Arrakis Finance v2 Follow up Audit Report

Nov 17, 2022

WATCHPUG

Table of Contents

Summary

This report has been prepared for Arrakis Finance v2 Follow up Audit Report smart contract, to discover issues and vulnerabilities in the source code of their Smart Contract as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- 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.

Overview

Project Summary

Project Name	**Arrakis Finance v2 Follow up Audit Report**
Codebase	**https://github.com/ArrakisFinance/vault-v2-core**
Commit	**376bfcec803f0644fdc601db3a5772d2179c13a0**
Language	**Solidity**

Audit Summary

Delivery Date	**Nov 17, 2022**
Audit Methodology	**Static Analysis, Manual Review**
Total Isssues	**9**

[WP-L1] `SetGelatoFeeCollector` Unnecessary/misleading parameter `vault`

Low

Issue Description

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMManagerStorage.sol#L205-L219

```
205  function setGelatoFeeCollector(address payable gelatoFeeCollector_)
206      external
207      override
208      whenNotPaused
209      onlyOwner
210  {
211      require(
212          gelatoFeeCollector != gelatoFeeCollector_,
213          "PALMManager: gelatoFeeCollector"
214      );
215      emit SetGelatoFeeCollector(
216          address(this),
217          gelatoFeeCollector = gelatoFeeCollector_
218      );
219  }
```

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/interfaces/IPALMManager.sol#L28-L31

```
28  event SetGelatoFeeCollector(
29      address indexed vault,
30      address gelatoFeeCollector
31  );
```

`gelatoFeeCollector` is a global configuration unrelated to a specific vault, there is no need to emit `SetGelatoFeeCollector` events with a `vault` parameter.

Recommendation

Consider removing the `vault` parameter from `SetGelatoFeeCollector` .

Status

✓ Fixed

[WP-L2] `requireIsOwnerOrDelegate` Flawed implementation

Low

Issue Description

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMTermsStorage.sol#L63-L73

```
modifier requireIsOwnerOrDelegate(address vault_) {
    address delegate = delegateByVaults[vault_];
    if (delegate != address(0))
        require(msg.sender == delegate, "PALMTerms: no delegate");
    else
        require(
            _vaults[msg.sender].contains(vault_),
            "PALMTerms: not owner"
        );
    _;
}
```

The current implementation of `requireIsOwnerOrDelegate` is actually `requireDelegateWhenDelegateExistsOtherwiseRequireIsOwner` , which is unconventional and mismatches the name of the modifier.

Recommendation

```
modifier requireIsOwnerOrDelegate(address vault_) {
    require(isOwnerOrDelegate(vault_, msg.sender), "PALMTerms: no owner and delegate");
    _;
}

function isOwnerOrDelegate(address vault_, address account) public view returns (bool) {
    if (_vaults[account].contains(vault_)) {
        return true;
    } else {
        address delegate = delegateByVaults[vault_];
        if (delegate == address(0)) {
            return false;
        }
        return account == delegate;
    }
}
```

Status

✓ Fixed

[WP-L3] Typo

Low

Issue Description

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMManagerStorage.sol#L194-L203

```
function setVaultStraByName(address vault_, string calldata strat_)
    external
    override
    whenNotPaused
    requireAddressNotZero(vault_)
    onlyVaultOwner(vault_)
    onlyManagedVaults(vault_)
{
    _setVaultStrat(vault_, keccak256(abi.encodePacked(strat_)));
}
```

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMTermsStorage.sol#L207-L222

```
function setVaultStratByName(address vault_, string calldata strat_)
    external
    override
    requireAddressNotZero(vault_)
    requireIsOwnerOrDelegate(vault_)
{
    IPALMManager(manager).setVaultStraByName(vault_, strat_);

    emit LogSetVaultStratByName(
        delegateByVaults[vault_] != address(0)
            ? delegateByVaults[vault_]
            : msg.sender,
        vault_,
        strat_
        );
    }
```

`setVaultStraByName` -> `setVaultStratByName`

Status

✓ **Fixed**

[WP-L4] `LogWithdrawVaultBalance` Misleading/wrong value for parameter `sentBack`

Low

Issue Description

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMTermsStorage.sol#L234-L244

```
234  function withdrawVaultBalance(
235      address vault_,
236      uint256 amount_,
237      address payable to_
238  ) external override requireAddressNotZero(vault_) requireIsOwner(vault_) {
239      IPALMManager manager_ = IPALMManager(manager);
240      (uint256 balance, , , , ) = manager_.vaults(vault_);
241      manager_.withdrawVaultBalance(vault_, amount_, to_);
242
243      emit LogWithdrawVaultBalance(msg.sender, vault_, to_, balance);
244  }
```

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/interfaces/IPALMTerms.sol#L82-L87

```
82  event LogWithdrawVaultBalance(
83      address creator,
84      address vault,
85      address to,
86      uint256 sentBack
87  );
```

The `sentBack` parameter in `LogWithdrawVaultBalance` sounds like the amount being sent back to the user, ie, the withdrawal amount, but actually, it's the balance BEFORE the withdrawal.

