AegisX

Smart Contract Audits | KYC

PALLADIUM

Security Assessment

DayOfDefeat NFT

November 5, 2022

Assessment Summary

This report has been prepared for DayOfDefeat NFT on the Binance Smart Chain network. AegisX provides both client-centered and user-centered examination of the smart contracts and their current status when applicable. This report represents the security assessment made to find issues and vulnerabilities on the source code along with the current liquidity and token holder statistics of the protocol.

A comprehensive examination has been performed, utilizing Cross Referencing, Static Analysis, In-House Security Tools, and line-by-line Manual Review.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Inspecting liquidity and holders statistics to inform the current status to both users and client when applicable.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Verifying contract functions that allow trusted and/or untrusted actors to mint, lock, pause, and transfer assets.
- 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.

Technical Findings Summary

Classification of Risk

Severity	Description
🔴 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.
🟠 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.
🟡 Medium	Risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform
🟢 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.
ℹ Informational	Errors are often recommended to improve the code's style or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

Findings

Severity	Found	Pending	Resolved
🔴 Critical	0	0	0
🟠 Major	0	0	0
🟡 Medium	0	0	0
🟢 Minor	5	5	0
ℹ Informational	1	1	0
Total	6	6	0

Project Overview

Contract Summary

Parameter	Result
Address	
Name	DayOfDefeat
Token Tracker	DayOfDefeat (DOD)
Decimals	N/A
Supply	N/A
Platform	Binance Smart Chain
compiler	v0.8.0^
Contract Name	DayofdefeatNFT
Optimization	N/A
LicenseType	MIT
Language	Solidity
Codebase	N/A
Payment Tx	

Project Overview

Risk Analysis Summary

Parameter	Result
Buy Tax	0%
Sale Tax	0%
Is honeypot?	Clean
Can edit tax?	N/A
Is anti whale?	Yes
Is blacklisted?	No
Is whitelisted?	No
Holders	Clean
Security Score	85/100
Auditor Score	85/100
Confidence Level	Medium

The following quick summary it's added to the project overview; however, there are more details about the audit and its results. Please read every detail.

Main Contract Assessed
Contract Name

Name	Contract	Live
DayOfDefeat		No

TestNet Contract Assessed
Contract Name

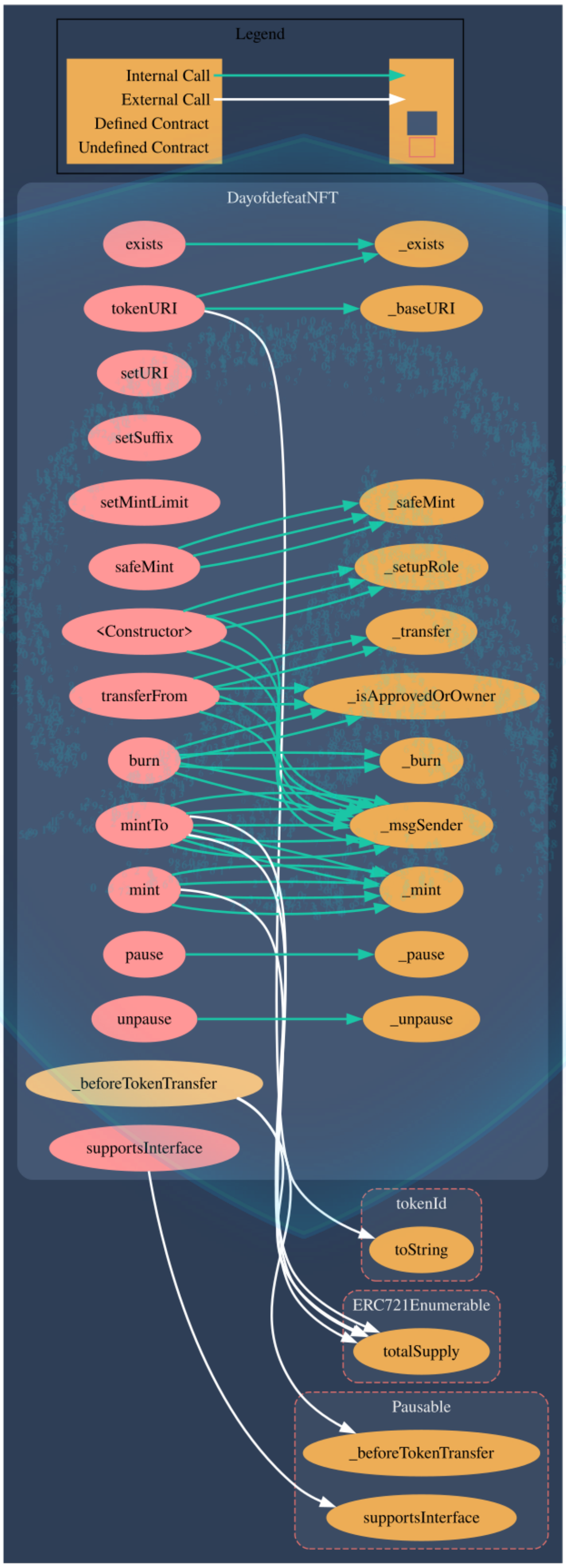
Name	Contract	Live
DayOfDefeat	0xA249A665C2d2B2593495BFA5479730B5Bcf5140E	No

