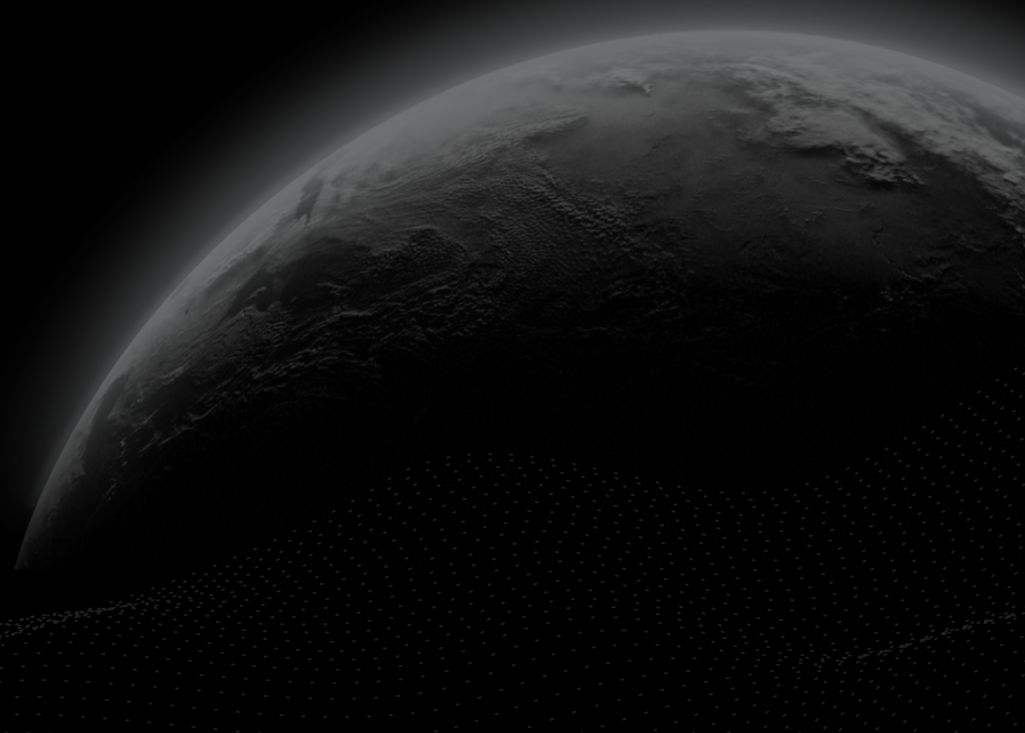
CERTIK

Security Assessment

Cyberconnect

CertiK Verified on Sept 9th, 2022



CertiK Verified on Sept 9th, 2022

Cyberconnect

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

Executive Summary

TYPES	ECOSYSTEM	METHODS
DeFi	Ethereum	Manual Review, Static Analysis

LANGUAGE	TIMELINE	KEY COMPONENTS
Solidity	Delivered on 09/09/2022	N/A

CODE BASE

github.com/cyberconnecthq/cybercontracts

COMMITS

base: 94b89c97b2be41f5534e8ee15d0b5f74ecdd72c6

update: 3f293faaa990ea06624ce7faff6446d21d7e7468

Vulnerability Summary

18	15	2	0	1	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 Mitigated	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.
2	Medium	2 Resolved	Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
5	Minor	5 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.
9	Informational	8 Resolved, 1 Acknowledged	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 | CYBERCONNECT

CODE BASE | CYBERCONNECT

Repository

https://github.com/cyberconnecthq/cybercontracts

Commit

base: 94b89c97b2be41f5534e8ee15d0b5f74ecdd72c6
update: 3f293faaa990ea06624ce7faff6446d21d7e7468

AUDIT SCOPE | CYBERCONNECT

56 files audited ● 2 files with Acknowledged findings ● 9 files with Mitigated findings ● 11 files with Resolved findings ● 34 files without findings

ID	Repo	Commit	File	SHA256 Checksum
CNF	cyberconnecthq/cybercontracts	94b89c9	src/base/CyberNFTBase.sol	d3f95695bdbbf38b9c6a85eb6884ee57ec5aa9ce12f92fa28233d2bebb0fe988
PNF	cyberconnecthq/cybercontracts	94b89c9	src/core/ProfileNFT.sol	59212aabd2774122812b6d7ac7a94a56d7e74297cd8457b2be4831467d3e63a0
CEB	cyberconnecthq/cybercontracts	94b89c9	src/core/CyberEngine.sol	a819c65118ba928d832b62dd20a374d19a4ad88f8955dc2031bb583d50e632d6
ENF	cyberconnecthq/cybercontracts	94b89c9	src/core/EssenceNFT.sol	19a508c9cec7cc128e047d15c09293fab1ba99f9523eda032149e795f2b5087c
SNF	cyberconnecthq/cybercontracts	94b89c9	src/core/SubscribeNFT.sol	12b5855023051fdbf3f7e67fa28d14c9ba98d79fdf0780dec7e4e273550bb1f8
AUT	cyberconnecthq/cybercontracts	94b89c9	src/dependencies/solmate/Auth.sol	4ae02f9e7627160b012c6066061c745ee8de48b17cdf4119ad3cbbcb339fbf76
OWN	cyberconnecthq/cybercontracts	94b89c9	src/dependencies/solmate/Owned.sol	40bbf37846c9b71b311ee8d2227a0399942bdc25ef3ebfa2b0a280dd5dc8425c
TRE	cyberconnecthq/cybercontracts	94b89c9	src/middlewares/base/Treasury.sol	6928c62a318dbab84d8698214dd562a63044bf90bd41cec074d569844fcc2831
PFC	cyberconnecthq/cybercontracts	94b89c9	src/middlewares/profile/PermissionedFeeCreationMw.sol	1e56b7274b8f6bba9274fa88bf240b5cc5aa13b884790a8b2dc37f01d49ab73c
CBN	cyberconnecthq/cybercontracts	94b89c9	src/periphery/CyberBoxNFT.sol	d1bd8bfe8e1eeb3f8971c7f06c7841bcb907a66bfb47d7d3ac86f4f61c6cf6c5

