Cyberscope

Audit Report

Disney

June 2023

Network BSC

Address 0x74d11B742ca7be933C175c48F7A409C908a17634

Audited by © cyberscope

Analysis

Critical Medium Minor / Informative Pass

Severity	Code	Description	Status
	ST	Stops Transactions	Unresolved
	OTUT	Transfers User's Tokens	Passed
	ELFM	Exceeds Fees Limit	Passed
	MT	Mints Tokens	Unresolved
	BT	Burns Tokens	Passed
	BC	Blacklists Addresses	Passed

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
	L18	Multiple Pragma Directives	Unresolved
	L19	Stable Compiler Version	Unresolved

Table of Contents

Review

Contract Name	Disney
Compiler Version	v0.8.9+commit.e5eed63a
Optimization	200 runs
Explorer	https://bscscan.com/address/0x74d11b742ca7be933c175c48f7a409c908a17634
Address	0x74d11b742ca7be933c175c48f7a409c908a17634
Network	BSC
Decimals	18

Audit Updates

Initial Audit	02 Jun 2023 https://github.com/cyberscope-io/audits/blob/main/dis/v1/audit.pdf
Corrected Phase 2	08 Jun 2023

Source Files

Filename	SHA256
Disney.sol	55e1b4d35feba3d070347c00a243b6b66a09ecaa78eb07ffbebc960b21b80481

Findings Breakdown

Critical	1
Medium	0
Minor / Informative	3

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	1	0	0	0
Medium	0	0	0	0
Minor / Informative	3	0	0	0

ST - Stops Transactions

Criticality	Minor / Informative
Location	Disney.sol#L4220
Status	Unresolved

Description

The contract owner has the authority to stop the transactions for all users. The owner may take advantage of it by calling the `pause` to method.

```
function _beforeTokenTransfer(
  address from,
  address to,
  uint256 amount
) internal override whenNotPaused {
  super._beforeTokenTransfer(from, to, amount);
}
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. Some suggestions are:

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-sign wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.
- Renouncing the ownership will eliminate the threats but it is non-reversible.

MT - Mints Tokens

Criticality	Critical
Location	Disney.sol#L4200
Status	Unresolved

Description

The contract owner has the authority to mint tokens. The owner may take advantage of it by calling the `mint` function. As a result, the contract tokens will be highly inflated.

```
function mint(address to, uint256 amount) public onlyOwner {
  _mint(to, amount);
}
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. Some suggestions are:

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-sign wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.
- Renouncing the ownership will eliminate the threats but it is non-reversible.

L18 - Multiple Pragma Directives

Criticality	Minor / Informative
Location	Disney.sol#L6,97,126,145,169,1306,1371,1417,1765,1837,2052,2115,2337,2504,2704,2814,2937,2977,3074,3193,3278,3308,3711,3822,4112,4164
Status	Unresolved

Description

If the contract includes multiple conflicting pragma directives, it may produce unexpected errors. To avoid this, it's important to include the correct pragma directive at the top of the contract and to ensure that it is the only pragma directive included in the contract.

```
pragma solidity ^0.8.0;
pragma solidity ^0.8.0;
solidity ^0.8.0;
solidity ^0.8.9;
...
```

Recommendation

It is important to include only one pragma directive at the top of the contract and to ensure that it accurately reflects the version of Solidity that the contract is written in.

By including all required compiler options and flags in a single pragma directive, the potential conflicts could be avoided and ensure that the contract can be compiled correctly.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	Disney.sol#L6,97,126,145,169,1306,1371,1417,1765,1837,2052,2115,2337,2504,2704,2814,2937,2977,3074,3193,3278,3308,3711,3822,4112,4164
Status	Unresolved

Description

The `^` symbol indicates that any version of Solidity that is compatible with the specified version (i.e., any version that is a higher minor or patch version) can be used to compile the contract. The version lock is a mechanism that allows the author to specify a minimum version of the Solidity compiler that must be used to compile the contract code. This is useful because it ensures that the contract will be compiled using a version of the compiler that is known to be compatible with the code.

```
pragma solidity ^0.8.0;
pragma solidity ^0.8.0;
solidity ^0.8.0;
solidity ^0.8.9;
...
```

Recommendation

The team is advised to lock the pragma to ensure the stability of the codebase. The locked pragma version ensures that the contract will not be deployed with an unexpected version. An unexpected version may produce vulnerabilities and undiscovered bugs. The compiler should be configured to the lowest version that provides all the required functionality for the codebase. As a result, the project will be compiled in a well-tested LTS (Long Term Support) environment.

Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
StorageSlotUpgradeable	Library			
	getAddressSlot	Internal		
	getBooleanSlot	Internal		
	getBytes32Slot	Internal		
	getUint256Slot	Internal		
IERC1967Upgradeable	Interface			
IBeaconUpgradeable	Interface			
	implementation	External		-
IERC1822ProxiableUpgradeable	Interface			
	proxiableUUID	External		-
SafeCastUpgradeable	Library			
	toUint248	Internal		
	toUint240	Internal		

	toUint232	Internal		
	toUint224	Internal		
	toUint216	Internal		
	toUint208	Internal		
	toUint200	Internal		
	toUint192	Internal		
	toUint184	Internal		
	toUint176	Internal		
	toUint168	Internal		
	toUint160	Internal		
	toUint152	Internal		
	toUint144	Internal		
	toUint136	Internal		
	toUint128	Internal		
	toUint120	Internal		
	toUint112	Internal		
	toUint104	Internal		
	toUint96	Internal		
	toUint88	Internal		
	toUint80	Internal		
	toUint72	Internal		
	toUint64	Internal		
	toUint56	Internal		

	toUint48	Internal		
	toUint40	Internal		
	toUint32	Internal		
	toUint24	Internal		
	toUint16	Internal		
	toUint8	Internal		
	toUint256	Internal		
	toInt248	Internal		
	toInt240	Internal		
	toInt232	Internal		
	toInt224	Internal		
	toInt216	Internal		
	toInt208	Internal		
	toInt200	Internal		
	toInt192	Internal		
	toInt184	Internal		
	toInt176	Internal		
	toInt168	Internal		
	toInt160	Internal		
	toInt152	Internal		
	toInt144	Internal		
	toInt136	Internal		
	toInt128	Internal		

	toInt120	Internal		
	toInt112	Internal		
	toInt104	Internal		
	toInt96	Internal		
	toInt88	Internal		
	toInt80	Internal		
	toInt72	Internal		
	toInt64	Internal		
	toInt56	Internal		
	toInt48	Internal		
	toInt40	Internal		
	toInt32	Internal		
	toInt24	Internal		
	toInt16	Internal		
	toInt8	Internal		
	toInt256	Internal		
IVotesUpgradeable	Interface			
	getVotes	External		-
	getPastVotes	External		-
	getPastTotalSupply	External		-
	delegates	External		-
	delegate	External	✓	-

	delegateBySig	External	✓	-
CountersUpgradeable	Library			
	current	Internal		
	increment	Internal	✓	
	decrement	Internal	✓	
	reset	Internal	✓	
MathUpgradeable	Library			
	max	Internal		
	min	Internal		
	average	Internal		
	ceilDiv	Internal		
	mulDiv	Internal		
	mulDiv	Internal		
	sqrt	Internal		
	sqrt	Internal		
	log2	Internal		
	log2	Internal		
	log10	Internal		
	log10	Internal		
	log256	Internal		
	log256	Internal		

StringsUpgradeable	Library			
	toString	Internal		
	toHexString	Internal		
	toHexString	Internal		
	toHexString	Internal		
ECDSAUpgradeable	Library			
	_throwError	Private		
	tryRecover	Internal		
	recover	Internal		
	tryRecover	Internal		
	recover	Internal		
	tryRecover	Internal		
	recover	Internal		
	toEthSignedMessageHash	Internal		
	toEthSignedMessageHash	Internal		
	toTypedDataHash	Internal		
IERC20PermitUpgradeable	Interface			
	permit	External	✓	-
	nonces	External		-
	DOMAIN_SEPARATOR	External		-

AddressUpgradeable	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	verifyCallResultFromTarget	Internal		
	verifyCallResult	Internal		
	_revert	Private		
Initializable	Implementation			
	_disableInitializers	Internal	✓	
	_getInitializedVersion	Internal		
	_isInitializing	Internal		
ERC1967UpgradeUpgradeable	Implementation	Initializable, IERC1967Upgradeable		
	__ERC1967Upgrade_init	Internal	✓	onlyInitializing
	__ERC1967Upgrade_init_unchained	Internal	✓	onlyInitializing
	_getImplementation	Internal		

	_setImplementation	Private	✓	
	_upgradeTo	Internal	✓	
	_upgradeToAndCall	Internal	✓	
	_upgradeToAndCallUUPS	Internal	✓	
	_getAdmin	Internal		
	_setAdmin	Private	✓	
	_changeAdmin	Internal	✓	
	_getBeacon	Internal		
	_setBeacon	Private	✓	
	_upgradeBeaconToAndCall	Internal	✓	
	_functionDelegateCall	Private	✓	
UUPSUpgradeable	Implementation	Initializable, IERC1822ProxiableUpgradeable, ERC1967UpgradeUpgradeable		
	__UUPSUpgradeable_init	Internal	✓	onlyInitializing
	__UUPSUpgradeable_init_unchained	Internal	✓	onlyInitializing
	proxiableUUID	External		notDelegated
	upgradeTo	External	✓	onlyProxy
	upgradeToAndCall	External	Payable	onlyProxy
	_authorizeUpgrade	Internal	✓	
EIP712Upgradeable	Implementation	Initializable		

