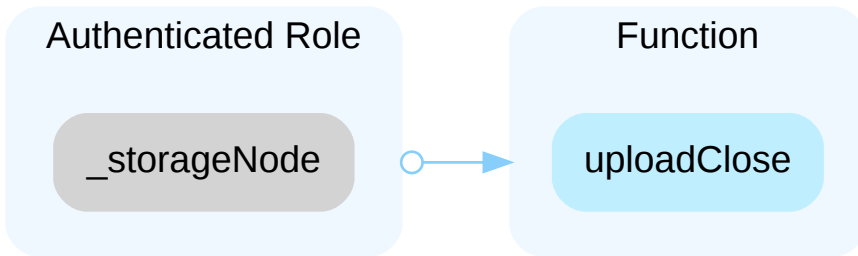
Category	Severity	Location	Status
Centralization / Privilege	**Major**	**Arcana.sol: 146, 160, 260, 340, 353, 384, 402; ArcanaBeacon.sol: 23; DID.sol: 51; Factory.sol: 60, 74, 81, 142, 152, 176; Forwarder.sol: 47, 100; NodeList.sol: 60, 129, 137, 145, 171**	**Acknowledged**

Description

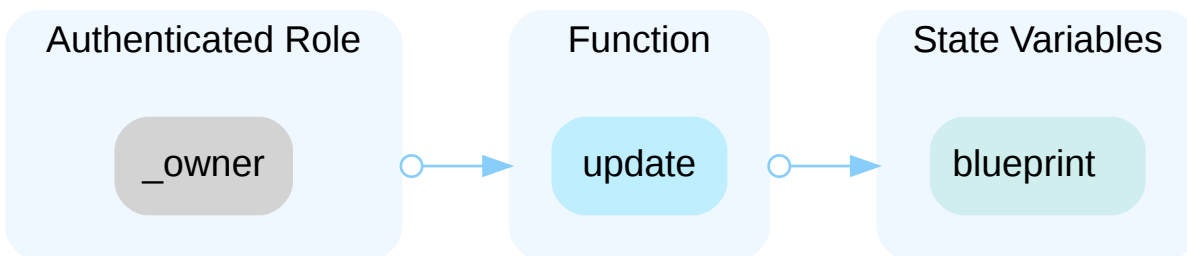
In the contract `Arcana` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority.



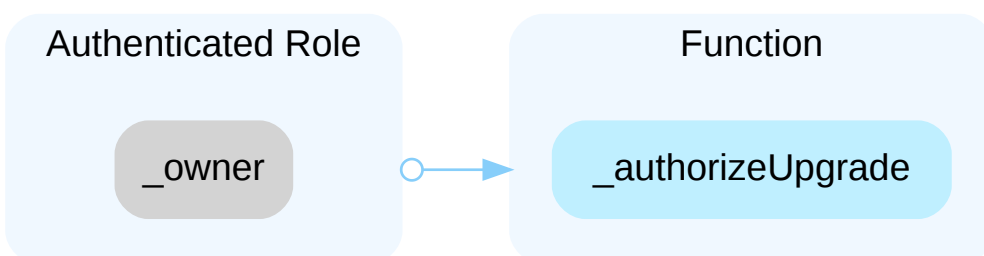
In the contract `Arcana` the role `_storageNode` has authority over the functions shown in the diagram below. Any compromise to the `_storageNode` account may allow the hacker to take advantage of this authority.