This is wrong or at least misleading.

Recommendation

Consider renaming the `sentBack` parameter to `balanceBefore` or change the value to `amount_` in `withdrawVaultBalance()` .

Status

✓ Fixed

[WP-G5] Copying the whole `vault` to memory is unnecessary and gas expensive

Gas

Issue Description

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/PALMManager.sol#L41-L58

```
function _preExec(address vault_, uint256 feeAmount_)
    internal
    returns (uint256 balance)
{
    VaultInfo memory vaultInfo = vaults[vault_];
    require(
        vaultInfo.balance >= feeAmount_,
        "PALMManager: Not enough balance to pay fee"
    );
    balance = vaultInfo.balance - feeAmount_;

    // update lastRebalance time
    // solhint-disable-next-line not-rely-on-time
    vaults[vault_].lastRebalance = block.timestamp;
    vaults[vault_].balance = balance;

    Address.sendValue(gelatoFeeCollector, feeAmount_);
}
```

Recommendation

```
function _preExec(address vault_, uint256 feeAmount_)
    internal
    returns (uint256 balance)
{
    uint256 vaultBalance = vaults[vault_].balance;
    require(
       vaultBalance >= feeAmount_,
        "PALMManager: Not enough balance to pay fee"
    );
    balance = vaultBalance - feeAmount_;

    // update lastRebalance time
    // solhint-disable-next-line not-rely-on-time
    vaults[vault_].lastRebalance = block.timestamp;
    vaults[vault_].balance = balance;

    Address.sendValue(gelatoFeeCollector, feeAmount_);
}
```

Status

✓ Fixed

[WP-G6] Using `EnumerableSet.AddressSet` can avoid unnecessary storage reads and simplify the code

Gas

Issue Description

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMManagerStorage.sol#L65

```
    address[] public operators;
```

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMManagerStorage.sol#L221-L234

```
    function addOperators(address[] calldata operators_)
        external
        override
        whenNotPaused
        onlyOwner
    {
        for (uint256 i = 0; i < operators_.length; i++) {
            (bool isOperator, ) = _isOperator(operators_[i]);
            require(!isOperator, "PALMManager: operator");
            operators.push(operators_[i]);
        }

        emit AddOperators(address(this), operators_);
    }
```

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMManagerStorage.sol#L236-L243

```
    function removeOperators(address[] calldata operators_)
        external
        override
        whenNotPaused
        onlyOwner
    {
        _removeOperators(operators_);
    }
```

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMManagerStorage.sol#L404-L413

```
    function _removeOperators(address[] memory operators_) internal {
        for (uint256 i = 0; i < operators_.length; i++) {
            (bool isOperator, uint256 index) = _isOperator(operators_[i]);
            require(isOperator, "PALMManager: no operator");

            delete operators[index];
        }

        emit RemoveOperators(address(this), operators_);
    }
```

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMManagerStorage.sol#L430-L440

```
    function _isOperator(address operator_)
        internal
        view
        requireAddressNotZero(operator_)
        returns (bool, uint256)
    {
        for (uint256 index = 0; index < operators.length; index++) {
            if (operators[index] == operator_) return (true, index);
        }
        return (false, 0);
    }
```

Recommendation

```
    function addOperators(address[] calldata operators_)
        external
        override
        whenNotPaused
        onlyOwner
    {
        for (uint256 i = 0; i < operators_.length; ++i) {
            require(operators.add(operators_[i]), "PALMManager: operator");
        }

        emit AddOperators(address(this), operators_);
    }
```

```
    function removeOperators(address[] calldata operators_)
        external
        override
        whenNotPaused
        onlyOwner
    {
        for (uint256 i = 0; i < operators_.length; ++i) {
            require(operators.remove(operators_[i]), "PALMManager: no operator");
        }

        emit RemoveOperators(address(this), operators_);
    }
```

```
    function _isOperator(address operator_)
        internal
        view
        requireAddressNotZero(operator_)
        returns (bool)
    {
        return operators.contains(operator_);
    }
```