Solidity Code Provided

SolID	File Sha-1	FileName
DayofdefeatNFT	23448fba856247ca8038fc9e9e181c596bbdef4e	DayofdefeatNFT.sol

Call Graph

The contract for DayOfDefeat has the following call graph structure.



KYC Information

The Project Owners of DayOfDefeat is not KYC. .

The owner wallet has the power to call the functions displayed on the priviliged functions chart below, if the owner wallet is compromised this privileges could be exploited.

We recommend the team to renounce ownership at the right timing if possible, or gradually migrate to a timelock with governing functionalities in respect of transparency and safety considerations.

KYC Information Notes:

Auditor Notes: N/A

Project Owner Notes:

Smart Contract Vulnerability Checks

ID	Severity	Name	File	locatio n
SWC-100	Pass	Function Default Visibility	DayofdefeatNFT.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-103	Low	A floating pragma is set.	DayofdefeatNFT.sol	L: 2 C: 3
SWC-104	Pass	Unchecked Call Return Value.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	DayofdefeatNFT.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	DayofdefeatNFT.sol	L: 0 C: 0
SWC-109	Pass	Uninitialized Storage Pointer.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-111	Pass	Use of Deprecated Solidity Functions.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-112	Pass	Delegate Call to Untrusted Callee.	DayofdefeatNFT.sol	L: 0 C: 0

ID	Severity	Name	File	locatio n
SWC-113	Pass	Multiple calls are executed in the same transaction.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-114	Pass	Transaction Order Dependence.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-115	Pass	Authorization through tx.origin.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-116	Pass	A control flow decision is made based on The block.timestamp environment variable.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-117	Pass	Signature Malleability.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-118	Pass	Incorrect Constructor Name.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-119	Pass	Shadowing State Variables.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-120	Pass	Potential use of block.number as source of randonmness.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-121	Pass	Missing Protection against Signature Replay Attacks.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-122	Pass	Lack of Proper Signature Verification.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-123	Pass	Requirement Violation.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-124	Pass	Write to Arbitrary Storage Location.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-125	Pass	Incorrect Inheritance Order.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-126	Pass	Insufficient Gas Griefing.	DayofdefeatNFT.sol	L: 0 C: 0

ID	Severity	Name	File	locatio n
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-129	Pass	Typographical Error.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-130	Pass	Right-To-Left-Override control character (U+202E).	DayofdefeatNFT.sol	L: 0 C: 0
SWC-131	Pass	Presence of unused variables.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-132	Pass	Unexpected Ether balance.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	DayofdefeatNFT.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	DayofdefeatNFT.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	DayofdefeatNFT.sol	L: 0 C: 0

We scan the contract for additional security issues using MYTHX and industry-standard security scanning tools.

Smart Contract Vulnerability Details

SWC-103 - Floating Pragma.

CWE-664: Improper Control of a Resource Through its Lifetime.

References:

Description:

Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly. Locking the pragma helps to ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

Remediation:

Lock the pragma version and also consider known bugs (https://github.com/ethereum/solidity/releases) for the compiler version that is chosen.