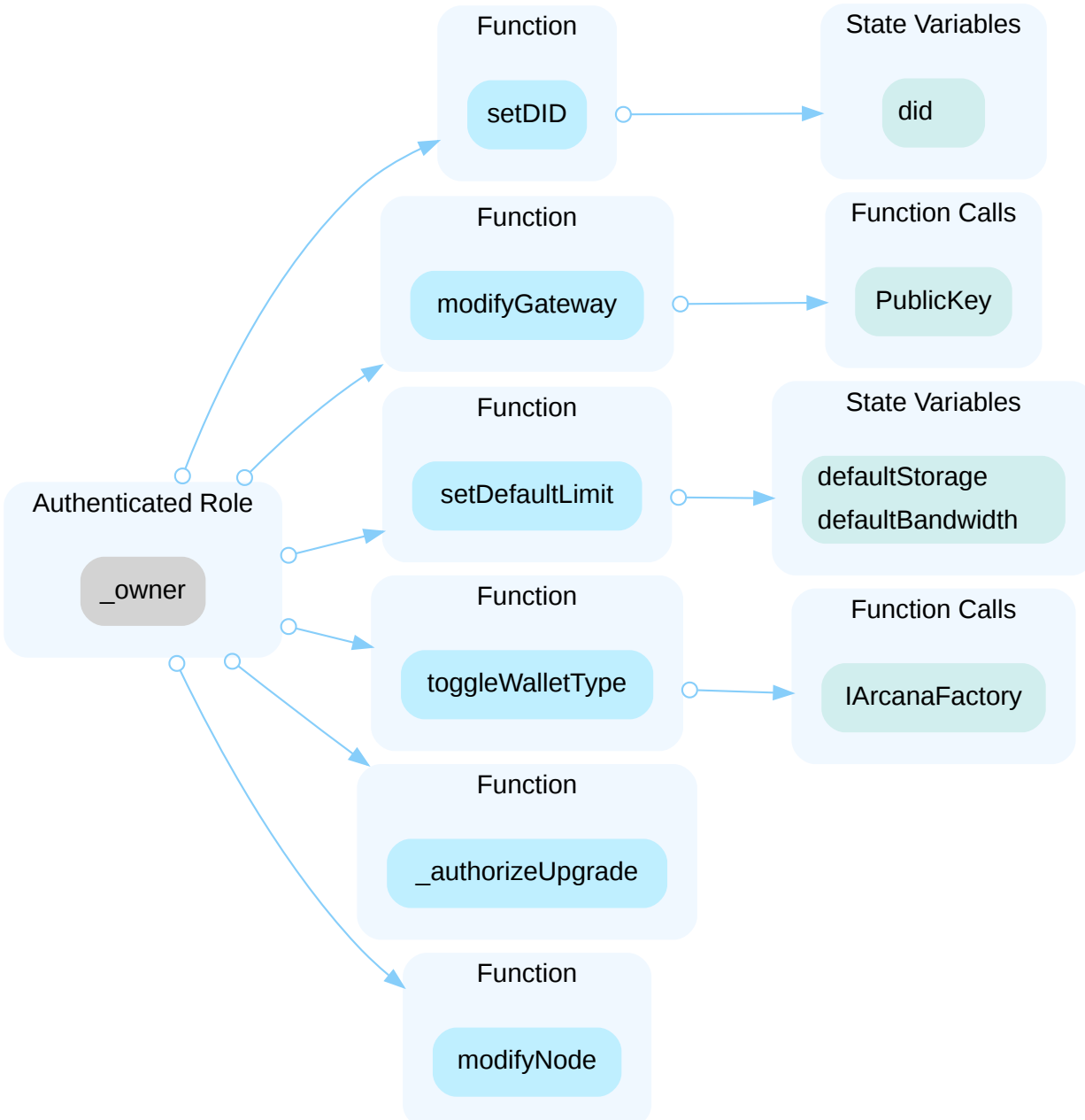
In the contract `ArcanaBeacon` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority.



