Smart Contract
Security Audit Report

Table Of Contents

1 Executive Summary

On 2025.02.25, the SlowMist security team received the team's security audit application for HashKey ComplianceToken, developed the audit plan according to the agreement of both parties and the characteristics of the project, and finally issued the security audit report.
The SlowMist security team adopts the strategy of "white box lead, black, grey box assists" to conduct a complete security test on the project in the way closest to the real attack.

The test method information:

Test method	Description
Black box testing	Conduct security tests from an attacker's perspective externally.
Grey box testing	Conduct security testing on code modules through the scripting tool, observing the internal running status, mining weaknesses.
White box testing	Based on the open source code, non-open source code, to detect whether there are vulnerabilities in programs such as nodes, SDK, etc.

The vulnerability severity level information:

Level	Description
Critical	Critical severity vulnerabilities will have a significant impact on the security of the DeFi project, and it is strongly recommended to fix the critical vulnerabilities.
High	High severity vulnerabilities will affect the normal operation of the DeFi project. It is strongly recommended to fix high-risk vulnerabilities.
Medium	Medium severity vulnerability will affect the operation of the DeFi project. It is recommended to fix medium-risk vulnerabilities.
Low	Low severity vulnerabilities may affect the operation of the DeFi project in certain scenarios. It is suggested that the project team should evaluate and consider whether these vulnerabilities need to be fixed.
Weakness	There are safety risks theoretically, but it is extremely difficult to reproduce in engineering.
Suggestion	There are better practices for coding or architecture.

2 Audit Methodology

The security audit process of SlowMist security team for smart contract includes two steps:

- Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using automated analysis tools.
- Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that were considered during the audit of the smart contract:

NO.	Audit Class	Audit Subclass	Result
1	Overflow Audit	-	Passed
2	Reentrancy Attack Audit	-	Passed
3	Replay Attack Audit	-	Passed
4	Flashloan Attack Audit	-	Passed
5	Race Conditions Audit	Reordering Attack Audit	Passed
6	Permission Vulnerability Audit	Access Control Audit	Passed
		Excessive Authority Audit	Some Risk
7	Security Design Audit	External Module Safe Use Audit	Passed
		Compiler Version Security Audit	Passed
		Hard-coded Address Security Audit	Passed
		Fallback Function Safe Use Audit	Passed
		Show Coding Security Audit	Passed
		Function Return Value Security Audit	Passed
		External Call Function Security Audit	Passed

NO.	Audit Class	Audit Subclass	Result
7	Security Design Audit	External Call Function Security Audit	Passed
		Block data Dependence Security Audit	Passed
		tx.origin Authentication Security Audit	Passed
8	Denial of Service Audit	-	Passed
9	Gas Optimization Audit	-	Passed
10	Design Logic Audit	-	Passed
11	Variable Coverage Vulnerability Audit	-	Passed
12	"False Top-up" Vulnerability Audit	-	Passed
13	Scoping and Declarations Audit	-	Passed
14	Malicious Event Log Audit	-	Passed
15	Arithmetic Accuracy Deviation Audit	-	Passed
16	Uninitialized Storage Pointer Audit	-	Passed

3 Project Overview

3.1 Project Introduction

Bosera USD Money Market ETF, a Sub-Fund of the Bosera Global Exchange Traded Funds Series, is an actively managed ETF established under Hong Kong law. It offers listed and unlisted share classes, including tokenised and non-tokenised options. The tokenised Class of Shares, available only in the primary market, is not listed on the SEHK.

3.2 Information of the discovery

The following is the status of the vulnerabilities found in this audit:

NO.	Title	Category	Level	Status
N1	Risk of excessive authority	Authority Control Vulnerability Audit	Medium	Acknowledged
N2	Missing the event records	Others	Suggestion	Acknowledged
N3	Missing zero address check	Others	Suggestion	Fixed

3.3 Information of the Smart Contract

In accordance with the requirements of Circular on intermediaries engaging in tokenised securities-related activities, Part A, (b)(i), (d), the following is a description of the basic information of smart contracts:

Basic information of the smart contract					
Token total supply	Ownership management	Smart Contract code open source and verification	Programming language	Deployment network	Migratable
No limited	Ownership is managed by Gnosis safe multisig. Ownership_address	The code is open source and verified.	Solidity	HashKey Chain Mainnet	It can be migrated to other DLT networks or other contracts.