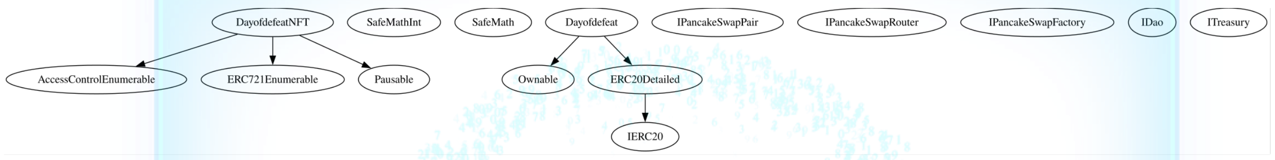
Pragma statements can be allowed to float when a contract is intended for consumption by other developers, as in the case with contracts in a library or EthPM package. Otherwise, the developer would need to manually update the pragma in order to compile locally.

References:

Ethereum Smart Contract Best Practices - Lock pragmas to specific compiler version.

Inheritance

The contract for DayOfDefeat has the following inheritance structure.



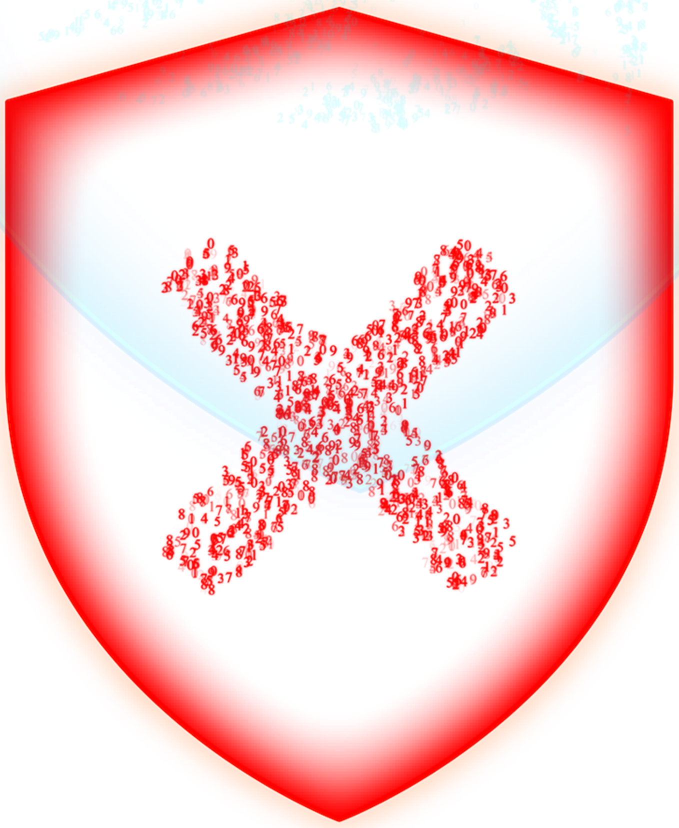
Privileged Functions (onlyOwner)

Function Name	Parameters	Visibility
N/A	N/A	N/A

Assessment Results

- DEFAULT_ADMIN_ROLE can set a limit on how many LIMIT_MINTER_ROLE can mint.
- There isn't a limit on how many the MINTER_ROLE can mint.
- PAUSE_ROLE can pause the minting or trading of NFT's.
- The aforementioned roles carry centralization risks and in light of the project's goal to eliminate centralization, it is strongly recommended to utilize multi-sig contract safes for these roles.
- LIMIT_MINTER_ROLE isn't defined in the contract.

Audit Fail



DOD-02 | Function Visibility Optimization.

Category	Severity	Location	Status
Gas Optimization	Informational	DayofdefeatNFT.sol:	Pending

Description

The following functions are declared as public and are not invoked in any of the contracts contained within the projects scope:

Function Name	Parameters	Visibility
tokenURI		public
setURI		public
setSuffix		public
setMintLimit		public
exists		public
transferFrom		public
safeMint		public
mint		public
mintTo		public
burn		public
pause		public
unpause		public

Function Name	Parameters	Visibility
supportsInterface		public

The functions that are never called internally within the contract should have external visibility

Remediation

We advise that the function's visibility specifiers are set to external, and the array-based arguments change their data location from memory to calldata, optimizing the gas cost of the function.