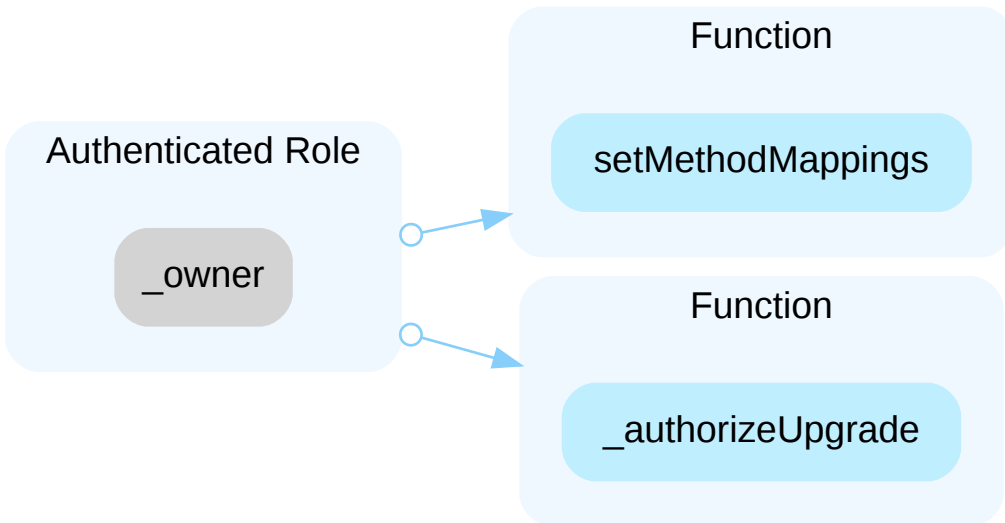
In the contract `DID` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority.



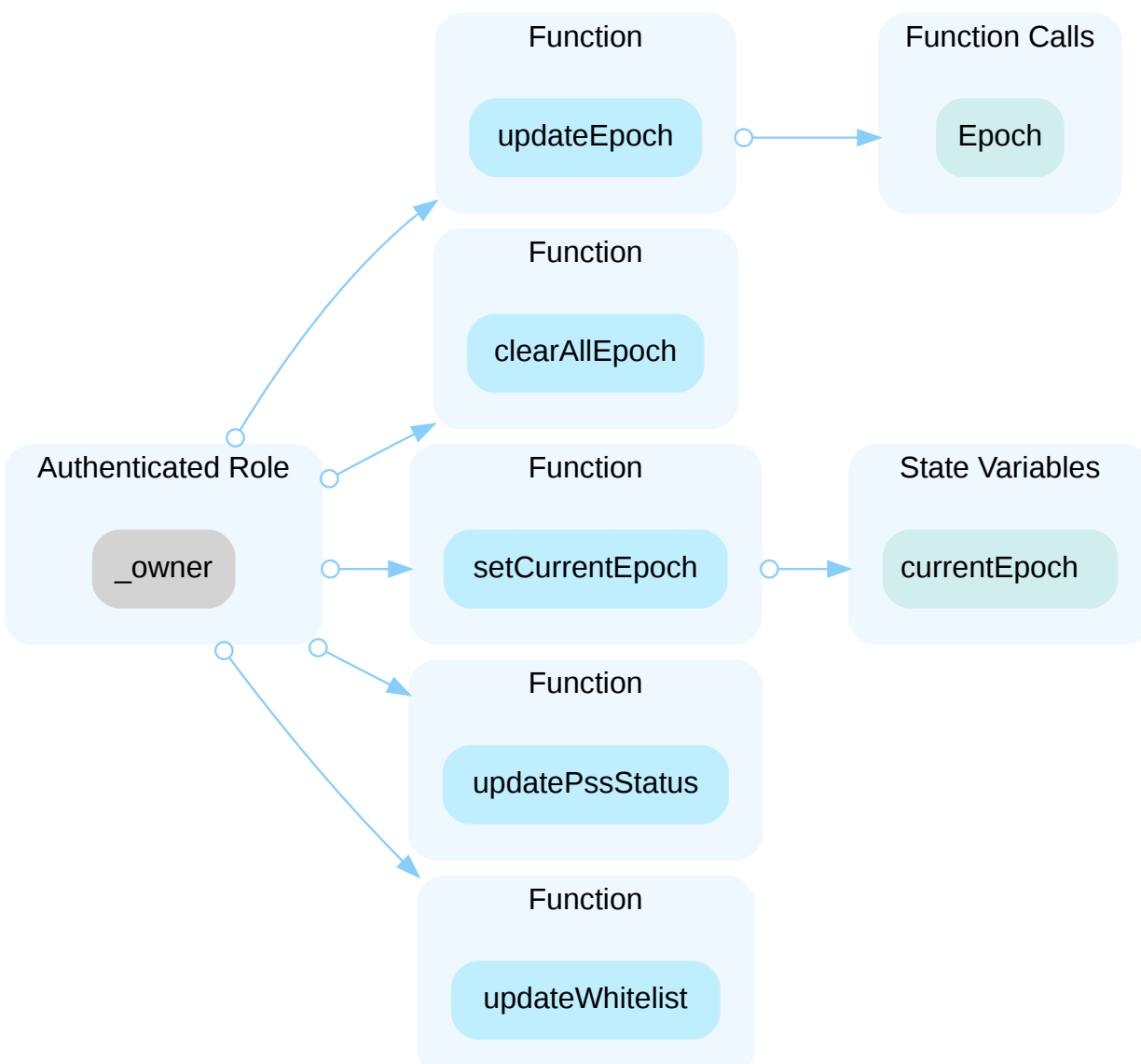
In the contract `Factory` the role `_owner` has authority over the functions shown in the diagram below. Any compromise to the `_owner` account may allow the hacker to take advantage of this authority.



