Cyberscope

Audit Report

KonnektVPN

May 2024

Network MATIC

Address 0x8328e6fceC9477C28298c9f02d740Dd87a1683e5

Audited by © cyberscope

Analysis

Critical Medium Minor / Informative Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	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
●	MWIU	MultiSig Wallet Ineffective Usage	Unresolved
●	MEE	Missing Events Emission	Unresolved
●	L07	Missing Events Arithmetic	Unresolved
●	L08	Tautology or Contradiction	Unresolved
●	L19	Stable Compiler Version	Unresolved

Table of Contents

Review

Contract Name	KPNToken
Compiler Version	v0.8.20+commit.a1b79de6
Optimization	No with 200 runs
Explorer	https://polygonscan.com/address/0x8328e6fceC9477C28298c9f02d740Dd87a1683e5
Address	0x8328e6fceC9477C28298c9f02d740Dd87a1683e5
Name	KonnektVPN
Symbol	KPN
Decimals	18
Total Supply	251,663,110

Audit Updates

Initial Audit	02 May 2024

Source Files

Filename	SHA256
contracts/KPNToken.sol	a8b2bd55351a3a7721f4ed8deb5e8edc6098a7885de10157280cb291125cfd64
@openzeppelin/contracts/utils/Context.sol	847fda5460fee70f56f4200f59b82ae622bb03c79c77e67af010e31b7e2cc5b6
@openzeppelin/contracts/utils/Address.sol	b3710b1712637eb8c0df81912da3450da6ff67b0b3ed18146b033ed15b1aa3b9
@openzeppelin/contracts/utils/introspection/IERC165.sol	07ae1ac964ab74dedada999e2dfc642031a6495469cffc0bf715daa4f1e4f904
@openzeppelin/contracts/utils/introspection/ERC165.sol	99348354365cbdeb90157e2903334b861a00d69faab7720ae542d911d5c70d87
@openzeppelin/contracts/token/ERC20/IERC20.sol	6f2faae462e286e24e091d7718575179644dc60e79936ef0c92e2d1ab3ca3cee
@openzeppelin/contracts/token/ERC20/ERC20.sol	2d874da1c1478ed22a2d30dcf1a6ec0d09a13f897ca680d55fb49fbcc0e0c5b1
@openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol	471157c89111d7b9eab456b53ebe9042bc69504a64cb5cc980d38da9103379ae
@openzeppelin/contracts/token/ERC20/extensions/IERC20Permit.sol	912509e0e9bf74e0f8a8c92d031b5b26d2d35c6d4abf3f56251be1ea9ca946bf
@openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol	1d079c20a192a135308e99fa5515c27acfbb071e6cdb0913b13634e630865939
@openzeppelin/contracts/interfaces/draft-IERC6093.sol	4aea87243e6de38804bf8737bf86f750443d3b5e63dd0fd0b7ad92f77cdbd3e3
@openzeppelin/contracts/access/IAccessControl.sol	1d6ef09193265172824fa1139e85cba422117ca918961183f080e692489d8c3b

@openzeppelin/contracts/access/AccessControl.sol	1086a1ad3788972b885ff3f209da510615dde6214d46b29e1cd2a4924f66c06d

Findings Breakdown

Critical	2
Medium	0
Minor / Informative	4

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	2	0	0	0
Medium	0	0	0	0
Minor / Informative	4	0	0	0

MT - Mints Tokens

Criticality	Critical
Location	contracts/KPNToken.sol#L43,48
Status	Unresolved

Description

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

```
function mint(uint256 amount) public onlyRole(ADMIN_ROLE){
    require(liveSupply + amount <= maxSupply, "ERC20: Over Max Supply
Error");
    liveSupply += amount;
    _mint(msg.sender, amount);
}
function mintToMiners(address target, uint256 amount) external
onlyRole(ADMIN_ROLE){
    require(liveSupply + amount <= maxSupply, "ERC20: Over Max Supply
Error");
    liveSupply += amount;
    _mint(target, amount);
}
```

Recommendation

The team should carefully manage the private keys of the admins' accounts. At the time being, the ownership has been transferred to a multiSig wallet.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

MWIU - MultiSig Wallet Ineffective Usage

Criticality	Critical
Location	contracts/KPNToken.sol#L35
Status	Unresolved

Description

The contract incorporates a multiSig wallet for increased security measures. However, a critical flaw arises during initialization, wherein the admin role is assigned not only to the multiSig wallet but also to the original owner. This assignment undermines the intended purpose of the multiSig wallet, as the original owner retains the ability to execute functions with the `onlyRole(ADMIN_ROLE)` modifier. Additionally, the admin role has the authority to grant roles outside the context of the multiSig wallet. Consequently, the contract fails to fully leverage the security benefits offered by a multiSig setup.

```
_grantRole(DEFAULT_ADMIN_ROLE, multiSigWalletAddress); // Assigns the
multiSigWalletAddress the default admin role
_grantRole(ADMIN_ROLE, multiSigWalletAddress);
_grantRole(ADMIN_ROLE, owners[0]);
...
function grantRole(bytes32 role, address account) public override
onlyRole(getRoleAdmin(role)) {
    super.grantRole(role, account);
    if (role == ADMIN_ROLE && !_isInArray(account, adminAddresses)) {
        adminAddresses.push(account);
    } else {
        revert("Role not found!");
    }
}
```

Recommendation

The team is advised to either modify the contract's initialization process to exclusively assign the admin role to the multiSig wallet, thus ensuring that critical functions can only be executed through multi-signature authorization, or revoke the ADMIN_ROLE for the owner.