ID	Repo	Commit	File	SHA256 Checksum
LPD	cyberconnecthq/cybercontracts	94b89c9	src/periphery/Link3ProfileDescriptor.sol	f01963914413e7c74ebdcd406335813ed5e74a9752c1bcd9b129eb696a3d38a1
EIP	cyberconnecthq/cybercontracts	94b89c9	src/base/EIP712.sol	f5653068875689fb84d95027e225dc1bdfb35a16a7939c26a79ab5bc0a96ef37
RGB	cyberconnecthq/cybercontracts	94b89c9	src/dependencies/openzeppelin/ReentrancyGuard.sol	d3af5f291d92d03347599325254c16bb5038e9603a70fcf2189a1578b0e6ddd0
ERC	cyberconnecthq/cybercontracts	94b89c9	src/dependencies/solmate/ERC721.sol	08e6c8bbe793042eb6e5b4469f3fc61d1af1d851c7cca38e9f60d163d02290ee
IEM	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/IEssenceMiddleware.sol	764e9ee702a2fefd9e61fe8f30b68e717c9a067280b50a9858f996094bf7de03
IPT	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/IProfileNFTEvents.sol	c0c00a0e0337021d3955627880a75084f0558995eaceadbef47c6a4c2b328f2e
ISM	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/ISubscribeMiddleware.sol	66fe322d4f12bffe9115c960829d1f4edc5c10ebb64728c2dd5e1693aa5d810b
ACT	cyberconnecthq/cybercontracts	94b89c9	src/libraries/Actions.sol	67aece3c2f9bf9e948756dbfa64eb1b9a7cde72a38d6d194f379174b3e605672
CON	cyberconnecthq/cybercontracts	94b89c9	src/libraries/Constants.sol	26f288be95518ce41e50478720e7acc8eb7699815b2aa95b56f632118695644a
QRS	cyberconnecthq/cybercontracts	94b89c9	src/libraries/QRSVG.sol	6affa3ee7931c39b400ca2f4a568813a53e3aa2d3be79db309554d2906a48b69
SOO	cyberconnecthq/cybercontracts	94b89c9	src/middlewares/subscribe/SubscribeOnlyOnceMw.sol	133647109af2d4f5c70dc14a07e6b1e508c8f00430369c38a8e2456080d166cb
UBB	cyberconnecthq/cybercontracts	94b89c9	src/upgradeability/UpgradeableBeacon.sol	c319c0a30d8f9bf02eb67a79bb6e57611c3c614c70ba2dd42a2e4954f8506d4d

ID	Repo	Commit	File	SHA256 Checksum
PAU	cyberconnecthq/cybercontracts	94b89c9	src/dependencies/openzeppelin/Pausable.sol	bcd6b6bda6671e082422abf3a0df875be48189fc41d73611c007d85b1f128dd5
RAB	cyberconnecthq/cybercontracts	94b89c9	src/dependencies/solmate/RolesAuthority.sol	6890a193c4d373b8a5a66c5e700a875031716580b1ef63a130aa2787c2ddf320
EDB	cyberconnecthq/cybercontracts	94b89c9	src/deployer/EssenceDeployer.sol	6c1878040211322e1acb2ce255a4c8793c79580273f2efa44534d31fb0c85415
PDB	cyberconnecthq/cybercontracts	94b89c9	src/deployer/ProfileDeployer.sol	12b937fe26d8680db5b0b6d18738c3e69f8fe19905c8bb999ade4f95bc3a72ed
SDB	cyberconnecthq/cybercontracts	94b89c9	src/deployer/SubscribeDeployer.sol	bd7b42a07273858e3ec8ba29660876bfd3b6f2218e62345abc1c0731ecb87aaf
ICB	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/ICyberBox.sol	3d8bae6d2cb481f2ee75e181d842ae2e9dcdb98be111761d99e872b2b21d3da7
ICE	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/ICyberBoxEvents.sol	4b316ddcc800ee0048fd6f74dd23a7f041e4a02aa747c0e978cb6ec45e1d39bc
ICY	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/ICyberEngine.sol	fdc4658e09ef40f765054f45b4494c9baca2402ee84bebd840d4770d9d6047c2
IEE	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/ICyberEngineEvents.sol	ae0a4ee92eb3506abeb5c5856196bad806dcfdfb6195f1040c3794e957e61767
ICN	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/ICyberNFTBase.sol	cc80024496d0eaae4d6b4422251baeb046d210f283cdbdf52d53eb4dfe1fb23b
IED	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/IEssenceDeployer.sol	c3ce79902c47bbc392e5ff8dbce6dc2139be0d505596736e03de55f1afda97f9
IEN	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/IEssenceNFT.sol	3bdd63ee1edd39a44a40fc0a76121cb9c52694b2b009d28662d4b9a59bb15878

ID	Repo	Commit	File	SHA256 Checksum
IEF	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/IEssenceNFTEvents.sol	09af472ce6b325d0382499838939398e80ea03318c4dba393a0a8735694d365251
IPD	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/IProfileDeployer.sol	540715e490bc51ec2bbd625dfc1cdc0bceb8350aeb427b078888111ad6189125
IPM	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/IProfileMiddleware.sol	252fcdd786dce129cef89e46614d293ddbdcbdccbfa52629cf274edaa60d4917
IPN	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/IProfileNFT.sol	6aa6aa790e254a1516919a10913426b83bcb69262bed255181a461b0ef36a361
IPF	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/IProfileNFTDescriptor.sol	9423b35b713e3f0194830b0effd70c1e5152f2bb073517f26262ff81aa7afd57
ISD	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/ISubscribeDeployer.sol	b26c2a9b6e2f8116f127764365187043a771a918c64574766c18c3999ca34326
ISN	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/ISubscribeNFT.sol	b62e0a176a295dc43fd38fd7b92cb1435b46174ecf6f172631fb091012b4b180
ISF	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/ISubscribeNFTEvents.sol	7ad45a0986f72f74612e52672eef0b547c1f02f743b9175ce3e2234500993236
ITB	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/ITreasury.sol	a6663c5e0baa36ec06e93a8c6a784330acee9e3d0fe506e3749308c5a9198ad8
IUB	cyberconnecthq/cybercontracts	94b89c9	src/interfaces/IUpgradeable.sol	d43b458bba02391027cd7fbd78c4097013b8ac3fc826289af82b256404ebd9f0
DTB	cyberconnecthq/cybercontracts	94b89c9	src/libraries/DataTypes.sol	75726624493bf517154ce537766e6278a336bf2afae08408c60841c935a09eb9
LSB	cyberconnecthq/cybercontracts	94b89c9	src/libraries/LibString.sol	5a78b310e0d0fccc42b8900a536688661458cb2faec763b89d0961bc2de90f3c
FMB	cyberconnecthq/cybercontracts	94b89c9	src/middlewares/base/FeeMw.sol	02d33ee368dd48da9203c9e7d3f749dbfcd29d4df77f572ec12988de6a3b61d2