	__EIP712_init	Internal	✓	onlyInitializing
	__EIP712_init_unchained	Internal	✓	onlyInitializing
	_domainSeparatorV4	Internal		
	_buildDomainSeparator	Private		
	_hashTypedDataV4	Internal		
	_EIP712NameHash	Internal		
	_EIP712VersionHash	Internal		
ContextUpgradeable	Implementation	Initializable		
	__Context_init	Internal	✓	onlyInitializing
	__Context_init_unchained	Internal	✓	onlyInitializing
	_msgSender	Internal		
	_msgData	Internal		
OwnableUpgradeable	Implementation	Initializable, ContextUpgradeable		
	__Ownable_init	Internal	✓	onlyInitializing
	__Ownable_init_unchained	Internal	✓	onlyInitializing
	owner	Public		-
	_checkOwner	Internal		
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	

PausableUpgradeable	Implementation	Initializable, ContextUpgradeable		
	__Pausable_init	Internal	✓	onlyInitializing
	__Pausable_init_unchained	Internal	✓	onlyInitializing
	paused	Public		-
	_requireNotPaused	Internal		
	_requirePaused	Internal		
	_pause	Internal	✓	whenNotPaused
	_unpause	Internal	✓	whenPaused
IERC20Upgradeable	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
IERC20MetadataUpgradeable	Interface	IERC20Upgradeable		
	name	External		-
	symbol	External		-
	decimals	External		-

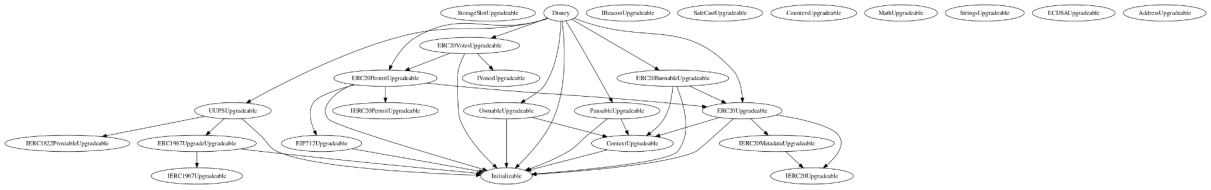
ERC20Upgradeable	Implementation	Initializable, ContextUpgradeable, IERC20Upgradeable, IERC20MetadataUpgradeable		
	__ERC20_init	Internal	✓	onlyInitializing
	__ERC20_init_unchained	Internal	✓	onlyInitializing
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_spendAllowance	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
	_afterTokenTransfer	Internal	✓	

ERC20PermitUpgradeable	Implementation	Initializable, ERC20Upgradeable, IERC20PermitUpgradeable, EIP712Upgradeable		
	__ERC20Permit_init	Internal	✓	onlyInitializing
	__ERC20Permit_init_unchained	Internal	✓	onlyInitializing
	permit	Public	✓	-
	nonces	Public		-
	DOMAIN_SEPARATOR	External		-
	_useNonce	Internal	✓	
ERC20VotesUpgradeable	Implementation	Initializable, IVotesUpgradeable, ERC20PermitUpgradeable		
	__ERC20Votes_init	Internal	✓	onlyInitializing
	__ERC20Votes_init_unchained	Internal	✓	onlyInitializing
	checkpoints	Public		-
	numCheckpoints	Public		-
	delegates	Public		-
	getVotes	Public		-
	getPastVotes	Public		-
	getPastTotalSupply	Public		-
	_checkpointsLookup	Private		