Status

✓ **Fixed**

[WP-I7] `SetTermEnd` Misleading event parameter names

Informational

Issue Description

The parameters in the `SetTermEnd` event is named `**Duration`, while the actual values are the `termEnd` timestamps:

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/interfaces/IPALMManager.sol#L33-L37

```
33  event SetTermEnd(
34      address indexed vault,
35      uint256 oldtermDuration,
36      uint256 newtermDuration
37  );
```

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMManagerStorage.sol#L284-L299

```
285 function renewTerm(address vault_)
286     external
287     override
288     whenNotPaused
289     onlyPALMTerms
290     requireAddressNotZero(vault_)
291     onlyManagedVaults(vault_)
292 {
293     emit SetTermEnd(
294         vault_,
295         vaults[vault_].termEnd,
296         // solhint-disable-next-line not-rely-on-time
297         vaults[vault_].termEnd = block.timestamp + termDuration
298     );
299 }
```

Recommendation

Change to:

```
event SetTermEnd(
    address indexed vault,
    uint256 oldtermEnd,
    uint256 newtermEnd
);
```

Status

✓ Fixed

[WP-G8] `subtractAdminFeesOnAmounts` can be optimized

Gas

Issue Description

https://github.com/ArrakisFinance/vault-v2-core/blob/376bfcec803f0644fdc601db3a5772d2179c13a0/contracts/libraries/Underlying.sol#L26-L84

```
26 function totalUnderlyingWithFees(
27         UnderlyingPayload memory underlyingPayload_
28     )
29         public
30         view
31         returns (
32             uint256 amount0,
33             uint256 amount1,
34             uint256 fee0,
35             uint256 fee1
36         )
37     {
38         for (uint256 i = 0; i < underlyingPayload_.ranges.length; i++) {
39             {
40                 IUniswapV3Pool pool = IUniswapV3Pool(
41                     underlyingPayload_.factory.getPool(
42                         underlyingPayload_.token0,
43                         underlyingPayload_.token1,
44                         underlyingPayload_.ranges[i].feeTier
45                     )
46                 );
47                 (uint256 a0, uint256 a1, uint256 f0, uint256 f1) = underlying(
48                     RangeData({
49                         self: underlyingPayload_.self,
50                         range: underlyingPayload_.ranges[i],
51                         pool: pool
52                     })
53                 );
54                 amount0 += a0 + f0;
55                 amount1 += a1 + f1;
56                 fee0 += f0;
57                 fee1 += f1;
58             }
```

```
        }

        IArrakisV2 arrakisV2 = IArrakisV2(underlyingPayload_.self);

        (amount0, amount1) = subtractAdminFeesOnAmounts(
            fee0,
            fee1,
            Manager.getManagerFeeBPS(arrakisV2.manager()),
            arrakisV2.arrakisFeeBPS(),
            amount0,
            amount1
        );

        amount0 +=
            IERC20(underlyingPayload_.token0).balanceOf(
                underlyingPayload_.self
            ) -
            arrakisV2.managerBalance0() -
            arrakisV2.arrakisBalance0();
        amount1 +=
            IERC20(underlyingPayload_.token1).balanceOf(
                underlyingPayload_.self
            ) -
            arrakisV2.managerBalance1() -
            arrakisV2.arrakisBalance1();
    }
```

https://github.com/ArrakisFinance/vault-v2-core/blob/376bfcec803f0644fdc601db3a5772d2179c13a0/contracts/libraries/Underlying.sol#L162-L192

```
    function subtractAdminFees(
        uint256 rawFee0_,
        uint256 rawFee1_,
        uint16 managerFeeBPS_,
        uint16 arrakisFeeBPS_
    ) public pure returns (uint256 fee0, uint256 fee1) {
        fee0 =
            rawFee0_ -
            ((rawFee0_ * (managerFeeBPS_ + arrakisFeeBPS_)) / 10000);
        fee1 =
            rawFee1_ -
            ((rawFee1_ * (managerFeeBPS_ + arrakisFeeBPS_)) / 10000);
    }

    function subtractAdminFeesOnAmounts(
        uint256 rawFee0_,
        uint256 rawFee1_,
        uint16 managerFeeBPS_,
        uint16 arrakisFeeBPS_,
        uint256 amount0_,
        uint256 amount1_
    ) public pure returns (uint256 amount0, uint256 amount1) {
        (uint256 fee0, uint256 fee1) = subtractAdminFees(
            rawFee0_,
            rawFee1_,
            managerFeeBPS_,
            arrakisFeeBPS_
        );
        amount0 = amount0_ - (rawFee0_ - fee0);
        amount1 = amount1_ - (rawFee1_ - fee1);
    }
```