ID	Repo	Commit	File	SHA256 Checksum
PMB	cyberconnecthq/cybercontracts	94b89c9	src/middlewares/base/PermissionedMw.sol	53d75d08e397bb83ad25e887ae735878bff71700ee88e4d2ace0f364d91322d9
COS	cyberconnecthq/cybercontracts	94b89c9	src/middlewares/essence/CollectOnlySubscribedMw.sol	ad74cdb3409e8946a1780785293beaaa26c5a725f9fe73e7969598ea7e68d693
CBF	cyberconnecthq/cybercontracts	94b89c9	src/storages/CyberBoxNFTStorage.sol	2a23c961c478b18c38340ebde5da35eb71d028a3449d9c63c76fb6e0fe777955
CES	cyberconnecthq/cybercontracts	94b89c9	src/storages/CyberEngineStorage.sol	e89df2863921edbc87bbfcd3e6b798925b5e41de94dcc3ddb7a001459e8b809c
ENT	cyberconnecthq/cybercontracts	94b89c9	src/storages/EssenceNFTStorage.sol	075a1430c493c2ece01fe3d87ea858cf9e842ff5066ba3ab36eb52366899df07
LPS	cyberconnecthq/cybercontracts	94b89c9	src/storages/Link3ProfileDescriptorStorage.sol	66717d5b74ccc56c75f718d0b22988f54502e908f521fcc02167b3560288717d
PNT	cyberconnecthq/cybercontracts	94b89c9	src/storages/ProfileNFTStorage.sol	7dabd197af5a1aad0d20d45141afb49cf59975a3e7e8ad0678b936f333e54f2d
SNT	cyberconnecthq/cybercontracts	94b89c9	src/storages/SubscribeNFTStorage.sol	51759da61c5eec1f4fb0b29f469afe95517dfd86977ec226c739008af647e61a
INI	cyberconnecthq/cybercontracts	94b89c9	src/upgradeability/Initializable.sol	847f468e04fb9b043ec79f2df75cbaccc9d639b53759314447354ecd35c38666

APPROACH & METHODS | CYBERCONNECT

This report has been prepared for Cyberconnect to discover issues and vulnerabilities in the source code of the Cyberconnect 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 | CYBERCONNECT

18	0	2	2	5	9
Total Findings	Critical	Major	Medium	Minor	Informational

This report has been prepared to discover issues and vulnerabilities for Cyberconnect. Through this audit, we have uncovered 18 issues ranging from different severity levels. Utilizing Static Analysis techniques to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
SRC-01	**Centralized Control Of Contract Upgrade**	**Centralization / Privilege**	**Major**	**Mitigated**
SRC-02	**Centralization Risks**	**Centralization / Privilege**	**Major**	**Mitigated**
SRC-03	Potential Reentrancy Attack	Volatile Code	Medium	Resolved
TRE-01	Wrong Variable In Check	Logical Issue	Medium	Resolved
ACT-01	Usage Of Non-Allowed Middlewares	Logical Issue	Minor	Resolved
PFC-01	Usage Of `transfer()` For Sending Ether	Volatile Code	Minor	Resolved
PNF-01	Profile Upgradability	Logical Issue	Minor	Resolved
SRC-04	Missing Zero Address Validation	Volatile Code	Minor	Resolved
SRC-05	Missing Checks Or Input Validations	Volatile Code	Minor	Resolved
ACT-02	Lack Of Input Validation	Logical Issue	Informational	Resolved
AUT-01	Missing Error Messages	Coding Style	Informational	Resolved

ID	Title	Category	Severity	Status
CBN-01	Token With Empty URI	Logical Issue	Informational	Resolved
DEP-01	Lack Of `onlyInitializing` Modifier	Coding Style	Informational	Resolved
EIP-01	Missing Check For `v` And `s`	Logical Issue	Informational	Resolved
PNF-02	Usage Of `_currentIndex`	Coding Style	Informational	Resolved
SRC-06	Missing Emit Events	Coding Style	Informational	Resolved
SRC-07	Typos	Coding Style	Informational	Resolved
SRC-08	Multiple Functions Use Same Nonce	Logical Issue	Informational	Acknowledged

SRC-01 | FINDING DETAILS

Finding Title

Centralized Control Of Contract Upgrade

Category	Severity	Location	Status
Centralization / Privilege	**Major**	**src/core/CyberEngine.sol (base): 31; src/core/EssenceNFT.sol (base): 18; src/core/ProfileNFT.sol (base): 28; src/core/SubscribeNFT.sol (base): 21; src/periphery/CyberBoxNFT.sol (base): 23; src/periphery/Link3ProfileDescriptor.sol (base): 24**	**Mitigated**

Description

`CyberEngine.sol`, `EssenceNFT.sol`, `ProfileNFT.sol`, `SubscribeNFT.sol`, `CyberBoxNFT.sol` and `Link3ProfileDescriptor.sol` are upgradeable contracts, authorized accounts can upgrade these contracts without the community's commitment. If an attacker compromises the account, they 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

`[CertiK]`: The client implemented a timelock and multisig:

- Multisig Wallet (Timelock owner): 0x9aEd1dA7127bF39838f6a1F407563437b362C64f;
- Timelock Contract: 0xd861Ea72fAFB554d531bA8077CB0d8a42C78f4bF and transferred the ownership to the Multisig Wallet with this transaction 0x2834a1c5de9b00b68cdc42235cf11fc59102490fd17f075d62681f46610a262e.

The client transferred ownership of the following contracts to the Timelock Contract:

- CyberEngine Auth 0x5cf03F4997AFa9A94506990D24c12D6aBaD61E6F with this transaction 0x91b044a5dd36e37d069687635a0c7754e21abe3ba3a21bf004932400fbf8714b;
- CyberBoxNFT 0xcE4F341622340d56E397740d325Fd357E62b91CB with this transaction 0x100d070342456e7b3e8e167dceeda39d2156f3d0aeb82a8beab9b9fa93177b23;
- Link3ProfileDescriptor 0x818CBEE6081ae4C89caBc642Ac2542b2585F68Bb with this transaction 0x4f63967a268cef742f2c9abcbbd7259d9a1e63384cf752679902ba029eafa50e;
- CyberTreasury 0x5DA0eD64A9868d128F8d6f56dC78B727F85ff2D0 with this transaction 0x8e18f4dc6afcc8f81ea4fe098c6cdf2f27118b83ec04f7bd9b75a0773cae174a.

