

SKELETON ECOSYSTEM

SMART CONTRACT AUDIT



DVPN Network

DVPN

ERC20

0x22994fdB3f8509cf6a729BBfA93F939Db0B50

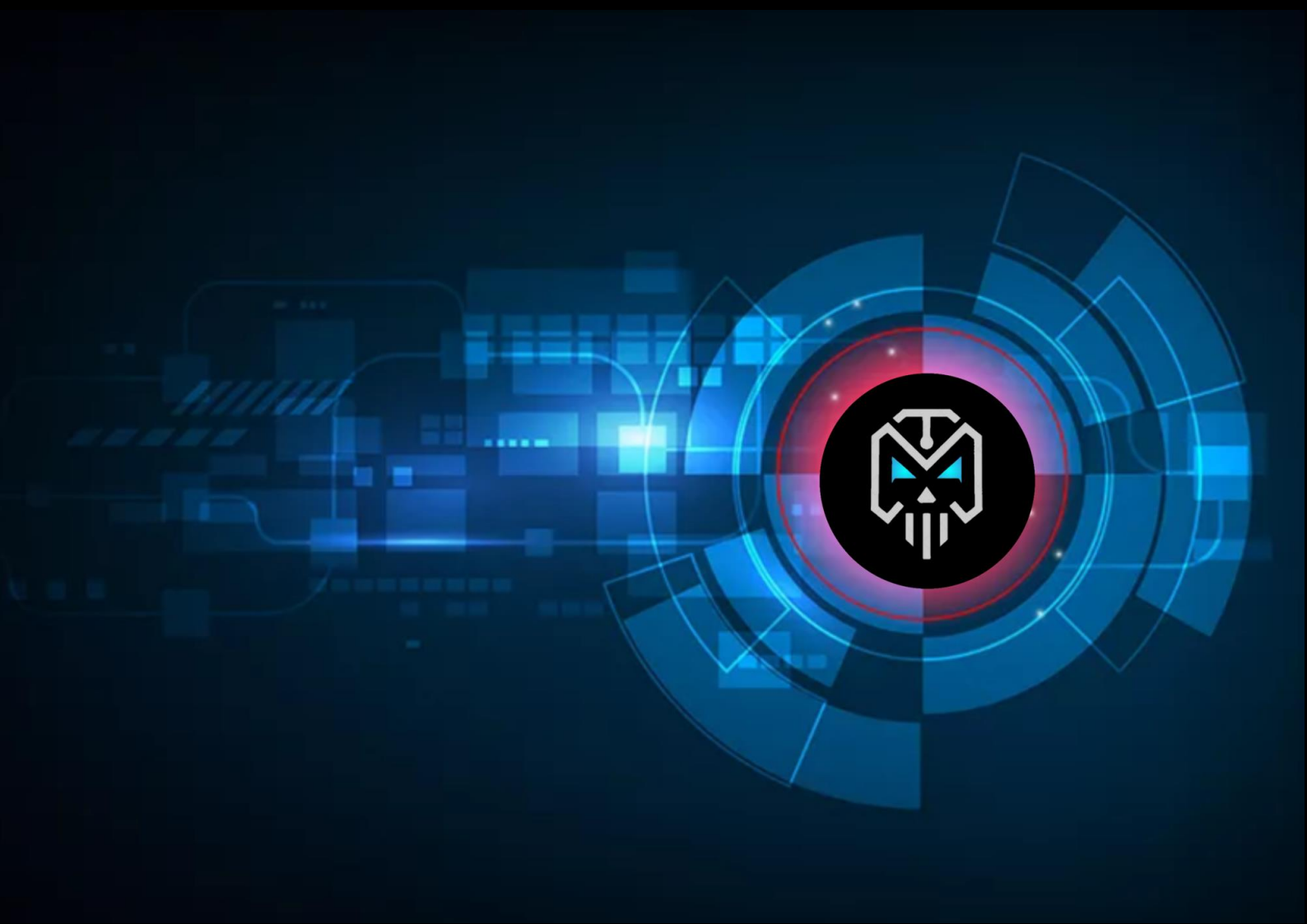


Table of Contents

| | |
|---------------------------------------|----|
| Table of Contents | 1 |
| Disclaimer | 2 |
| Overview | 3 |
| Creation/Audit Date | 3 |
| Verified Socials | 3 |
| Contract Functions Analysis | 4 |
| Contract Safety and Weakness | 8 |
| Detected Vulnerability Description | 12 |
| Contract Flow Graph | 16 |
| Contract Interaction Graph | 17 |
| Inheritance Graph | 18 |
| Contract Descriptions | 19 |
| Audit Scope | 24 |

Global Disclaimer

This document serves as a disclaimer for the crypto smart contract audit conducted by Skeleton Ecosystem. The purpose of the audit was to review the codebase of the smart contracts for potential vulnerabilities and issues. It is important to note the following:

Limited Scope: The audit is based on the code and information available up to the audit completion date. It does not cover external factors, system interactions, or changes made after the audit. The audit itself can not guarantee 100% safety and can not detect common scam methods like farming and developer sell-out.