In the contract `Forwarder` 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 `NodeList` 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

The team acknowledged this issue and stated that:

"We have 2 different types of contracts:

- Factory, Nodelist, Forwarder, DID all these contracts are owned by us.
- Arcana is owned by developers.

For contracts owned by us, we can use multi-sig wallet to deploy the contracts, and hence the owner will be a multi-sig wallet instead of a single EOA. In the future, we will move towards DAO/governance/voting module. For Arcana contract developers can use MFA(Multi-Factor Authentication). In MFA, the key is formed by 2 keys out of 3. So even if a user loses one key he can generate a new share through the remaining key."

CON-02 | CENTRALIZED CONTROL OF CONTRACT UPGRADE

Category	Severity	Location	Status
Centralization / Privilege	**Major**	**Arcana.sol: 15; DID.sol: 17; ERC2771ContextUpgradeable.sol: 12; Factory.sol: 17; Forwarder.sol: 17; NodeList.sol: 6**	**Acknowledged**

Description

The aforementioned contracts are upgradeable contracts, the owner can upgrade the contract without the community's commitment. If an attacker compromises the account, he can change the implementation of the contract and drain tokens from the contract.

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

The team acknowledged this issue and stated that:

"We have 2 different types of contracts:

- Factory, Nodelist, Forwarder, DID all these contracts are owned by us.
- Arcana is owned by developers.

For contracts owned by us, we can use multi-sig wallet to deploy the contracts, and hence the owner will be a multi-sig wallet instead of a single EOA. In the future, we will move towards DAO/governance/voting module. For Arcana contract developers can use MFA(Multi-Factor Authentication). In MFA, the key is formed by 2 keys out of 3. So even if a user loses one key he can generate a new share through the remaining key."

ARA-01 | LOGICAL ISSUE OF THE `download()`

Category	Severity	Location	Status
Logical Issue	Medium	Arcana.sol: 277	Resolved

Description

Anyone, including users who are not the owner or delegator, is able to call the 'download()' function and access the designated file by using the app's bandwidth and `ruleset`. As a result, it appears that the permission system is not functioning correctly in this instance, as accounts that the owner does not share with are still able to download the file.

Recommendation

We advise the client to add checks if the `_msgSender()` is not delegator or the file owner.

Alleviation

The client has confirmed that the current implementation aligns with their original project design, and they have provided the following illustration to further clarify their design for the 'download' feature:

"There are 2 types of people who can download a file.

- On-chain access: Owner or delegator.
- Off-chain access: If a user is specified in the rule.