SRC-02 | FINDING DETAILS

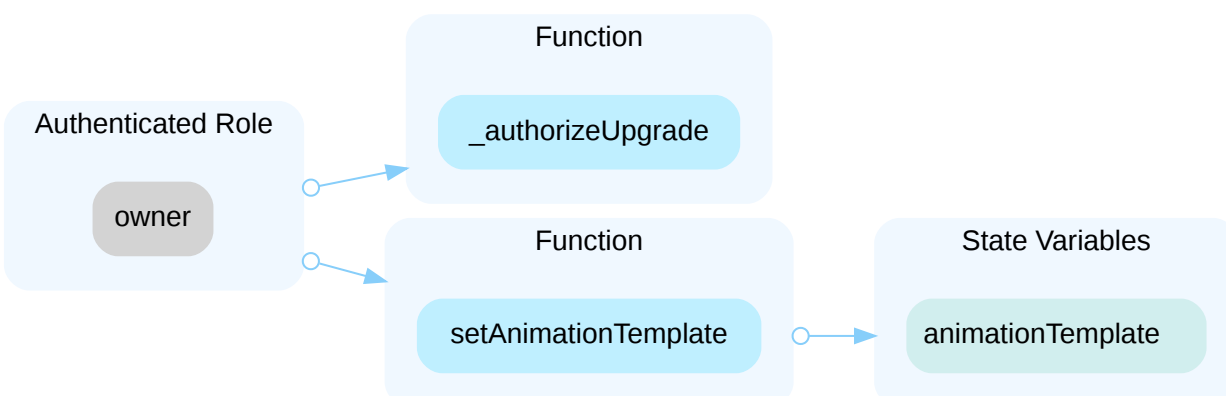
Finding Title

Centralization Risks

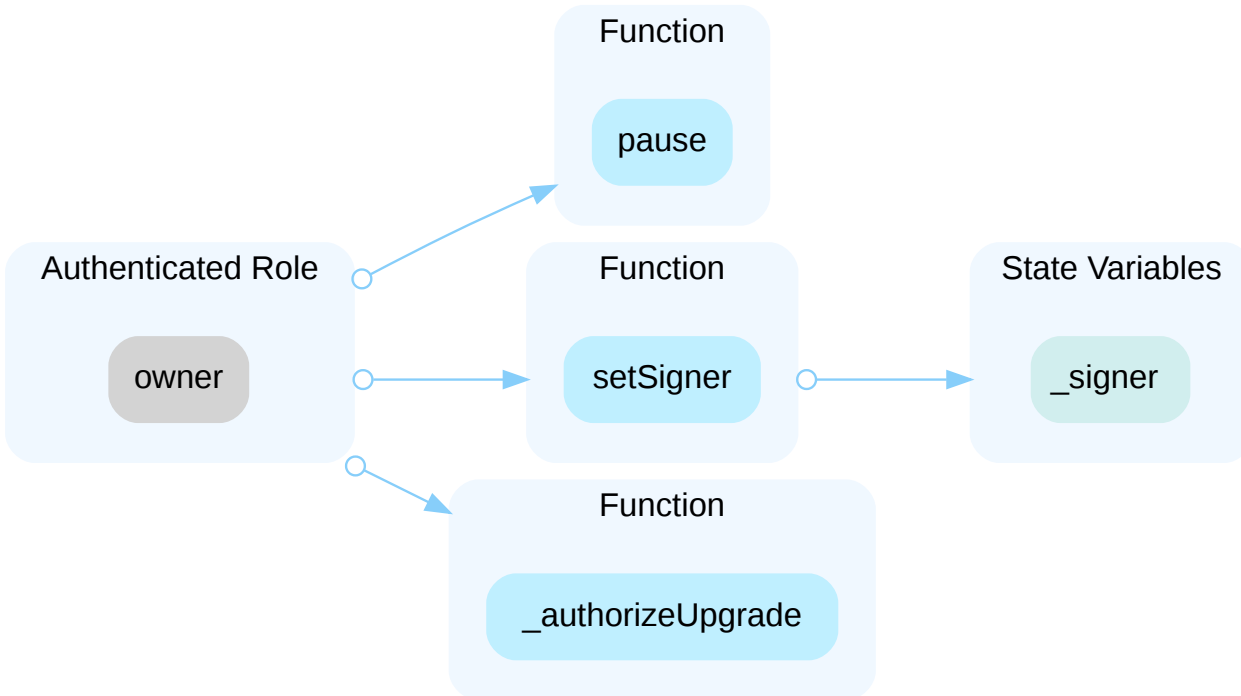
Category	Severity	Location	Status
Centralization / Privilege	**Major**	**src/base/CyberNFTBase.sol (base): 92; src/core/CyberEngine.sol (base): 219; src/core/EssenceNFT.sol (base): 61; src/core/ProfileNFT.sol (base): 250, 292, 301, 313, 358, 428, 467, 733; src/dependencies/solmate/Auth.sol (base): 41; src/dependencies/solmate/Owned.sol (base): 43; src/middlewares/base/Treasury.sol (base): 46, 56; src/middlewares/profile/PermissionedFeeCreationMw.sol (base): 108; src/periphery/CyberBoxNFT.sol (base): 67, 124, 191; src/periphery/Link3ProfileDescriptor.sol (base): 58, 294**	**Mitigated**

Description

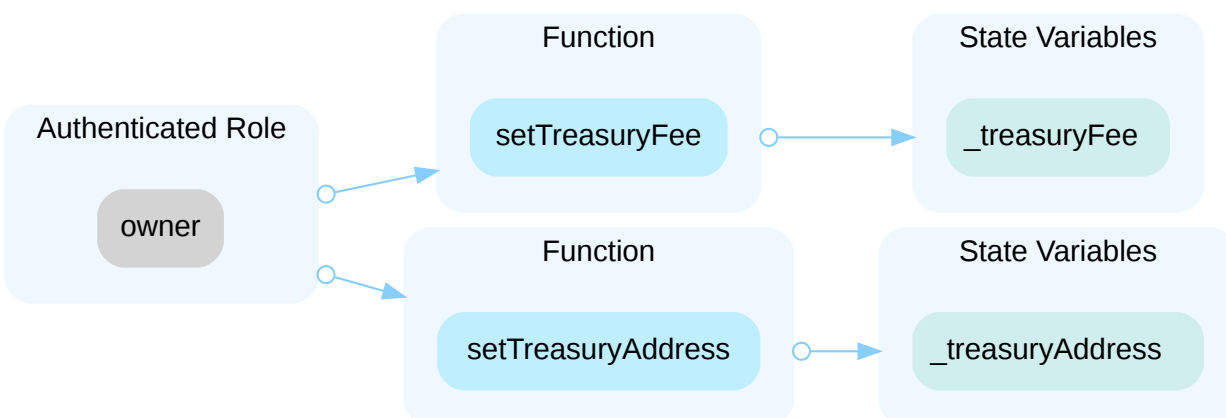
In the contract `Link3ProfileDescriptor` the role `owner` has authority over the functions shown in the diagram below. Any compromise to the `owner` account may allow the hacker to take advantage of this authority and set new animation templates.