No Guarantee of Security: While we have taken reasonable steps to identify vulnerabilities, it is impossible to guarantee the complete absence of security risks or issues. The audit report provides an assessment of the contract's security as of the audit date.

Continued Development: Smart contracts and blockchain technology are evolving fields. Updates, forks, or changes to the contract post-audit may introduce new risks that were not present during the audit.

Third-party Code: If the smart contract relies on third-party libraries or code, those components were not thoroughly audited unless explicitly stated. Security of these dependencies is the responsibility of their respective developers.

Non-Exhaustive Testing: The audit involved automated analysis, manual review, and testing under controlled conditions. It is possible that certain vulnerabilities or issues may not have been identified.

Risk Evaluation: The audit report includes a risk assessment for identified vulnerabilities. It is recommended that the development team carefully reviews and addresses these risks to mitigate potential exploits.

Not Financial Advice: This audit report is not intended as financial or investment advice. Decisions regarding the use, deployment, or investment in the smart contract should be made based on a comprehensive assessment of the associated risks.

By accessing and using this audit report, you acknowledge and agree to the limitations outlined above. Skeleton Ecosystem and its auditors shall not be held liable for any direct or indirect damages resulting from the use of the audit report or the smart contract itself.

Please consult with legal, technical, and financial professionals before making any decisions related to the smart contract.

Overview

| | |
|-----------------------|---|
| Contract Name | DVPN |
| Ticker/Symbol | DVPN |
| Blockchain | Ethereum ERC20 |
| Contract Address | 0x22994fdB3f8509cf6a729BBfA93F939Db0B50D06 |
| Creator Address | 0x79B10f7A443D4050F5f12354d95D925bF747845d |
| Current Owner Address | 0x00 |
| Contract Explorer | https://etherscan.io/address/0x22994fdB3f8509cf6a729BBfA93F939Db0B50D06#code |
| Compiler Version | v0.8.21+commit.d9974bed |
| License | MIT |
| Optimisation | No with 200 Runs |
| Total Supply | 100,000,000DVPN |
| Decimals | 18 |

Creation/Audit

| | |
|-------------------|------------|
| Contract Deployed | 22.05.2024 |
| Audit Created | 23.05.2023 |
| Audit Update | V 1.0 |

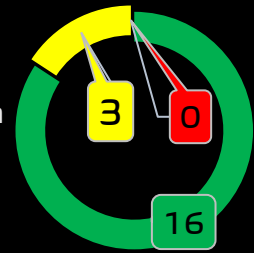
Verified Socials








| | |
|-------------|---|
| Website | https://devpn-erc.com/ |
| Telegram | https://t.me/dvpnnetwork |
| Twitter (X) | https://x.com/dvpnnetworkerc |











Contract Function Analysis

 Pass
  Attention Item
  Risky Item

■ Pass
 ■ Attention
 ■ Risk



| | | |
|---------------------------|---|--|
| Contract Verified |  | The contract source code is uploaded to blockchain explorer and is open source, so everybody can read it. |
| Contract Ownership | | 0x00 Deployer |
| Buy Tax | 5 % | Shows the taxes for purchase transactions. Above 10% may be considered a high tax rate. More than 50% tax rate means may not be tradable. Fee can be set! |
| Sell Tax | 5 % | Shows the taxes for sell transactions. Above 10% may be considered a high tax rate. More than 50% tax rate means may not be tradable. Fee can be set! |
| Honeypot Analyse |  | Holder is able to buy and sell. If honeypot: The contract blocks sell transfer from holder wallet. Multiple events may cause honeypot. Trading disabled, extremely high tax |
| Liquidity Status |  | Liquidity status on 22.05.2024 Locked 99.00% on Unicrypt locker for 364 days. https://etherscan.io/tx/0x2bb79fb5f7b2b5e44464c6e7a3acfbcb3208fa425ff638d47025436311fb42bc9 |
| Trading Disable Functions |  | No Trading suspendable function found. If a suspendable code is included, the token maybe neither be bought or sold (honeypot risk). If contract is renounced this function can't be used |
| Set Fees function |  max 5% | Fee Setting function found. Contract renounced, function can not be triggered by owner. The contract owner may contain the authority to modify the transaction tax. If the transaction tax is increased to more than 49%, the tokens may not be able to be traded (honeypot risk). |
| Proxy Contract |  | Not a Proxy contract |
| Mint Function |  | No Mint Function detected Mint function is transparent or non-existent. Hidden mint functions may increase the amount of tokens in circulation and effect the price of the token. Owner can mint new tokens and sell. |