MEE - Missing Events Emission

Criticality	Minor / Informative
Location	contracts/KPNToken.sol#L70,79
Status	Unresolved

Description

The contract performs actions and state mutations from external methods that do not result in the emission of events. Emitting events for significant actions is important as it allows external parties, such as wallets or dApps, to track and monitor the activity on the contract. Without these events, it may be difficult for external parties to accurately determine the current state of the contract.

```
adminAddresses.push(account);
_removeFromArray(account, adminAddresses);
```

Recommendation

It is recommended to include events in the code that are triggered each time a significant action is taking place within the contract. These events should include relevant details such as the user's address and the nature of the action taken. By doing so, the contract will be more transparent and easily auditable by external parties. It will also help prevent potential issues or disputes that may arise in the future.

L07 - Missing Events Arithmetic

Criticality	Minor / Informative
Location	contracts/KPNToken.sol#L45,58
Status	Unresolved

Description

Events are a way to record and log information about changes or actions that occur within a contract. They are often used to notify external parties or clients about events that have occurred within the contract, such as the transfer of tokens or the completion of a task.

It's important to carefully design and implement the events in a contract, and to ensure that all required events are included. It's also a good idea to test the contract to ensure that all events are being properly triggered and logged.

```
liveSupply += amount
liveSupply -= amount
```

Recommendation

By including all required events in the contract and thoroughly testing the contract's functionality, the contract ensures that it performs as intended and does not have any missing events that could cause issues with its arithmetic.

L08 - Tautology or Contradiction

Criticality	Minor / Informative
Location	contracts/KPNToken.sol#L56
Status	Unresolved

Description

A tautology is a logical statement that is always true, regardless of the values of its variables. A contradiction is a logical statement that is always false, regardless of the values of its variables.

Using tautologies or contradictions can lead to unintended behavior and can make the code harder to understand and maintain. It is generally considered good practice to avoid tautologies and contradictions in the code.

```
require(liveSupply - amount >= 0, "ERC20: Cannot Burn more than current supply")
```

Recommendation

The team is advised to carefully consider the logical conditions is using in the code and ensure that it is well-defined and make sense in the context of the smart contract.