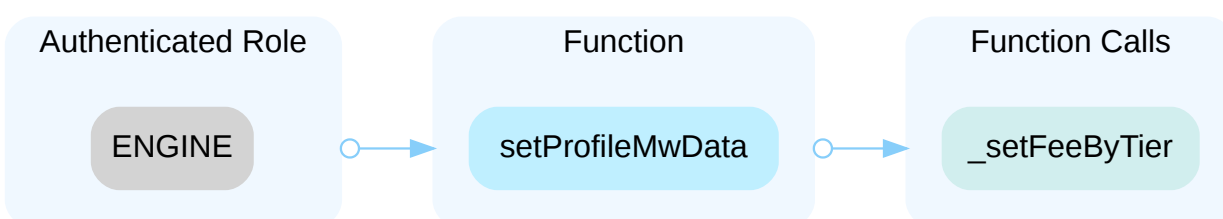
In the contract `CyberBoxNFT` the role `owner` has authority over the functions shown in the diagram below. Any compromise to the `owner` account may allow the hacker to take advantage of this authority and pause/unpause the contract and set a new signer. The new signer can then call `claimBox()` and mint as many boxes as they want.



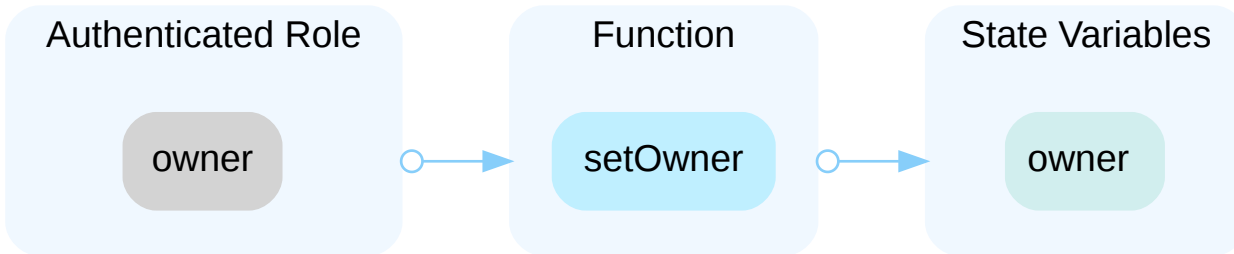
In the contract `Treasury` the role `owner` has authority over the functions shown in the diagram below. Any compromise to the `owner` account may allow the hacker to take advantage of this authority and change the treasuryAddress to one they control or alter the value set for the treasury fee.



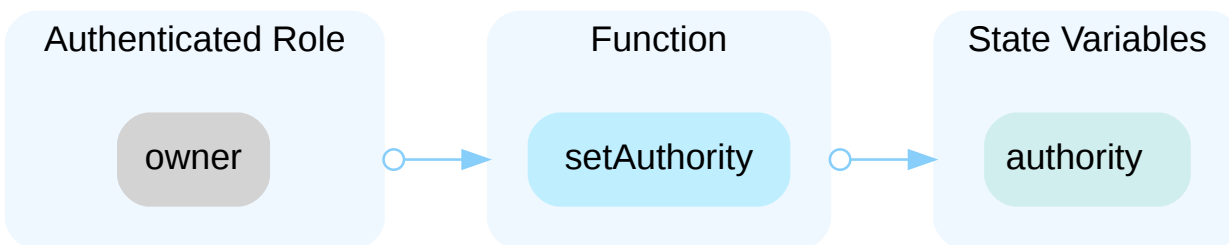
In the contract `PermissionedFeeCreationMw` the role `ENGINE` has authority over the functions shown in the diagram below. Any compromise to the `ENGINE` account may allow the hacker to take advantage of this authority and change the recipient address, the signer address, and all the values for the different fee tiers for a given namespace.



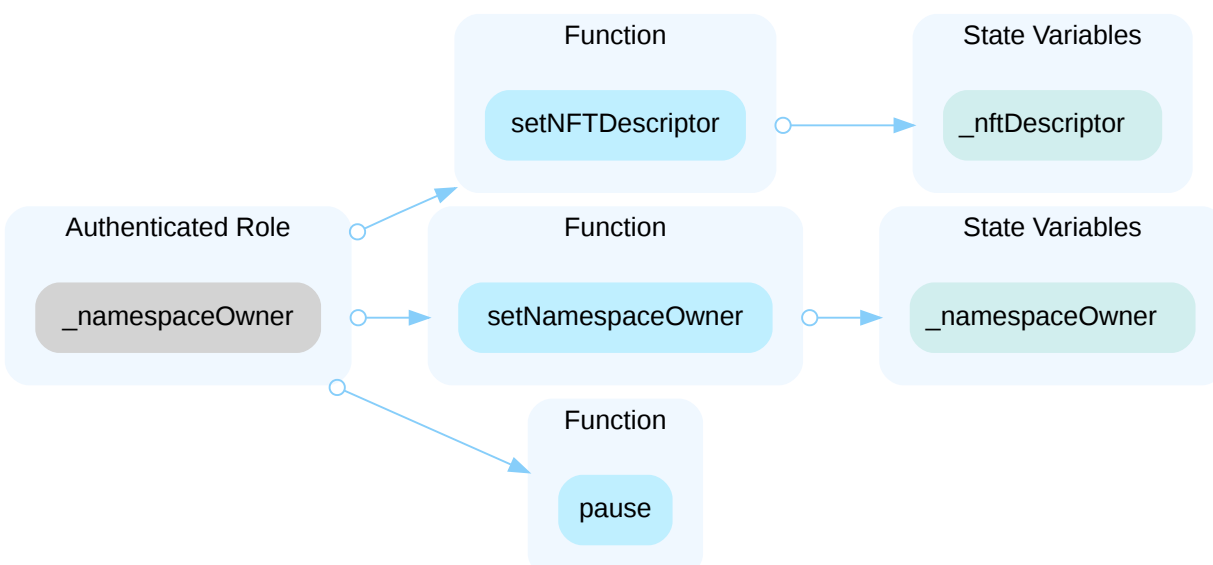
In the contract `Owned` the role `owner` has authority over the functions shown in the diagram below. Any compromise to the `owner` account may allow the hacker to take advantage of this authority and set new malicious address as the owner of the contract.



In the contract `Auth` 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 and change the `Authority`.