| | | |
|---|---|--|
| Balance Modifier Function |  | <p>No Balance Modifier function found.</p> <p>If there is a function for this, the contract owner can have the authority to modify the balance of tokens at other addresses. For example revoke the bought tokens from the holders wallet. Common form of scam: You buy the token, but it's disappearing from your wallet.</p> |
| Blacklist Function |  | <p>No Blacklist Setting function found.</p> <p>If there is a blacklist, some addresses may not be able to trade normally. Example: you buy the token and right after your Wallet getting blacklisted. Like so you will be unable to sell. Honeypot Risk.</p> |
| Whitelist Function |  | <p>Whitelist Setting function found. Contract renounced, function can not be triggered by owner.</p> <p>If there is a function for this Developer can set zero fee or no max wallet size for addresses (for example team wallets can trade without fee. Can cause farming)</p> |
| Hidden Owner Analysis |  | <p>No Hidden or multi owner with authorisation</p> <p>For contract with a hidden owner, developer can still manipulate the contract even if the ownership has been abandoned.</p> |
| Retrieve Ownership Function |  | <p>No Functions found which can retrieve ownership of the contract.</p> <p>If this function exists, it is possible for the project owner to regain ownership even after relinquishing it. Also known as fake renounce.</p> |
| Self Destruct Function |  | <p>No Self Destruct function found.</p> <p>If this function exists and is triggered, the contract will be destroyed, all functions will be unavailable, and all related assets will be erased.</p> |
| Specific Tax Changing Function |  | <p>No Specific Tax Changing Functions found.</p> <p>If it exists, the contract owner may set a very outrageous tax rate for assigned address to block it from trading. Can assign all wallets at once!</p> |
| Trading Cooldown Function |  | <p>No Trading Cooldown Function found. If there is a trading cooldown function, the user will not be able to sell the token within a certain time or block after buying. Like a temporary honeypot.</p> |
| Max Transaction and Holding Modify Function |  | <p>Max Transaction and Holding Modify function found. Contract renounced, function can not be triggered by owner.</p> <p>If there is a function for this, the maximum trading amount or maximum position can be modified. Can cause honeypot</p> |
| Transaction Limiting Function |  | <p>No Transaction Limiter Function Found.</p> <p>The number of overall token transactions may be limited (honeypot risk)</p> |

Details of Risk - Attention Items

Removing Risk of contract function based on renounced ownership

Transaction Receipt Event Logs

470

Address `0x22994fdb3f8509cf6a729bbfa93f939db0b50d06`  

Name OwnershipTransferred (index_topic_1 address previousOwner, index_topic_2 address newOwner) [View Source](#)

Topics

0 `0x8be079c531659141344cd1fd0a4f28419497f9722a3daafe3b4186f6b6457e0`

1: previousOwner `Dec` \Rightarrow `0x79810f7A443D4050F5f12354d950925bF747845d`

2: newOwner `Dec` \Rightarrow `0x00`

Data `0x`

Following detected contract functions serve as informational purposes about the contract. The owner has no more authorisation to trigger the following functions.

Set Fee (Max 5% limit found)

Contract renounced, function can not be triggered by owner.

The contract owner may contain the authority to modify the transaction tax. If the transaction tax is increased to more than 49%, the tokens may not be able to be traded (honeypot risk).

```

564   ftrace | funcSig
565   function updateFees(uint256 _buyt, uint256 _sellt) external onlyOwner {
566       require(_buyt <= 5, "Exceed the limit");
567       require(_sellt <= 5, "Exceed the limit");
568       buyTotalFees = _buyt;
569       sellTotalFees = _sellt;
570   }

```

Contract renounced, function can not be triggered by owner.

```

513     function whitelistContract(address _whitelist!, bool isWL!) public onlyOwner {
514         isExcludedMaxTransactionAmount[_whitelist!] = isWL!;
515
516         isExcludedFromFees[_whitelist!] = isWL!;
517     }
518
519     ftrace | funcSig
520     function excludeFromMaxTransaction(address updAdst!, bool isEx!) public onlyOwner {
521         isExcludedMaxTransactionAmount[updAdst!] = isEx!;
522     }
523
524     // only use to disable contract sales if absolutely necessary (emergency use only)
525     ftrace | funcSig
526     function updateSwapEnabled(bool enabled!) external onlyOwner {
527         swapEnabled = enabled!;
528     }
529
530     ftrace | funcSig
531     function excludeFromFees(address account!, bool excluded!) public onlyOwner {
532         isExcludedFromFees[account!] = excluded!;
533         emit ExcludeFromFees(account!, excluded!);
534     }
535
536     ftrace | funcSig