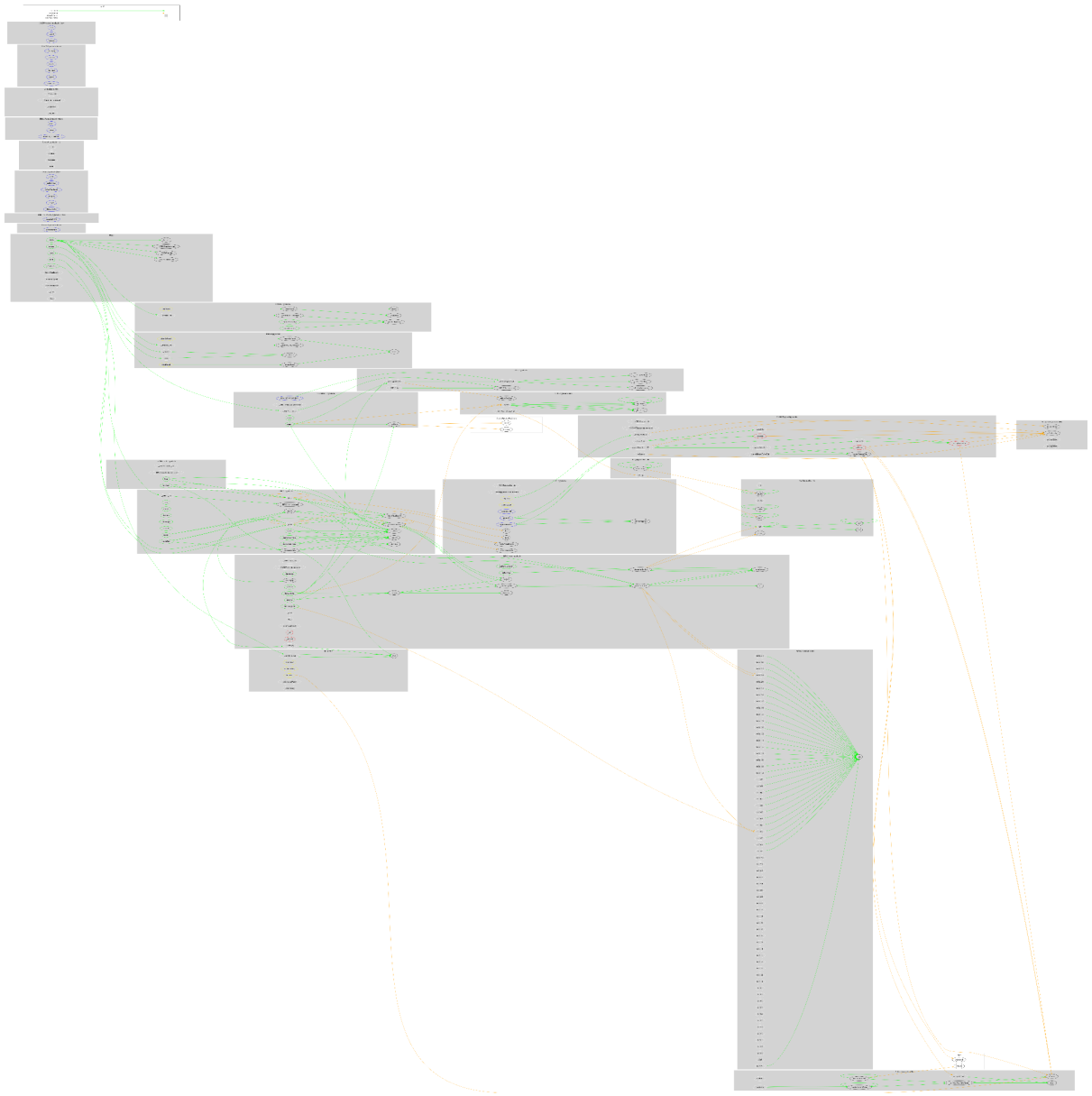
	delegate	Public	✓	-
	delegateBySig	Public	✓	-
	_maxSupply	Internal		
	_mint	Internal	✓	
	_burn	Internal	✓	
	_afterTokenTransfer	Internal	✓	
	_delegate	Internal	✓	
	_moveVotingPower	Private	✓	
	_writeCheckpoint	Private	✓	
	_add	Private		
	_subtract	Private		
	_unsafeAccess	Private		
ERC20Burnable Upgradeable	Implementation	Initializable, ContextUpgradeable, ERC20Upgradeable		
	__ERC20Burnable_init	Internal	✓	onlyInitializing
	__ERC20Burnable_init_unchained	Internal	✓	onlyInitializing
	burn	Public	✓	-
	burnFrom	Public	✓	-

Disney	Implementation	Initializable, ERC20Upgradeable, ERC20BurnableUpgradeable, PausableUpgradeable, OwnableUpgradeable, ERC20PermitUpgradeable, ERC20VotesUpgradeable, UUPSUpgradeable		
		Public	✓	-
	initialize	Public	✓	initializer
	pause	Public	✓	onlyOwner
	unpause	Public	✓	onlyOwner
	mint	Public	✓	onlyOwner
	_beforeTokenTransfer	Internal	✓	whenNotPaused
	_authorizeUpgrade	Internal	✓	onlyOwner
	_afterTokenTransfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	

Inheritance Graph



Flow Graph



Summary

Disney contract implements a token mechanism. The contract is implemented using an upgradable proxy pattern. This audit investigates security issues, business logic concerns and potential improvements. There are some functions that can be abused by the owner like stop transactions and mint tokens. if the contract owner abuses the mint functionality, then the contract will be highly inflated. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. 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 Cyberscope's prior written consent. 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 regarded as an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Cyberscope to perform a security assessment. This document 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 represents an extensive assessment 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 Cyberscope's position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope'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 Cyberscope are subject to dependencies and are 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.

About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.

The Cyberscope team

https://www.cyberscope.io