In the contract `ProfileNFT` the role `_namespaceOwner` has authority over the functions shown in the diagram below. Any compromise to the `_namespaceOwner` account may allow the hacker to take advantage of this authority and change the `NFTDescriptor` for the current profile, change the namespace owner, and pause/unpause the state of the contract.



In the contract `ProfileNFT` the modifier `onlyProfileOwner()` gives authority over the function setOperatorApproval(). Any compromise to the profile owner account may allow the hacker to take advantage of this authority and set malicious address as operators for a certain profile.

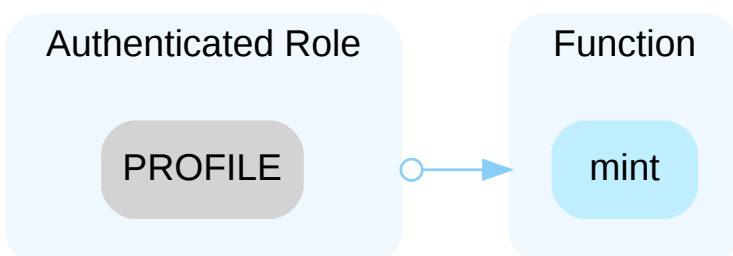
In the contract `ProfileNFT` the modifier `onlyProfileOwnerOrOperator()` gives authority on different functions including:

- `registerEssence()`

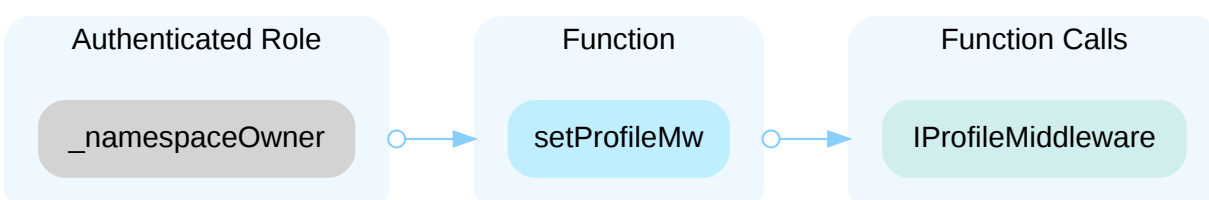
- `setSubscribeData()`
- `setEssenceData()`

Any compromise to the profile owner or operator account may allow the hacker to take advantage of this authority and register new essences, change the essence/subscribe middleware, and change essences' token URIs.

In the contract `EssenceNFT` the role `PROFILE` has authority over the functions shown in the diagram below. Any compromise to the `PROFILE` account may allow the hacker to take advantage of this authority and mint new essences for a given profile.



In the contract `CyberEngine` the role `_namespaceOwner` has authority over the functions shown in the diagram below. Any compromise to the `_namespaceOwner` account may allow the hacker to take advantage of this authority and change the profile middleware for a given namespace to a malicious one.



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

`[CertiK]`: The client implemented a timelock and multisig:

- Multisig Wallet (Timelock owner): 0x9aEd1dA7127bF39838f6a1F407563437b362C64f;
- Timelock Contract: 0xd861Ea72fAFB554d531bA8077CB0d8a42C78f4bF and transferred the ownership to the Multisig Wallet with this transaction 0x2834a1c5de9b00b68cdc42235cf11fc59102490fd17f075d62681f46610a262e.

The client transferred ownership of the following contracts to the Timelock Contract:

- CyberEngine Auth 0x5cf03F4997AFa9A94506990D24c12D6aBaD61E6F with this transaction 0x91b044a5dd36e37d069687635a0c7754e21abe3ba3a21bf004932400fbf8714b;

- CyberBoxNFT 0xcE4F341622340d56E397740d325Fd357E62b91CB with this transaction 0x100d070342456e7b3e8e167dceeda39d2156f3d0aeb82a8beab9b9fa93177b23;
- Link3ProfileDescriptor 0x818CBEE6081ae4C89caBc642Ac2542b2585F68Bb with this transaction 0x4f63967a268cef742f2c9abcbbd7259d9a1e63384cf752679902ba029eafa50e;
- CyberTreasury 0x5DA0eD64A9868d128F8d6f56dC78B727F85ff2D0 with this transaction 0x8e18f4dc6afcc8f81ea4fe098c6cdf2f27118b83ec04f7bd9b75a0773cae174a.

SRC-03 | FINDING DETAILS

Finding Title

Potential Reentrancy Attack

Category	Severity	Location	Status
Volatile Code	Medium	src/core/ProfileNFT.sol (base): 158, 167, 212, 221; src/libraries/Actions.sol (base): 71, 127	Resolved

Description

A reentrancy attack can occur when the contract creates a function that makes an external call to another untrusted contract before resolving any effects. If an attacker controls the untrusted contract, they can make a recursive call back to the original function, repeating interactions that would have otherwise not run after the external call resolved the effects.

In this case the functions `subscribe()`, `subscribeWithSig()`, `collect()`, and `collectWithSig()` can all be reentered as they use the `_safeMint()` function which calls `onERC721Received()` if the address `to` has any code (in which case it is a contract). This checks the contract intends to be able to receive `ERC721` tokens, but this function can also be coded to make a recursive call.

Currently, the only issue this poses is that events can be emitted out of order. However, if middleware is introduced that uses a post-process, then the post-processes will be called out of order and may pose a security risk.

Recommendation

We recommend applying the OpenZeppelin ReentrancyGuard library - `nonReentrant` modifier for the `subscribe()`, `subscribeWithSig()`, `collect()`, and `collectWithSig()` functions to prevent any possible reentrancy attack.

Alleviation

`[CertiK]`: The client made the recommended changes.

TRE-01 | FINDING DETAILS

Finding Title

Wrong Variable In Check

Category	Severity	Location	Status
Logical Issue	Medium	src/middlewares/base/Treasury.sol (base): 57	Resolved

Description

It the function `setTreasuryFee()` in `Treasury.sol`, it is checked that `_treasuryFee <= Constants._MAX_BPS`. This checks that the current treasury fee, not the input treasury fee, is valid.

Recommendation

We recommend replacing `_treasuryFee` with `treasuryFee` in the require statement.

Alleviation

`[CertiK]`: The client made the recommended changes.

ACT-01 | FINDING DETAILS

Finding Title

Usage Of Non-Allowed Middlewares

Category	Severity	Location	Status
Logical Issue	Minor	src/libraries/Actions.sol (base): 228, 251, 277	Resolved

Description

If a user sets a middleware and that middleware is later disabled with `allowEssenceMw()`, `allowSubscribeMw()`, or `allowProfileMw()` it will only disable users who have not already implemented it.