L19 - Stable Compiler Version

Criticality	Minor / Informative
Location	contracts/KPNToken.sol#L3
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 ^0.8.9;
```

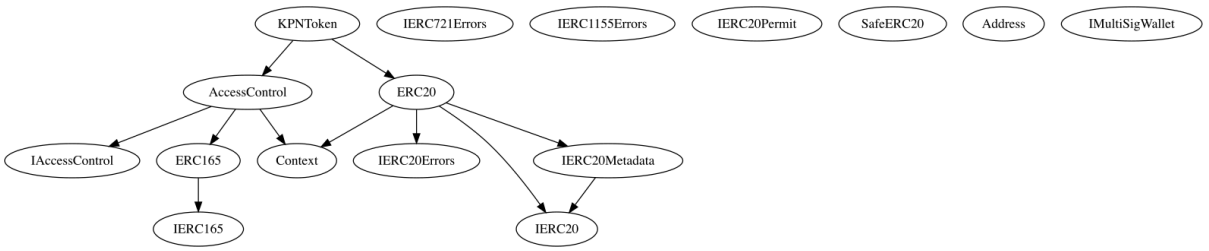
Recommendation

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.

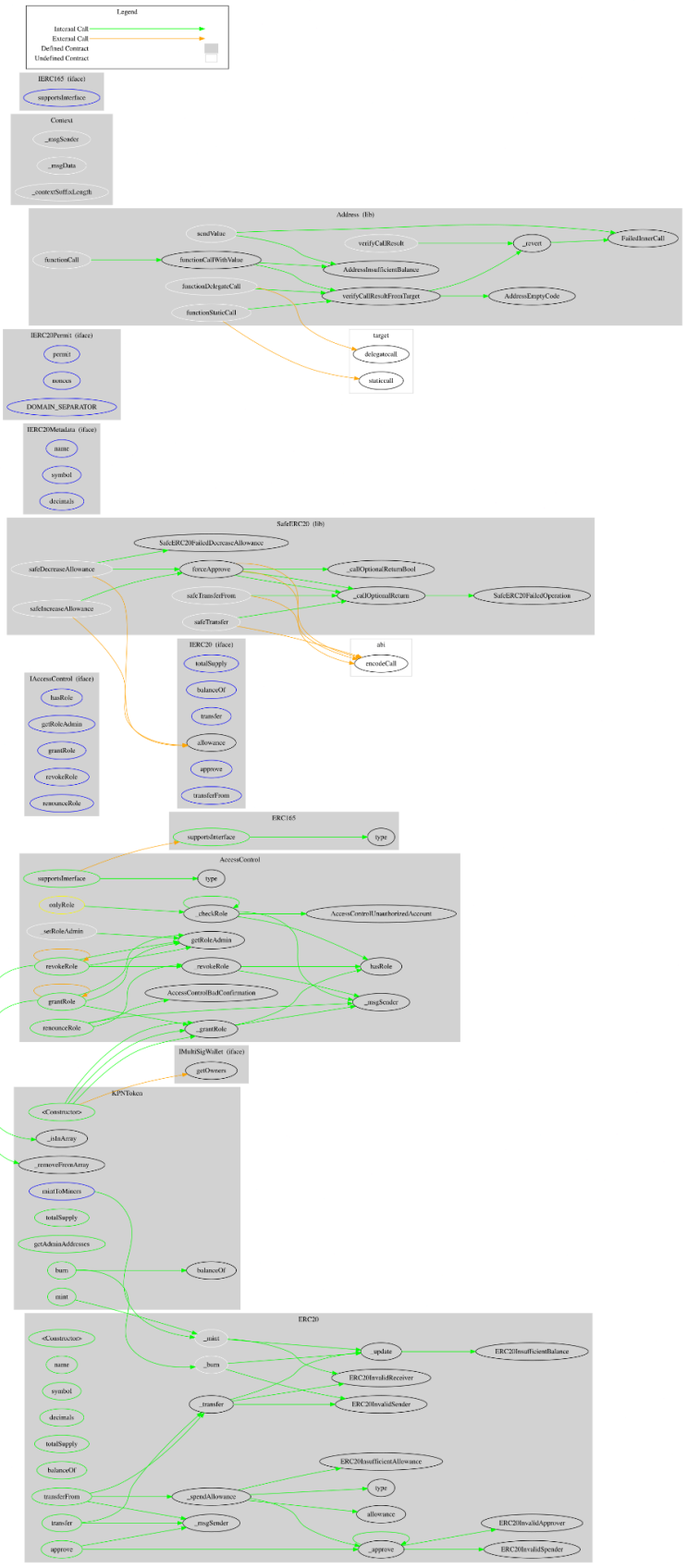
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IMultiSigWallet	Interface			
	getOwners	External		-
KPNToken	Implementation	ERC20, AccessControl		
		Public	✓	ERC20
	mint	Public	✓	onlyRole
	mintToMiners	External	✓	onlyRole
	burn	Public	✓	-
	totalSupply	Public		-
	grantRole	Public	✓	onlyRole
	revokeRole	Public	✓	onlyRole
	getAdminAddresses	Public		-
	_isInArray	Private		
	_removeFromArray	Private	✓	

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
IERC165 (iface)
supportsInterface
Context
_msgSender
_msgData
_contextSuffixLength
Address (lib)
sendValue
functionCall
functionCallWithValue
functionDelegateCall
functionStaticCall
verifyCallResult
verifyCallResultFromTarget
_revert
FailedInnerCall
AddressInsufficientBalance
AddressEmptyCode
target
delegatecall
staticcall
IERC20Permit (iface)
permit
nonces
DOMAIN_SEPARATOR
IERC20Metadata (iface)
name
symbol
decimals
SafeERC20 (lib)
SafeERC20FailedDecreaseAllowance
safeDecreaseAllowance
safeIncreaseAllowance
forceApprove
safeTransferFrom
safeTransfer
_callOptionalReturnBool
_callOptionalReturn
SafeERC20FailedOperation
abi
encodeCall
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
IAccessControl (iface)
hasRole
getRoleAdmin
grantRole
revokeRole
renounceRole
ERC165
supportsInterface
type
AccessControl
onlyRole
setRoleAdmin
checkRole
AccessControlUnauthorizedAccount
AccessControlBadConfirmation
_grantRole
IMultiSigWallet (iface)
getOwners
KPNToken
<Constructor>
_isInArray
_removeFromArray
minterIsMinter
totalSupply
getAdminAddresses
burn
mint
ERC20
_mint
_burn
_transfer
_update
ERC20InsufficientBalance
ERC20InvalidReceiver
ERC20InvalidSender
name
symbol
decimals
totalSupply
balanceOf
transferFrom
transfer
approve
_spendAllowance
_msgSender
ERC20InsufficientAllowance
type
allowance
_approve
ERC20InvalidApprover
ERC20InvalidSpender

Summary

KonnektVPN contract implements a token mechanism. This audit investigates security issues, business logic concerns, and potential improvements. There are some functions that can be abused by the owner like mint tokens. If the contract owner abuses the mint functionality, then the contract will be highly inflated. 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