```

If there is a function for this, the maximum trading amount or maximum position can be modified. Can cause honeypot

```

499 |         ftrace | funcSig
      | function updateMaxSwap(uint256 newAmount!) external onlyOwner {
500 |             maxSwapAmount = newAmount! * (10 ** 18);
501 |         }
502 |
503 |         ftrace | funcSig
      | function updateMaxTxnAmount(uint256 newNum!) external onlyOwner {
504 |             require(newNum! >= ((totalSupply() * 1) / 1000) / 1e18, "Cannot set maxTransactionAmount lower than 0.1%");
505 |             maxTransactionAmount = newNum! * (10 ** 18);
506 |         }
507 |
508 |         ftrace | funcSig
      | function updateMaxWalletAmount(uint256 newNum!) external onlyOwner {
509 |             require(newNum! >= ((totalSupply() * 5) / 1000) / 1e18, "Cannot set maxWallet lower than 0.5%");
510 |             maxWallet = newNum! * (10 ** 18);
511 |         }
512 |

```


Contract Security

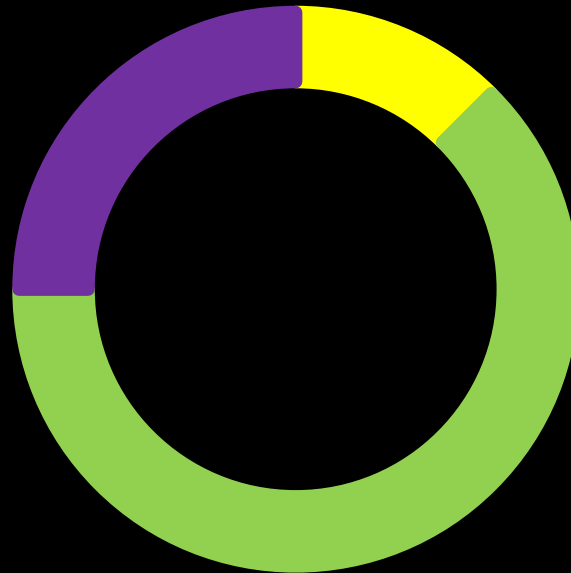
Total Findings: 8

■ High 0

■ Medium 1

■ Low 5

■ Info 2



■ **High Severity Issues:** High possibility to cause problems, need to be resolved.

■ **Medium Severity Issue:** Will likely cause problems, recommended to resolve.

■ **Low Severity Issues:** Won't cause problems, but for improvement purposes could be adjusted.

■ **Informational Severity Issues:** Not harmful in any way, information for the developer team.

Contract Security

List of Found Issues

High severity Issues: (0)

Medium severity issues: (1)

- Incorrect Access Control

Low severity issues: (5)

- Missing Events
- Long number literals
- Outdated compiler Version
- Approve of Front Running Attack
- Unchecked Array Length

Informational severity issues: (2)

- Public Functions Should be Declared External
- Costly Loop Operations

Contract Weakness Classisication

THE SMART CONTRACT WEAKNESS CLASSIFICATION REGISTRY (SWC REGISTRY) IS AN IMPLEMENTATION OF THE WEAKNESS CLASSIFICATION SCHEME PROPOSED IN EIP-1470. IT IS LOOSELY ALIGNED TO THE TERMINOLOGIES AND STRUCTURE USED IN THE COMMON WEAKNESS ENUMERATION (CWE) WHILE OVERLAYING A WIDE RANGE OF WEAKNESS VARIANTS THAT ARE

| ID | Description | AI | Manual | Result |
|---------|--------------------------------------|--------|--------|--------|
| SWC-100 | Function Default Visibility | Passed | Passed | Passed |
| SWC-101 | Integer Overflow and Underflow | Passed | Passed | Passed |
| SWC-102 | Outdated Compiler Version | low | low | low |
| SWC-103 | Floating Pragma | Passed | Passed | Passed |
| SWC-104 | Unchecked Call Return Value | Passed | Passed | Passed |
| SWC-105 | Unprotected Ether Withdrawal | Passed | Passed | Passed |
| SWC-106 | Unprotected SELFDESTRUCT Instruction | Passed | Passed | Passed |
| SWC-107 | Reentrancy | Passed | Passed | Passed |
| SWC-108 | State Variable Default Visibility | Passed | Passed | Passed |
| SWC-109 | Uninitialized Storage Pointer | Passed | Passed | Passed |
| SWC-110 | Assert Violation | Passed | Passed | Passed |
| SWC-111 | Use of Deprecated Solidity Functions | Passed | Passed | Passed |
| SWC-112 | Delegatecall to Untrusted Callee | Passed | Passed | Passed |
| SWC-113 | DoS with Failed Call | Passed | Passed | Passed |
| SWC-114 | Transaction Order Dependence | Passed | Passed | Passed |
| SWC-115 | Authorization through tx.origin | Passed | Passed | Passed |
| SWC-116 | Block values as a proxy for time | Passed | Passed | Passed |
| SWC-117 | Signature Malleability | Passed | Passed | Passed |
| SWC-118 | Incorrect Constructor Name | Passed | Passed | Passed |

