THE ARCADIA GROUP

AUDIT

arcadiamgroup.com

Table of Contents

Executive Summary

Arcadia was engaged on April 7th with ProPane Finance to review their token implementation on Binance Smart Chain. An initial notation of findings was submitted on April 12th, with findings being reviewed and initial remediation being set on April 13th, with additional remediation being provided on April 19th.

Findings

1. When buying or withdrawing liquidity, there is probability swapAndLiquify will be triggered

PANE-1
Severity: Medium
Impact: Medium

Contract: Pane Token
Finding Type: Dynamic
Status: Resolved

```
// is the token balance of this contract address over the min number of
// tokens that we need to initiate a swap + liquidity lock?
// also, don't get caught in a circular liquidity event.
// also, don't swap & liquify if sender is uniswap pair.
uint256 contractTokenBalance = balanceOf(address(this));
bool overMinTokenBalance = contractTokenBalance >= minTokensBeforeSwap;
if (
    overMinTokenBalance &&
    !inSwapAndLiquify &&
    msg.sender != uniswapV2Pair &&
    swapAndLiquifyEnabled
) {
    swapAndLiquify(contractTokenBalance);
}
```

The intention of this code is to avoid swapAndLiquify when the sender is UniswapPair this happens in buying and withdrawing liquidity.

Action Recommended:

The functionality should be replaced by *from*

```
uint256 contractTokenBalance = balanceOf(address(this));
bool overMinTokenBalance = contractTokenBalance >= minTokensBeforeSwap;
if (
    overMinTokenBalance &&
    !inSwapAndLiquify &&
    from != uniswapV2Pair &&
    swapAndLiquifyEnabled
) {
    swapAndLiquify(contractTokenBalance);
}
```

2. Can't withdraw from AMM (Pancake/Uniswap)

PANE-2
Severity: Critical
Impact: HIgh

Contract: Pane.sol
Finding Type: Dynamic
Status: Resolved

The PaneToken transfer function is implemented in order to create the following intended functionality:

- 1% will be deducted and be sent to the treasury wallet
- a transfer fee will be deducted and used as pool liquidity (with ETH).
- The remaining will be sent to the receiver.

The implementation is done in the PaneToken transfer function as follows:

- Swap and liquify the transfer fee balance if large enough.
- If the sender is not the owner, calculate and send
    - The transfer fee to Pane contract address
    - 1% amount to treasury wallet
    - Remaining to receiver

This causes in certain situations:

- Users can't withdraw liquidity from AMM because of circular transfer calls.
- If there is no liquidity on AMM, users can't transfer their token.

```
// calculate the number of tokens to take as a fee
uint256 tokensToLock = calculateTokenFee(
    amount⬆,
    feeDecimals,
    feePercentage
);

// 1% treasury tax
uint256 treasuryVal = amount⬆.div(100);

// if(from != address(owner()) && from != uniswapV2Pair) {
if(from⬆ != address(owner())) {
    // take the fee and send those tokens to this contract address
    // and then send the remainder of tokens to original recipient
    uint256 tokensToTransfer = amount⬆.sub(tokensToLock).sub(treasuryVal);

    super._transfer(from⬆, address(this), tokensToLock);
    super._transfer(from⬆, to⬆, tokensToTransfer);
    super._transfer(from⬆, treasuryWallet, treasuryVal);
} else {
    super._transfer(from⬆, to⬆, amount⬆);
}
```

Action Recommended:

1. Implement whitelist and add owner, AMM pairs
2. Add liquidity with ETH right after deploying the token.

3. Implement Whitelist

PANE-3
Severity: Recommendation
Impact: Recommendation

Contract: Pane.sol
Finding Type: Dynamic
Status: Resolved

Declare

```
mapping (address => bool) private whitelist;
```

Init value

```
constructor(
    IUniswapV2Router02 _uniswapV2Router,
    uint8 _feeDecimals,
    uint32 _feePercentage,
    uint128 _minTokensBeforeSwap,
    bool _swapAndLiquifyEnabled,
    address _treasuryWallet
) public ERC20("Propane", "Pane") {
    // mint tokens which will initially belong to deployer
    // deployer should go seed the pair with some initial liquidity
    _mint(msg.sender, 20000 * 10**18);

    // Create a uniswap pair for this new token
    uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory())
        .createPair(address(this), _uniswapV2Router.WETH());

    whitelist[owner()] = true;
    whitelist[address(uniswapV2Pair)] = true;
    whitelist[address(this)] = true;
```

Add utils functions

```
function toWhitelist(address account) public onlyOwner {
    whitelist[account] = true;
}

function outWhitelist(address account) public onlyOwner {
    whitelist[account] = false;
}
```

Update the _transfer code

```
    uint256 treasuryVal = amount.div(100);

    if(whitelist[from] || whitelist[to]) {
      super._transfer(from, to, amount);
    } else {
       uint256 tokensToTransfer = amount.sub(tokensToLock).sub(treasuryVal);

       super._transfer(from, address(this), tokensToLock);
       super._transfer(from, to, tokensToTransfer);
       super._transfer(from, treasuryWallet, treasuryVal);
    }
```

Disclaimer

While best efforts and precautions have been taken in the preparation of this document, The Arcadia Group and the Authors assume no responsibility for errors, omissions, or damages resulting from the use of the provided information. Additionally, Arcadia would like to emphasize that the use of Arcadia's services does not guarantee the security of a smart contract or set of smart contracts and does not guarantee against attacks. One audit on its own is not enough for a project to be considered secure; that categorization can only be earned through extensive peer review and battle testing over an extended period.

Document Information

Title	PANE Token Audit
Client	ProPane Finance
Auditor(s)	Anhnt
Reviewed by	Joel Farris
Approved by	Rasikh Morani
Contact Details	Rasikh Morani (972) 543-3886 rasikh@arcadiamgroup.com https://t.me/thearcadiagroup