References:

external vs public best practices.

DOD-03 | Lack of Input Validation.

Category	Severity	Location	Status
Volatile Code	● Minor	DayofdefeatNFT.sol: 61,5, 69,5	Pending

Description

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

Remediation

We advise the client to add the check for the passed-in values to prevent unexpected errors as below:

```
...
 require(receiver != address(0), "Receiver is the zero address");
...
```

DOD-05 | Missing Event Emission.

Category	Severity	Location	Status
Volatile Code	🟢 Minor	DayofdefeatNFT.sol: 61,5, 69,5	Pending

Description

Detected missing events for critical arithmetic parameters. There are functions that have no event emitted, so it is difficult to track off-chain changes.The linked code does not create an event for the transfer.

Remediation

Emit an event for critical parameter changes. It is recommended emitting events for the sensitive functions that are controlled by centralization roles.

DOD-06 | Conformance with Solidity Naming Conventions.

Category	Severity	Location	Status
Coding Style	● Minor	DayofdefeatNFT.sol: 77,5, 184,5	Pending

Description

Solidity defines a naming convention that should be followed. Rule exceptions: Allow constant variable name/symbol/decimals to be lowercase. Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
transferFrom
safeMint
mint
mintTo
burn
```

Remediation

Follow the Solidity naming convention.

https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-convention

DOD-11 | Modifier: LIMIT_MINTER_ROLE.

Category	Severity	Location	Status
Modifier Usage	🟢 Minor	DayofdefeatNFT.sol: 14,5, 19,5	Pending

Description

During our review we noticed that LIMIT_MINTER_ROLE does not get assigned by the contract.

Remediation

Waiting for dev's response.

Project Action

Pending Customer Response

DOD-12 | Centralization Risks In The DEFAULT_ADMIN_ROLE, MINTER_ROLE, PAUSER_ROLE Role(s)

Category	Severity	Location	Status
Centralization / Privilege	🟢 Minor	DayofdefeatNFT.sol: 42, 58	Pending

Description

In the contract DayofdefeatNFT, the role DEFAULT_ADMIN_ROLE, MINTER_ROLE, PAUSER_ROLE has authority over the functions that lead to centralization risks.
Any compromise to the DEFAULT_ADMIN_ROLE, MINTER_ROLE, PAUSER_ROLE account(s) may allow the hacker to take advantage of this authority.

Remediation

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.

Project Action

Pending Customer Response

Social Media Checks

Social Media	URL	Result
Website	https://www.dayofdefeat.app/	Pass
Telegram	https://t.me/DayOfDefeatBSC	Pass
Twitter	https://twitter.com/dayofdefeatBSC	Pass
OtherSocial	https://titanservice.cn/dayofdefeatCN	Pass

We recommend to have 3 or more social media sources including a completed working websites.

Social Media Information Notes:

Auditor Notes: undefined

Project Owner Notes:

Appendix

Finding Categories

Centralization / Privilege

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

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimalEVM 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 howblock.timestamp works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functionsbeing invoke-able by anyone under certain circumstances.

Volatile Code

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

Coding Style

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

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code,such as a constructor assignment imposing different require statements on the input variables than a setterfunction.

Coding Best Practices

ERC 20 Conding Standards are a set of rules that each developer should follow to ensure the code meet a set of creterias and is readable by all the developers.

Disclaimer

AegisX has conducted an independent security assessment to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the reviewed code for the scope of this assessment. This report does not constitute agreement, acceptance, or advocation for the Project, and users relying on this report should not consider this as having any merit for financial advice in any shape, form, or nature. The contracts audited do not account for any economic developments that the Project in question may pursue, and the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude, and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are entirely free of exploits, bugs, vulnerabilities or deprecation of technologies.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence, regardless of the findings presented. Information is provided 'as is, and AegisX is under no covenant to audited completeness, accuracy, or solidity of the contracts. In no event will AegisX or its partners, employees, agents, or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions or actions with regards to the information provided in this audit report.

The assessment services provided by AegisX are subject to dependencies and are under continuing development. You agree that your access 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 with high levels of technical risk and uncertainty. The assessment reports could include false positives, negatives, and unpredictable results. The services may access, and depend upon, multiple layers of third parties.

AegisX