| | | | | |
|---------|---|--------|--------|--------|
| SWC-119 | Shadowing State Variables | Passed | Passed | Passed |
| SWC-120 | Weak Sources of Randomness from Chain Attributes | Passed | Passed | Passed |
| SWC-121 | Missing Protection against Signature Replay Attacks | Passed | Passed | Passed |
| SWC-122 | Lack of Proper Signature Verification | Passed | Passed | Passed |
| SWC-123 | Requirement Violation | Passed | Passed | Passed |
| SWC-124 | Write to Arbitrary Storage Location | Passed | Passed | Passed |
| SWC-125 | Incorrect Inheritance Order | Passed | Passed | Passed |
| SWC-126 | Insufficient Gas Griefing | Passed | Passed | Passed |
| SWC-127 | Arbitrary Jump with Function Type Variable | Passed | Passed | Passed |
| SWC-128 | DoS With Block Gas Limit | Passed | Passed | Passed |
| SWC-129 | Typographical Error | low | Passed | Passed |
| SWC-130 | Right-To-Left-Override control character (U+202E) | Passed | Passed | Passed |
| SWC-131 | Presence of unused variables | Passed | Passed | Passed |
| SWC-132 | Unexpected Ether balance | Passed | Passed | Passed |
| SWC-133 | Hash Collisions With Multiple Variable Length Arguments | Passed | Passed | Passed |
| SWC-134 | Message call with hardcoded gas amount | Passed | Passed | Passed |
| SWC-135 | Code With No Effects | Passed | Passed | Passed |
| SWC-136 | Unencrypted Private Data On-Chain | Passed | Passed | Passed |

Detected High and Medium Severity Vulnerability Description.

⚠️ Approve of front running attack. Also known as Sandwich bot attack. (2 Items)

| | | | | |
|---------|-----------|--------------|-----------|-----|
| Item: 1 | Location: | Line 298-301 | Severity: | Low |
|---------|-----------|--------------|-----------|-----|

| | |
|--------------------|---|
| Function | <p>The <code>approve()</code> method overrides current allowance regardless of whether the spender already used it or not, so there is no way to increase or decrease allowance by a certain value atomically unless the token owner is a smart contract, not an account. This can be abused by a token receiver when they try to withdraw certain tokens from the sender's account. Meanwhile, if the sender decides to change the amount and sends another approve transaction, the receiver can notice this transaction before it's mined and can extract tokens from both the transactions, therefore, ending up with tokens from both the transactions. This is a front-running attack affecting the ERC20 Approve function.</p> <p>The function approve can be front-run by abusing the <code>_approve</code> function.</p> |
| Remediation | <ol style="list-style-type: none"> 1. Introduce mechanisms that limit the maximum acceptable gas price for transactions. This can help prevent front-runners from drastically increasing the gas fees to prioritize their transactions. 2. Use transaction taxes to prevent against front-run attack |

```

ftrace | funcSig
298 function approve(address spender!, uint256 amount!) public virtual override returns (bool) {
299     _approve(_msgSender(), spender!, amount!);
300     return true;
301 }
302
  
```

| | | |
|---------|------------------------|--|
| Item: 2 | Location: Line 303-313 | Severity: ■ Low |
|---------|------------------------|--|

| | |
|-------------------|--|
| Function | <p>The transferFrom() method overrides current allowance regardless of whether the spender already used it or not, so there is no way to increase or decrease allowance by a certain value atomically unless the token owner is a smart contract, not an account.</p> <p>This can be abused by a token receiver when they try to withdraw certain tokens from the sender's account.</p> <p>Meanwhile, if the sender decides to change the amount and sends another approve transaction, the receiver can notice this transaction before it's mined and can extract tokens from both the transactions, therefore, ending up with tokens from both the transactions. This is a front-running attack affecting the ERC20 Approve function.</p> <p>The function approve can be front-run by abusing the <code>_approve</code> function.</p> |
| Remedation | <ol style="list-style-type: none"> 1. Introduce mechanisms that limit the maximum acceptable gas price for transactions. This can help prevent front-runners from drastically increasing the gas fees to prioritize their transactions. 2. Use transaction taxes to prevent against front-run attack |

```

303  function transferFrom(address sender!, address recipient!, uint256 amount!) public virtual override returns (bool) {
304      _transfer(sender!, recipient!, amount!);
305
306      uint256 currentAllowance = allowances[sender!][_msgSender()];
307      require(currentAllowance >= amount!, "ERC20: transfer amount exceeds allowance");
308      unchecked {
309          _approve(sender!, _msgSender(), currentAllowance - amount!);
310      }
311
312      return true;
313  }
314
  