Recommendation

It’s unnecessary to manipulate the value of `fee0` and `fee1` back and forth.

```
function totalUnderlyingWithFees(
        UnderlyingPayload memory underlyingPayload_
    )
        public
        view
        returns (
            uint256 amount0,
            uint256 amount1,
            uint256 fee0,
            uint256 fee1
        )
    {
        for (uint256 i = 0; i < underlyingPayload_.ranges.length; i++) {
            {
                IUniswapV3Pool pool = IUniswapV3Pool(
                    underlyingPayload_.factory.getPool(
                        underlyingPayload_.token0,
                        underlyingPayload_.token1,
                        underlyingPayload_.ranges[i].feeTier
                    )
                );
                (uint256 a0, uint256 a1, uint256 f0, uint256 f1) = underlying(
                    RangeData({
                        self: underlyingPayload_.self,
                        range: underlyingPayload_.ranges[i],
                        pool: pool
                    })
                );
                amount0 += a0;
                amount1 += a1;
                fee0 += f0;
                fee1 += f1;
            }
        }

        IArrakisV2 arrakisV2 = IArrakisV2(underlyingPayload_.self);

        (fee0After, fee1After) = subtractAdminFees(
            fee0,
            fee1,
            Manager.getManagerFeeBPS(arrakisV2.manager()),
            arrakisV2.arrakisFeeBPS(),
        );

        amount0 +=
            fee0After +
            IERC20(underlyingPayload_.token0).balanceOf(
                underlyingPayload_.self
            ) -
            arrakisV2.managerBalance0() -
            arrakisV2.arrakisBalance0();
        amount1 +=
            fee1After +
            IERC20(underlyingPayload_.token1).balanceOf(
                underlyingPayload_.self
            ) -
            arrakisV2.managerBalance1() -
            arrakisV2.arrakisBalance1();
    }
```

Status

✓ **Fixed**

[WP-N9] Unnecessary `manager` parameter in the events

Issue Description

1. AddOperators

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/interfaces/IPALMManager.sol#L22

```
event AddOperators(address indexed manager, address[] operators);
```

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMManagerStorage.sol#L221-L234

```
function addOperators(address[] calldata operators_)
    external
    override
    whenNotPaused
    onlyOwner
{
    for (uint256 i = 0; i < operators_.length; i++) {
        (bool isOperator, ) = _isOperator(operators_[i]);
        require(!isOperator, "PALMManager: operator");
        operators.push(operators_[i]);
    }

    emit AddOperators(address(this), operators_);
}
```

2. RemoveOperators

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/interfaces/IPALMManager.sol#L24

```
event RemoveOperators(address indexed manager, address[] operators);
```

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMManagerStorage.sol#L404-L413

```
404     function _removeOperators(address[] memory operators_) internal {
405         for (uint256 i = 0; i < operators_.length; i++) {
406             (bool isOperator, uint256 index) = _isOperator(operators_[i]);
407             require(isOperator, "PALMManager: no operator");
408 
409             delete operators[index];
410         }
411 
412         emit RemoveOperators(address(this), operators_);
413     }
```

3. WhitelistStrat

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/interfaces/IPALMManager.sol#L20

```
 20 event WhitelistStrat(address indexed manager, string strat);
```

https://github.com/ArrakisFinance/v2-palm//blob/e0a7c7b0474a5f492a5551b52564827e26989382/contracts/abstracts/PALMManagerStorage.sol#L301-L318

```
301 function whitelistStrat(string calldata strat_)
302         external
303         whenNotPaused
304         onlyOwner
305     {
306         bytes32 stratB32 = keccak256(abi.encodePacked(strat_));
307         require(
308             stratB32 != keccak256(abi.encodePacked("")),
309             "PALMManager: empty string"
310         );
311         require(
312             !_whitelistedStrat.contains(stratB32),
            "PALMManager: strat whitelisted."
        );
        _whitelistedStrat.add(stratB32);

        emit WhitelistStrat(address(this), strat_);
    }
```

Recommendation

Consider removing the `manager` parameter form these events:

```
event AddOperators(address[] operators);
```

```
event RemoveOperators(address[] operators);
```

```
event WhitelistStrat(string strat);
```

Status

✓ Fixed

Appendix

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by WatchPug; however, WatchPug does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication.

Disclaimer

This report is based on the scope of materials and documentation provided for a limited review at the time provided. Results may not be complete nor inclusive of all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Smart Contract technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. A report does not indicate the endorsement of any particular project or team, nor guarantee its security. No third party should rely on the reports in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.