SET V2
Smart Contracts. Phase 1. Final Review

Mikhail Vladimirov and Dmitry Khovratovich

25th August 2020

This document describes the audit process of the SET V2 smart contracts, phase 1, performed by ABDK Consulting.

1. Introduction

We've been asked to review the Set V2 smart contracts given in separate files.

2. BasicIssuanceModule

In this section we describe issues found in the BasicIssuanceModule.sol (originally IssuanceModule.sol).

2.1 Fixed Major Flaws

This section lists major flaws, which were found in the smart contract.

1. Line 146: the calling of the `executeIssuanceDeposit` may internally increase position balances. It happens not just by sending component tokens to Set token smart contracts, but in a more profitable way for the caller, for example by repaying a flash loan. This will allow the caller to use both component tokens in some profitable way and also the same component tokens to issue Set tokens. The `nonReentrant` modifier doesn't help here, as internal transactions may interact with the same token through some other module. Here is the negative scenario:
    - Attacker invokes specially-crafted smart contract A.
    - A obtains a flash loan of component tokens from a Set token smart contract.
    - A invokes issueFromContract on issuance module, providing A's address as `_issueContract`.
    - The issuance module calls the `executeIssuanceDeposit` function on A.
    - A repays flash loan, thus increasing component balances of the Set token.
    - A returns control to the issuance module.

- The Issuance module verifies that component balances were increased, mints Set tokens and send them to the attacker.

2. Line 148: there is no guarantee that the Set still has the same components here as it had two lives above, so comparing positions with components addresses could lead to unexpected results.
3. Line 150: the `validateExpectedComponentIncrease` check doesn't take the `int` account that Set token may have several positions for the same component.

All were resolved.

3. PriceOracle

No security significant issues were found in PriceOracle.sol.

4. InvokeLib

In this section we describe issues found in the InvokeLib.sol

4.1 Moderate Flaws

This section lists moderate flaws, which were found in the smart contract.

1. Line 55, 77: the returned value is ignored. Unsuccessful calls may be treated as a successful one.

5. Controller

In this section we describe issues found in the Controller.sol.

5.1 Moderate Flaws

This section lists moderate flaws, which were found in the smart contract.

1. Line164: the same ID could be specified for several different resources. The contract will map such ID only to the latest such resource, leaving all other resources with no IDs assigned. Consider checking that `resourceId[_resourceIds[i]]` is zero before setting it.
2. Line 256, 316, 329, 351, 366: the next lines is deadly inefficient:
   - `sets = sets.append(_setToken)`
   - `factories = factories.append(_factory)`
   - `modules = modules.append(_module)`
   - `modules = modules.remove(_module)`
   - `resources = resources.append(_resource)`

   They read the `registered sets`, the `factories`,the `modules` and the `resources` from storage into memory, then copy them from one in-memory array into another, appending new `registered sets`, `factories`, `modules` and `resources`, and then write all `registered sets`, `factories`, `modules` and `resources` back into storage. Efficient way to remove an element from a storage array is to copy the last array element into the slot occupied by the element that ought to be removed, and then decrease array length by one.

All were resolved.

3. Line 301, 331, 370: the `isFactory`, the `isModule` and the `isResource` functions remove only the first occurrence of given factory from factories array, other occurrences may still exist, so setting to false the `isFactory[_factory]`, the `isModule[_module]` and the `isResource[resourceToRemove]` could make the contract's state inconsistent.

6. SetToken

In this section we describe issues found in the SetToken.sol.

6.1 Fixed Moderate Flaws

This section lists moderate flaws, which were found in the smart contract.

1. Line 154: it is possible to specify the same component address several times, which will lead to a Set token with several positions sharing the same component address. This Set token will not work properly.

Found to be nonapplicable.

2. Line 213: perhaps, there should be `positions.length` instead of `positions.length.add(1)`.

Fixed.

7. StreamingFeeModule

No security significant issues were found in the StreamingFeeModule.sol.

8. IntegrationRegistry

No security significant issues were found in the IntegrationRegistry.sol.

9. PositionLib

No security significant issues were found in the PositionLib.sol.

10. PreciseUnitMath

In this section we describe issues found in the PreciseUnitMath.sol.

10.1 Major Flaws

We have identified potentially faulty behaviour in the contract related to multiple-voting scenario:

Line 187: the condition returns 0 if a and b are zero, while for all other values of a, the function reverts on `b==0`.

Fixed by adding a revert.

11. AddressArrayUtils.sol

No security significant issues were found in the AddressArrayUtils.sol.

12. SetTokenCreator

In this section we describe issues found in the SetTokenCreator.sol.

12.1 Moderate issues

1. Line 81: it is not checked that `_components[i]` are unique. Consider adding the check.

Fixed by adding a check.

13. ISetToken.sol

No security significant issues were found in the ISetToken.sol.

14. ExplicitERC20

No security significant issues were found in the ExplicitERC20.sol.

15. ModuleBase

No security significant issues were found in the ModuleBase.sol.

16. IModule

No security significant issues were found in the IModule.sol.

17. IOracle

No security significant issues were found in the IOracle.sol.

18. IOracleAdapter

No security significant issues were found in the IOracleAdapter.sol.

19. IController

No security significant issues were found in the IController.sol.

20. IManagerIssuanceHook

No security significant issues were found in the IManagerIssuanceHook.sol.