For example:

- `exampleMw` is allowed in `CyberEngine.allowExampleMw()`.
- `exampleMw` is set through `setSubscribeData()`, `setEssenceData()`, or `setProfileMw()`.
- For any reason `exampleMw` is then disabled using `CyberEngine.allowExampleMw()`.
- `exampleMw` will still be used because the mapping storing the middleware is not updated and it is only checked that the middleware is allowed when setting the middleware.

Recommendation

We recommend ensuring middleware that is not allowed cannot be used.

Alleviation

`[CertiK]`: The client made the recommended changes.

PFC-01 | FINDING DETAILS

Finding Title

Usage Of `transfer()` For Sending Ether

Category	Severity	Location	Status
Volatile Code	Minor	src/middlewares/profile/PermissionedFeeCreationMw.sol (base): 93, 95	Resolved

Description

It is not recommended to use Solidity's `transfer()` and `send()` functions for transferring Ether, since some contracts may not be able to receive the funds. Those functions forward only a fixed amount of gas (2300 specifically) and the receiving contracts may run out of gas before finishing the transfer. Also, EVM instructions' gas costs may increase in the future. Thus, some contracts that can receive now may stop working in the future due to the gas limitation.

```
93          payable(mwData.recipient).transfer(actualCollected);
94          if (treasuryCollected > 0) {
95              payable(_treasuryAddress()).transfer(treasuryCollected);
96          }
```

Recommendation

We recommend that the linked `.transfer()` calls are substituted with the utilization of the `sendValue()` function from OpenZeppelin's `Address.sol` either by directly importing the library or copying the linked code.

Alleviation

`[CertiK]`: The client made the recommended changes.

PNF-01 | FINDING DETAILS

Finding Title

Profile Upgradability

Category	Severity	Location	Status
Logical Issue	Minor	src/core/ProfileNFT.sol (base): 733	Resolved

Description

In `ProfileNFT.sol`, the function `_authorizeUpgrade()` has the `onlyEngine` modifier so that only the `ENGINE` can upgrade the implementation contract. However, there is no functionality in `CyberEngine.sol` that would enable it to be upgraded.

Recommendation

We recommend adding functionality to `CyberEngine.sol` to enable the Profile proxy to be upgraded to a new implementation.

Alleviation

`[CertiK]`: The client made the recommended changes.

SRC-04 | FINDING DETAILS

Finding Title

Missing Zero Address Validation

Category	Severity	Location	Status
Volatile Code	Minor	src/dependencies/solmate/Owned.sol (base): 43; src/middlewares/base/Treasury.sol (base): 32, 47; src/upgradeability/UpgradeableBeacon.sol (base): 32	Resolved

Description

The following addresses should be checked that they are not the zero address before assignment or external call:

- In `Owned.sol`, `newOwner` in `setOwner()` function.
- In `Treasury.sol`, `treasuryAddress` in the `constructor()`.
- In `Treasury.sol`, `treasuryAddress` in `setTreasuryAddress()` function.
- In `UgradeableBeacon.sol`, `owner` in the `constructor()`.

Recommendation

We recommend adding a zero-check for the passed-in address value to prevent unexpected errors.

Alleviation

`[CertiK]`: The client made the recommended changes.

SRC-05 | FINDING DETAILS

Finding Title

Missing Checks Or Input Validations

Category	Severity	Location	Status
Volatile Code	Minor	src/core/CyberEngine.sol (base): 204, 214; src/core/ProfileNFT.sol (base): 555, 575, 585, 617, 627, 637; src/middlewares/base/Treasury.sol (base): 32~33; src/middlewares/profile/PermissionedFeeCreationMw.sol (base): 144~150	Resolved

Description

In `upgradeSubscribeNFT()` and `upgradeEssenceNFT()` of `CyberEngine.sol`, there are no checks that the `namespace` addresses passed as inputs are valid.

In `ProfileNFT.sol` the following functions do not check the `profileId` is minted:

- `getSubscribeMw()`
- `getSubscribeNFT()`
- `getSubscribeNFTTokenURI()`
- `getEssenceNFT()`
- `getEssenceNFTTokenURI()`
- `getEssenceMw()`

Furthermore, the following functions do not check the `essenceId` exists:

- `getEssenceNFT()`
- `getEssenceNFTTokenURI()`
- `getEssenceMw()`

In the `constructor()` of `Treasury.sol`, there is no check that the input `treasuryFee` is less than or equal to `Constants._MAX_BPS`.

Recommendation

We recommend adding checks to the aforementioned variables or functions.

Alleviation

`[CertiK]` : The client made the recommended changes.

ACT-02 | FINDING DETAILS

Finding Title

Lack Of Input Validation

Category	Severity	Location	Status
Logical Issue	Informational	src/libraries/Actions.sol (base): 195	Resolved

Description

The function `registerEssence()` should verify the inputs `data.name`, `data.symbol`, and `data.essenceTokenURI` are not the empty string.

Recommendation

We recommend implementing checks to ensure users cannot register essences with empty strings for the name, symbol, or URI.

Alleviation

`[CertiK]`: The client made the recommended changes.

AUT-01 | FINDING DETAILS

Finding Title

Missing Error Messages

Category	Severity	Location	Status
Coding Style	Informational	src/dependencies/solmate/Auth.sol (base): 44	Resolved

Description

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

Recommendation

We recommend adding an error message to the **require** statement in the `setAuthority()` function in `Auth.sol`.

Alleviation

`[CertiK]`: The client made the recommended changes.

CBN-01 | FINDING DETAILS

Finding Title

Token With Empty URI

Category	Severity	Location	Status
Logical Issue	Informational	src/periphery/CyberBoxNFT.sol (base): 183	Resolved

Description

The function `tokenURI()` returns the metadata JSON object for a given `tokenId`, but returns an empty string.

Recommendation

We recommend returning a non-empty metadata JSON object.

Alleviation

`[CertiK]`: The client made the recommended changes.

DEP-01 | FINDING DETAILS

Finding Title

Lack Of `onlyInitializing` Modifier

Category	Severity	Location	Status
Coding Style	Informational	src/dependencies/openzeppelin/ReentrancyGuard.sol (base): 42; src/dependencies/solmate/Auth.sol (base): 19; src/dependencies/solmate/ERC721.sol (base): 60; src/dependencies/solmate/Owned.sol (base): 32	Resolved

Description

The functions `__ReentrancyGuard_init()`, `__Auth_Init`, `__ERC721_Init`, and `__Owned_Init` should only be called once during contract initialization.

Recommendation

We recommend adding the `onlyInitializing` modifier to these functions.

Alleviation

`[CertiK]`: The client made the recommended changes.