On-chain access is checked on the chain through checkPermission, i.e if checkPermission executes properly then it emits filePermission event, and storage nodes find if this event is emitted in the transaction then the file is given to them.

In Off-chain access, the event is not emitted but the download function is successfully executed, storage nodes check for access in the Merkle tree and if the user exists in the Merkle tree only then he is given the file and the downloadClose transaction is executed.

In this way, although the transaction gets executed successfully for everyone it doesn't mean they will get the file, there are checks on the storage node for off-chain access checks."

ARA-02 | THE `bandwidth` LIMIT NOT CHECKED IN `download()`

Category	Severity	Location	Status
Logical Issue	Minor	Arcana.sol: 277	Resolved

Description

The `bandwidth` limits of both user level and app level should be checked in `download()`.

Recommendation

We advise the client to add the aforementioned limits.

Alleviation

The team heeded our advice and resolved this issue in commit `a84d9fea9a626806b641f4c6139b0aedf471408c`.

ARA-03 | THE `ephemeralWallet` AND `ephemeralAddress` NOT USED

Category	Severity	Location	Status
Logical Issue	Minor	Arcana.sol: 243, 277	Resolved

Description

Based on the code comments, it appears that the `ephemeralWallet` and `ephemeralAddress` are intended to be used for signing the message in the upload transaction. However, upon review, we noticed that it is not implemented in the corresponding functions.

Recommendation

We advise the client to modify the code as the aforementioned information.

Alleviation

The client has confirmed that the implementation aligns with their original project design, and they have provided the following illustration to further clarify their design for the `ephemeralWallet` and `ephemeralAddress` features:

"`ephemeralWallet` is used for signing non-blockchain related messages, for eg: it is used to generate signatures that will prove that they own the private key. `ephemeralWallet` is used in communication between the client, DKG, and storage node. It is not used in any smart contract.

We need `ephemeralAddress` in the smart contract just to associate a key with an operation, e.g. during upload when we specify X as `ephemeralAddress` then dkg nodes while storing the key for upload operation will check whether the request to store the keys is signed by the wallet X."

DIC-01 | THE USE OF `controlRules[1]`

Category	Severity	Location	Status
Logical Issue	Minor	DID.sol: 133	Resolved

Description

The `controlRules[1]` is used to record the download rule set according to the comment says, however, it is not checked either in the `downloadNFT()` or `download()`.

Recommendation

We would like to confirm with the client if this aligns with the original project design.

Alleviation

The team acknowledged this issue and stated this aligns with the original project design:

"In the **download()** function, the owner and delegators are checked throw **checkPermission()** function in DID.sol, this function checks for controlRules[1].

Here it checks if delegator has access to specified control rules and in case of download the control is 1.

In case of downloadNFT, we are doing off-chain check. "

DIC-02 | THE `files[did].version` NOT USED

Category	Severity	Location	Status
Logical Issue	Minor	DID.sol: 194	Resolved

Description

The `files[did].version` would automatically increase one once the file owner changed, however, it is not used to check anything in the contract, nor could it be viewed by `getFile()`.

Recommendation

We would like to confirm with the client if this aligns with the original project design.

Alleviation

The team heeded our advice and resolved this issue in commit `a84d9fea9a626806b641f4c6139b0aedf471408c`.

NLU-01 | UNPROTECTED INITIALIZER

Category	Severity	Location	Status
Coding Style	Minor	NodeList.sol: 55	Resolved

Description

One or more logic contracts do not protect their initializers. An attacker can call the initializer and assume ownership of the logic contract, whereby he/she can perform privileged operations that trick unsuspecting users into believing that he/she is the owner of the upgradeable contract.

```
6  contract NodeList is OwnableUpgradeable {
```

- `NodeList` is an upgradeable contract that does not protect its initializer.

```
55      function initialize(uint256 epoch) public initializer {
```

- `initialize` is an unprotected initializer function.