```

Outdated Compiler Version.

| | | | | |
|---------|-----------|---------|-----------|---|
| Item: 1 | Location: | Line 15 | Severity: |  Low |
|---------|-----------|---------|-----------|---|

| | |
|-------------------|--|
| Function | Using an outdated compiler version can be problematic especially if there are publicly disclosed bugs and issues that affect the current compiler version. The following outdated versions were detected: /dvpn.sol - ^0.8.21 |
| Remedation | It is recommended to use a recent version of the Solidity compiler that should not be the most recent version, and it should not be an outdated version as well. Using very old versions of Solidity prevents the benefits of bug fixes and newer security checks. Consider using the solidity version v0.8.25, which patches most solidity vulnerabilities. |

⚠️ Incorrect Access Control (1 Item)

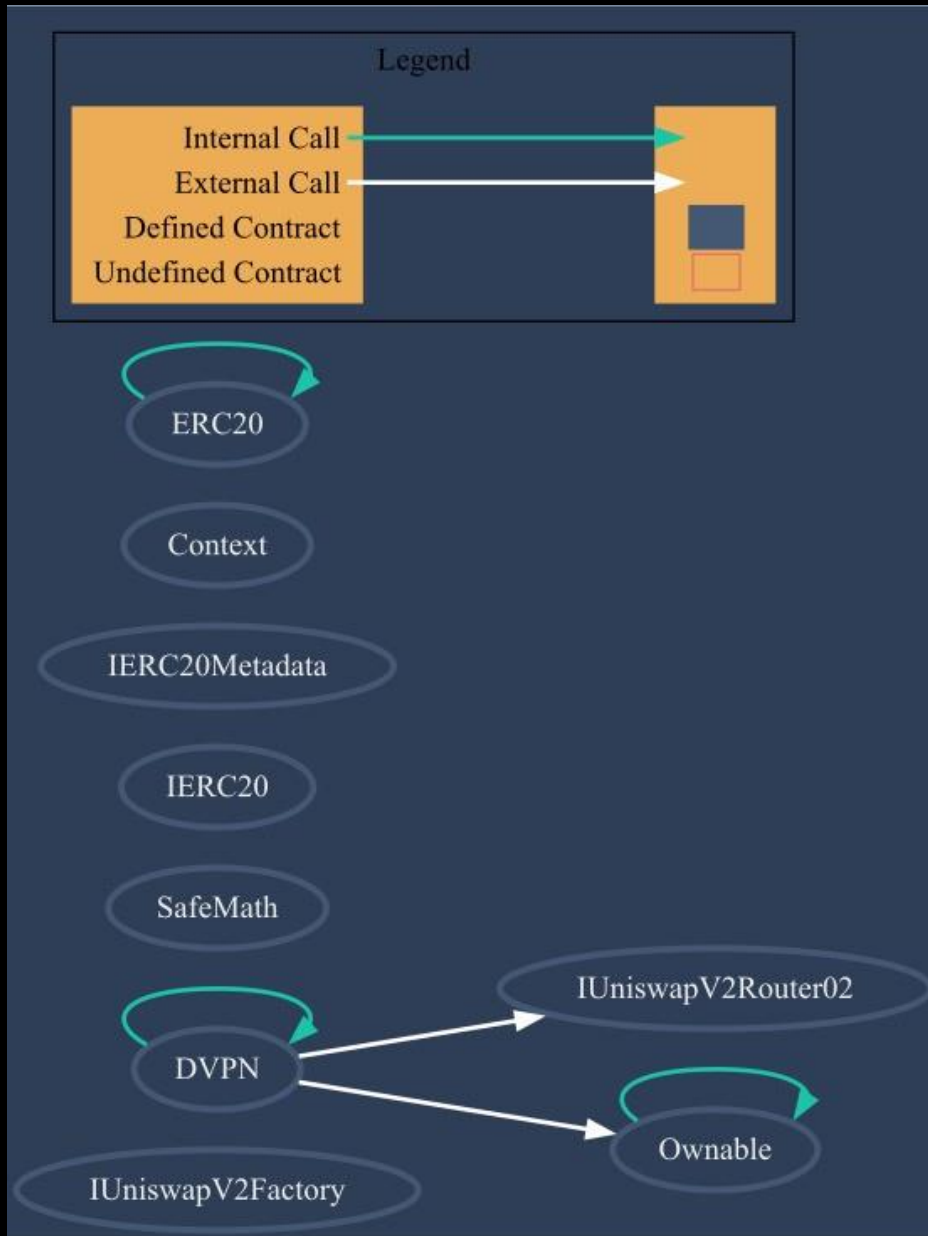
| | | | | |
|---------|-----------|--------------|-----------|----------|
| Item: 1 | Location: | Line 539-542 | Severity: | ■ Medium |
|---------|-----------|--------------|-----------|----------|