EIP-01 | FINDING DETAILS

Finding Title

Missing Check For `v` And `s`

Category	Severity	Location	Status
Logical Issue	Informational	src/base/EIP712.sol (base): 53	Resolved

Description

In `_requiresExpectedSigner()`, the `ecrecover()` function is being used. EIP-2 still allows signature malleability for `ecrecover()`. Appendix F in the Ethereum Yellow paper, defines the valid range for s in (311): `0 < s < secp256k1n ÷ 2 + 1` and for the recovery identifier (312): `v ∈ {0,1}`. This should not be confused with the input for `ecrecover()` where `v ∈ {27,28}`. (See doc) However, these values can be obtained by taking `27+"recovery identifier"`, so that they will also yield a unique signature and are often the `v` values returned from signatures. (For example `web3.eth.accounts.sign()`)

If your library generates malleable signatures, such as `s`-values in the upper range, calculate a new `s`-value with `0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFEBAAEDCE6AF48A03BBFD25E8CD0364141 - s1` and flip `v` from `27` to `28` or vice versa. If your library also generates signatures with `0/1` for `v` instead `27/28`, add `27` to `v` so that `ecrecover()` accepts these signatures as well.

Recommendation

We recommend adding the following checks or to consider the example in `ECDSA.sol` from the OpenZeppelin library.

```
require(uint256(s) <=
0x7FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF5D576E7357A4501DDFE92F46681B20A0, "ECDSA: invalid
signature 's' value");
require(uint8(v) == 27 || uint8(v) == 28, "ECDSA: invalid signature 'v' value");
```

Alleviation

`[CertiK]`: The client made the recommended changes.

PNF-02 | FINDING DETAILS

Finding Title

Usage Of `_currentIndex`

Category	Severity	Location	Status
Coding Style	Informational	src/core/ProfileNFT.sol (base): 789~803	Resolved

Description

In `_createProfile()`, `_currentIndex` is often used instead of `tokenID` when they are the same value.

Recommendation

We recommend using `tokenId` instead of `_currentIndex` for consistency and to improve readability.

Alleviation

`[CertiK]`: The client made the recommended changes.

SRC-06 | FINDING DETAILS

Finding Title

Missing Emit Events

Category	Severity	Location	Status
Coding Style	Informational	src/interfaces/IProfileNFTEvents.sol (base): 35; src/middlewares/base/Treasury.sol (base): 46, 56; src/middlewares/profile/PermissionedFeeCreationMw.sol (base): 216; src/periphery/Link3ProfileDescriptor.sol (base): 58	Resolved

Description

There should always be events emitted in the sensitive functions that are controlled by centralization roles. Also, in the contract `IProfileNFTEvents.sol`, event `SetAnimationTemplate` has been defined but not emitted in any contracts.

Recommendation

We recommend emitting events for the sensitive functions that are controlled by centralization roles. We also recommend either emitting the event `SetAnimationTemplate` where appropriate or removing it.

Alleviation

`[CertiK]`: The client made the recommended changes.

SRC-07 | FINDING DETAILS

Finding Title

Typos

Category	Severity	Location	Status
Coding Style	Informational	src/base/CyberNFTBase.sol (base): 128; src/base/EIP712.sol (base): 32, 84; src/interfaces/IEssenceMiddleware.sol (base): 20, 20; src/interfaces/ISubscribeMiddleware.sol (base): 27, 42; src/libraries/Constants.sol (base): 6, 38; src/libraries/QRSVG.sol (base): 40; src/middlewares/base/Treasury.sol (base): 11; src/middlewares/profile/PermissionedFeeCreationMw.sol (base): 159, 260; src/middlewares/subscribe/SubscribeOnlyOnceMw.sol (base): 32, 32, 37, 41, 50; src/upgradeability/UpgradeableBeacon.sol (base): 27	Resolved

Description

In `ISubscribeMiddleware.sol` and then also in `SubscribOnlyOnceMw.sol` there are typos in `preProcess()` and `postProcess()` function arguments: `subscrbeNFT` should be `subscribeNFT`.

In `Constants.sol` there is a typo at line 38 in the `_SET_OPERATOR_APPROVAL_TYPEHASH` constant. The string passed to keccak256 is `setOperatorApprovalWithSign(...)`. It should be `setOperatorApprovalWithSig()`.

In `CyberNFTBase.sol`, `EIP712.sol`, and `PermissionedFeeCreationMw.sol` the function named `_domainSeperatorName()` has a typo in it. It should be `_domainSeparatorName()`.

In `Constants.sol`, `CyebreEngine` is written instead of `CyberEngine`.

In `Treasury.sol`, `Treasury` should be written instead of `Treasurt`.

In `SubscribeOnlyOnceMw.sol` and `IEssenceMiddleware.sol`, `Proccess` and `aready` are written instead of `Process` and `already` respectively.

In `IEssenceMiddleware.sol`, `essenceeNFT` is written instead of `essenceNFT`.

In `PermissionedFeeCreationMw.sol`, `PUBLIC VIEW` should be `EXTERNAL VIEW`.

In `UpgradeableBeacon.sol`, the comment for the `constructor()` states the deployer account is the owner. However, the owner will be the address input to the constructor, not necessarily the deployer account.

In `QRSVG.sol`, the function `generateQRCode()` has `emit MatrixCreated(qrMatrix.matrix);` commented-out. This event is not defined and this commented-out code can be deleted.

Recommendation

We recommend fixing the typos.

Alleviation

`[CertiK]`: The client made the recommended changes.

SRC-08 | FINDING DETAILS

Finding Title

Multiple Functions Use Same Nonce

Category	Severity	Location	Status
Logical Issue	Informational	src/base/CyberNFTBase.sol (base): 55; src/core/ProfileNFT.sol (base): 200, 238, 280, 346, 382, 416, 455, 497, 529	Acknowledged

Description

The functions `permit()`, `subscribeWithSig()`, `collectWithSig()`, `registerEssenceWithSig()`, `setAvatarWithSig()`, `setOperatorApprovalWithSig()`, `setMetadataWithSig()`, `setSubscribeDataWithSig()`, `setEssenceDataWithSig()`, and `setPrimaryProfileWithSig()` all use the same `nonces` mapping. It may be possible for a user to provide multiple signatures before a function is executed so that they all use the same nonce. This will only allow one of the functions to be called as all the remaining signatures will become invalid when the nonce is incremented.

Recommendation

We recommend ensuring that a user cannot accidentally provide multiple signatures with the same nonce.

Alleviation

`[CertiK]`: The client acknowledged the finding and opted to not make any changes.

APPENDIX | CYBERCONNECT

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.
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 | 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.