Recommendation

We advise calling `_disableInitialize` in the constructor or giving the constructor the `initializer` modifier to prevent the initializer from being called on the logic contract.

Reference: https://docs.openzeppelin.com/upgrades-plugins/1.x/writing-upgradeable#initializing_the_implementation_contract

Alleviation

The team stated that they are going to conduct the deployment via scripts to ensure the initialize function is automatically called when the proxy contract is deployed, and cannot be called again.

They also removed self destruct and added `_disableInitializers` in commit 21f52e16900a524c1c3e0c5498f806a46070d8bb.

NLU-02 | LOGICAL ISSUE OF `updateEpoch()`

Category	Severity	Location	Status
Logical Issue	Minor	NodeList.sol: 145	Resolved

Description

When invoking `updateEpoch()` , the `whitelist[epoch][nodeAddress]` should be checked.

Recommendation

We advise the client to add the aforementioned checks.

Alleviation

The team acknowledged this issue and decided not to change as it is part of the design.

CON-03 | MISSING EMIT EVENTS

Category	Severity	Location	Status
Coding Style	Informational	Arcana.sol: 146, 160, 260, 340, 353, 384; ArcanaBeacon.sol: 23; DID.sol: 51; Factory.sol: 60, 74, 81, 142, 152, 176; Forwarder.sol: 47, 100; NodeList.sol: 129, 137, 145, 171	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 team heeded our advice and resolved this issue in commit `a84d9fea9a626806b641f4c6139b0aedf471408c`.

CON-04 | DECLARATION NAMING CONVENTION

Category	Severity	Location	Status
Coding Style	Informational	Arcana.sol: 71; DID.sol: 35; ERC2771ContextUpgradeable.sol: 16, 22, 30; Factory.sol: 49; Forwarder.sol: 31, 47, 47	Resolved

Description

One or more declarations do not conform to the Solidity style guide with regards to its naming convention.

Recommendation

We recommend adjusting those variable and function names to properly conform to Solidity's naming convention.

Alleviation

The team heeded our advice and resolved this issue in commit `a84d9fea9a626806b641f4c6139b0aedf471408c`.

CON-05 | SOLIDITY VERSION NOT RECOMMENDED

Category	Severity	Location	Status
Language Specific	Informational	Arcana.sol: 2; ArcanaBeacon.sol: 2; DID.sol: 2; ERC2771ContextUpgradeable.sol: 4; Factory.sol: 2; Forwarder.sol: 2; NodeList.sol: 2; RoleLib.sol: 2; interfaces/IArcana.sol: 2; interfaces/IArcanaDID.sol: 2; interfaces/IArcanaFactory.sol: 2; interfaces/IDID.sol: 2; interfaces/IFactory.sol: 2; interfaces/IFactoryArcana.sol: 2; interfaces/IFactoryDID.sol: 2; interfaces/IFactoryForwarder.sol: 2; interfaces/IForwarder.sol: 2	Resolved

Description

Solidity frequently releases new compiler versions. Using an old version prevents access to new Solidity security features. Also, recent versions may be too early to be trusted.

```
solc-0.8.17 is not recommended for deployment
```

```
Pragma version^0.8.16 (Arcana.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (ArcanaBeacon.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (DID.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (ERC2771ContextUpgradeable.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
 4  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (Factory.sol#2) necessitates a version too recent to be
trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (Forwarder.sol#2) necessitates a version too recent to be
trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (NodeList.sol#2) necessitates a version too recent to be
trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (RoleLib.sol#2) necessitates a version too recent to be
trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (interfaces/IArcana.sol#2) necessitates a version too recent
to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (interfaces/IArcanaDID.sol#2) necessitates a version too
recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (interfaces/IArcanaFactory.sol#2) necessitates a version too
recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (interfaces/IDID.sol#2) necessitates a version too recent to
be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (interfaces/IFactory.sol#2) necessitates a version too recent
to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (interfaces/IFactoryArcana.sol#2) necessitates a version too
recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (interfaces/IFactoryDID.sol#2) necessitates a version too
recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (interfaces/IFactoryForwarder.sol#2) necessitates a version
too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