| | |
|--------------------|---|
| Function | <p>Access control plays an important role in segregation of privileges in smart contracts and other applications. If this is misconfigured or not properly validated on sensitive functions, it may lead to loss of funds, tokens and in some cases compromise of the smart contract.</p> <p>The contract DVPN is importing an access control library @openzeppelin/contracts/access/Ownable.sol but the function manualsend is missing the modifier onlyOwner.</p> |
| Remediation | <ol style="list-style-type: none"> 1. Consider adding access control modifiers to the function to Ensure that initialization functions can only be called once and only by authorized entities. 2. Implement least-privilege roles using libraries like OpenZeppelin's Access Control. 3. Add proper access control modifiers to sensitive functions, such as onlyOwner or custom roles. |

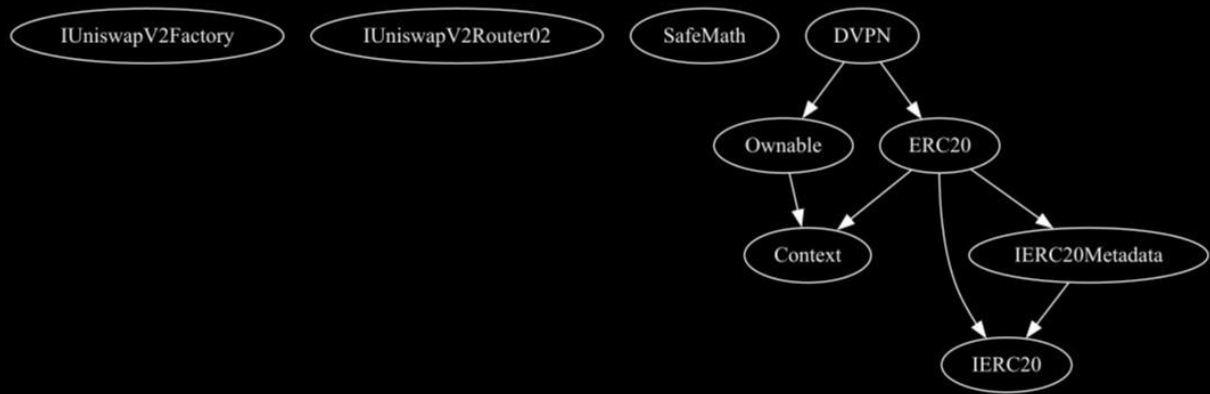
```

539      ftrace | funcSig
540      function manualsend() external {
541          bool success;
542          (success,) = address(marketingWallet).call{value: address(this).balance}("");
543      }
544      ftrace | funcSig
  
```








































Contract Interaction Graph





































Inheritance Graph


















Contract Functions

| Contract | Type | Bases | | |
|--------------------|---|--|---|--|
| L | Function Name | Visibility | Mutability | Modifiers |
| IUniswapV2Factory | Interface | | | |
| L | feeTo | External  | | NO  |
| L | feeToSetter | External  | | NO  |
| L | getPair | External  | | NO  |
| L | allPairs | External  | | NO  |
| L | allPairsLength | External  | | NO  |
| L | createPair | External  |  | NO  |
| L | setFeeTo | External  |  | NO  |
| L | setFeeToSetter | External  |  | NO  |
| IUniswapV2Router02 | Interface | | | |
| L | factory | External  | | NO  |
| L | WETH | External  | | NO  |
| L | addLiquidity | External  |  | NO  |
| L | addLiquidityETH | External  |  | NO  |
| L | swapExactTokensForTokensSupportingFeeOnTransferTokens | External  |  | NO  |
| L | swapExactETHForTokensSupportingFeeOnTransferTokens | External  |  | NO  |
| L | swapExactTokensForETHSupportingFeeOnTransferTokens | External  |  | NO  |