```
Pragma version^0.8.16 (interfaces/IForwarder.sol#2) necessitates a version too
recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.7
```

```
2  pragma solidity ^0.8.16;
```

Recommendation

We recommend deploying with any of the following Solidity versions:

- 0.5.16 - 0.5.17
- 0.6.11 - 0.6.12
- 0.7.5 - 0.7.6
- 0.8.15 - 0.8.16

Use a simple pragma version that allows any of these versions. Also, consider using the latest version of Solidity for testing.

Alleviation

The team heeded our advice and fix the issue in commit 1cadefaa8662b1daeb78bca48c2068f5b364d97d.

CON-06 | DIFFERENT SOLIDITY VERSIONS

Category	Severity	Location	Status
Language Specific	Informational	Arcana.sol: 2; ArcanaBeacon.sol: 2; DID.sol: 2; ERC2771ContextUpgradeable.sol: 4; Factory.sol: 2; Forwarder.sol: 2; NodeList.sol: 2; RoleLib.sol: 2; interfaces/IArcana.sol: 2; interfaces/IArcanaDID.sol: 2; interfaces/IArcanaFactory.sol: 2; interfaces/IDID.sol: 2; interfaces/IFactory.sol: 2; interfaces/IFactoryArcana.sol: 2; interfaces/IFactoryDID.sol: 2; interfaces/IFactoryForwarder.sol: 2; interfaces/IForwarder.sol: 2; test/ArcanaV2.sol: 3; test/FactoryV2.sol: 3; test/ForwarderV2.sol: 3	Resolved

Description

Multiple Solidity versions are used in the codebase. Using a floating pragma is not recommended.

Versions used: `^0.8.2`, `^0.8.16`, `^0.8.0`, `^0.8.1`

`^0.8.2` is used in node_modules/@openzeppelin/contracts/proxy/ERC1967/ERC1967Upgrade.sol file.

```
4  pragma solidity ^0.8.2;
```

`^0.8.16` is used in contracts/DID.sol file.

```
2  pragma solidity ^0.8.16;
```

`^0.8.0` is used in node_modules/@openzeppelin/contracts/access/IAccessControl.sol file.

```
4  pragma solidity ^0.8.0;
```

`^0.8.1` is used in node_modules/@openzeppelin/contracts/utils/Address.sol file.

```
4  pragma solidity ^0.8.1;
```

Recommendation

We recommend using one Solidity version.

Alleviation

The team heeded our advice and fix the issue in commit f66cba11e4fa62c637e0e6799327a8818367cb5e.

OPTIMIZATIONS | ARCANA NETWORK

ID	Title	Category	Severity	Status
CON-07	Function Should Be Declared External	Gas Optimization	Optimization	Resolved

CON-07 | FUNCTION SHOULD BE DECLARED EXTERNAL

Category	Severity	Location	Status
Gas Optimization	Optimization	Arcana.sol: 112, 330, 340; ArcanaBeacon.sol: 30; DID.sol: 40, 81, 103; Factory.sol: 49, 60, 142; Forwarder.sol: 91, 114, 122, 133, 179; NodeList.sol: 55, 129, 137, 145, 171; test/ArcanaV2.sol: 9	Resolved

Description

The functions which are never called internally within the contract should have external visibility for gas optimization.

Recommendation

We advise to change the visibility of the aforementioned functions to `external`.

Alleviation

The team heeded our advice and resolved this issue in commit `a84d9fea9a626806b641f4c6139b0aedf471408c`.

APPENDIX | ARCANA NETWORK

Finding Categories

Categories	Description
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.
Gas Optimization	Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.
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.
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 | CERTIK

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

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

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

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

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

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

FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

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

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

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

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

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

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

CertiK | Securing the Web3 World

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.