| SafeMath | Library | | | |
|-----------------------|--------------|--|---|--|
| L | tryAdd | Internal  | | |
| L | trySub | Internal  | | |
| L | tryMul | Internal  | | |
| L | tryDiv | Internal  | | |
| L | tryMod | Internal  | | |
| L | add | Internal  | | |
| L | sub | Internal  | | |
| L | mul | Internal  | | |
| L | div | Internal  | | |
| L | mod | Internal  | | |
| L | sub | Internal  | | |
| L | div | Internal  | | |
| L | mod | Internal  | | |
| | | | | |
| IERC20 | Interface | | | |
| L | totalSupply | External  | | NO  |
| L | balanceOf | External  | | NO  |
| L | transfer | External  |  | NO  |
| L | allowance | External  | | NO  |
| L | approve | External  |  | NO  |
| L | transferFrom | External  |  | NO  |
| | | | | |
| IERC20Metadata | Interface | IERC20 | | |
| L | name | External  | | NO  |
| L | symbol | External  | | NO  |
| L | decimals | External  | | NO  |
| | | | | |

| Context | Implementation | | | |
|---------|--------------------|---------------------------------|--|-----------|
| L | _msgSender | Internal | | |
| L | _msgData | Internal | | |
| Ownable | Implementation | Context | | |
| L | | Public | | NO |
| L | owner | Public | | NO |
| L | renounceOwnership | Public | | onlyOwner |
| L | transferOwnership | Public | | onlyOwner |
| L | _transferOwnership | Internal | | |
| ERC20 | Implementation | Context, IERC20, IERC20Metadata | | |
| L | | Public | | NO |
| L | name | Public | | NO |
| L | symbol | Public | | NO |
| L | decimals | Public | | NO |
| L | totalSupply | Public | | NO |
| L | balanceOf | Public | | NO |
| L | transfer | Public | | NO |
| L | allowance | Public | | NO |
| L | approve | Public | | NO |
| L | transferFrom | Public | | NO |
| L | increaseAllowance | Public | | NO |
| L | decreaseAllowance | Public | | NO |
| L | _transfer | Internal | | |

| | | | | |
|-------------|-----------------------------|----------------|---|-----------|
| L | _mint | Internal 🔒 | 🔒 | |
| L | _burn | Internal 🔒 | 🔒 | |
| L | _approve | Internal 🔒 | 🔒 | |
| L | _beforeTokenTransfer | Internal 🔒 | 🔒 | |
| L | _afterTokenTransfer | Internal 🔒 | 🔒 | |
| DVPN | Implementation | ERC20, Ownable | | |
| L | | Public 🔓 | 🔒 | ERC20 |
| L | | External 🔓 | 🔒 | NO 🔓 |
| L | AddLP | External 🔓 | 🔒 | onlyOwner |
| L | startTrading | External 🔓 | 🔒 | onlyOwner |
| L | removeLimits | External 🔓 | 🔒 | onlyOwner |
| L | updateSwapTokensAtAmount | External 🔓 | 🔒 | onlyOwner |
| L | updateMaxSwap | External 🔓 | 🔒 | onlyOwner |
| L | updateMaxTxnAmount | External 🔓 | 🔒 | onlyOwner |
| L | updateMaxWalletAmount | External 🔓 | 🔒 | onlyOwner |
| L | whitelistContract | Public 🔓 | 🔒 | onlyOwner |
| L | excludeFromMaxTransaction | Public 🔓 | 🔒 | onlyOwner |
| L | updateSwapEnabled | External 🔓 | 🔒 | onlyOwner |
| L | excludeFromFees | Public 🔓 | 🔒 | onlyOwner |
| L | manualswap | External 🔓 | 🔒 | NO 🔓 |
| L | manualsend | External 🔓 | 🔒 | NO 🔓 |
| L | setAutomatedMarketMakerPair | Public 🔓 | 🔒 | onlyOwner |

| | | | | |
|---|------------------------------|--|---|--|
| L | _setAutomatedMarketMakerPair | Private  |  | |
| L | updateFees | External  |  | onlyOwner |
| L | updateMarketingWallet | External  |  | onlyOwner |
| L | airdrop | External  |  | NO  |
| L | _transfer | Internal  |  | |
| L | swapTokensForEth | Private  |  | |
| L | swapBack | Private  |  | |



Function
can modify
state



Function
is payable

Audit Scope

Audit Method.

Our smart contract audit is an extensive methodical examination and analysis of the smart contract's code that is used to interact with the blockchain. Goal: discover errors, issues and security vulnerabilities in the code. Findings getting reported and improvements getting suggested.

Automatic and Manual Review

We are using automated tools to scan functions and weaknesses of the contract. Transfers, integer over-undeflow checks such as all CWE events.

Tools we use:

Visual Studio Code

CWE

SWC

Solidity Scan

SVD

In manual code review our auditor looking at source code and performing line by line examination. This method helps to clarify developer's coding decisions and business logic.

Skeleton Ecosystem

<https://skeletonecosystem.com>

<https://github.com/SkeletonEcosystem/Audits>