Note: In exceptional circumstances where a smart contract or DLT network becomes unavailable, the migrate function can be used to upgrade the smart contract or deploy it on another blockchain network to ensure uninterrupted service.

The following is an additional explanation of the smart contract, based on its implementation:

Additional Information	
Item	Explain
Source code verified	The contract has been deployed and tested in the test network. Can only be verified after a Smart Contract deployed in production. Source code will definitely be verified by SlowMist prior to Listing
Contract Bugs(e.g. miscalculation, nonce error, non-zero address verification...)	Refer to the SlowMist audit report. According to the SlowMist audit report, most of the bugs have passed

	with satisfaction (Section 2). The SlowMist audit identified one medium-risk concern and two suggestions. There are 3 bugs (section 3.2) mentioned. The development team is well aware of the issues and will address and fix them accordingly prior to project listing / go-live.
Proxy	Proxy is a single upgradable smart contract. It will adopt this unified proxy approach, allowing the smart contracts to be upgradable. This design enables seamless updates while maintaining the contract's state and functionality.
ForceTransfer requirements	Authorized controllers can force-transfer tokens through the forceTransfer function, which will relocate tokens between addresses while maintaining the total supply. Fund initiators have the highest authority over the Chairman's multi-signature private keys and may force-transfer tokens to secure addresses when significant security vulnerabilities occur. SlowMist confirms these force-transfer mechanisms are reasonable business functions aligned with high standards.
Mint requirements	Minter can issue tokens through the mint function, which will affect the total amount of tokens. SlowMist confirms the current minting functions are reasonable business functions aligned with high standards.
Burn requirements	The minter can perform the token redemption through the burn function. These operations will affect the total amount of token, but it will only burn the tokens in deadAddr, not directly burn the tokens of other holding addresses. SlowMist confirms these are reasonable business functions with high standards.
Transfer Limitations or Restrictions	SlowMist confirms that the ComplianceToken has comprehensive Transfer Limitations functionality to ensure regulatory compliance. The token implements a sophisticated multi-layered transfer restriction system that: Enforces Whitelisting Pairs: Users on the first whitelist can only transfer to their designated paired addresses on the second whitelist, ensuring KYC-verified transfers through pre-approved channels. Controls Fund Flow Paths: The system enforces strict fund flow paths where: Second whitelist addresses can only transfer to the deposit address. Withdrawal address can only transfer to first whitelist addresses. The deposit address can only transfer to the withdrawal address. The house address can only receive transfers from the deposit

	address. Burning transactions are limited to specific administrative addresses. Restriction Enforcement: All token transfers (both transfer and transferFrom functions) are validated against these restrictions before execution. Error Handling: Different restriction violations generate specific error codes for clear identification of compliance issues. Unrestricted Public Transfers: Standard transfers between non-whitelisted addresses remain unrestricted, allowing normal token functionality while maintaining compliance boundaries for regulated entities. Administrative Controls: The whitelist is managed by authorized addresses with the ability to add or remove address pairs as regulatory requirements change. This implementation ensures that tokens cannot be distributed to retail investors or users who have not completed the required KYC/onboarding process, while simultaneously providing a clear path for compliant operations. The system architecture allows administrative addresses to enforce regulatory requirements when needed, with particular attention to AML risk prevention through controlled transaction flows.
Transfer fees and beneficiaries	Each transfer requires the use of ETH as gas fees paid to miners, typically borne by the token holder. There are no other handling fees within the smart contract parameter.
Any function that makes it possible for project owner to change contract	Chairman can forcefully transfer tokens from one address to another through the forceTransfer function, but the total token supply remains unchanged. The Minter role can mint tokens through the mint function, and can also burn tokens on the specified deadAddr through the burn function. These two operations will affect the total amount of tokens, but they are all reasonable business functions.
Contract owner can change balance	Minter cannot modify Tokenholders' balance, but chairman can. It's technically not feasible on the operating platform for anyone except chairman to modify Tokenholders' balance. Tokenholders' Balance is otherwise entirely subject to their own trading activities.
No Hidden owners	SlowMist confirms there are "No Hidden owners".
This token cannot self destruct	SlowMist confirms there is "This token cannot self destruct".
No external call risk found, depend on other contracts	SlowMist confirms there is "No hook for external call".

Definition of whitelist and blacklist if any	There is a white list, no blacklist, which is used to limit the transfer of the specified user or not under special circumstances.
If tokens are issued on multiple chains, describe the cross-chain mechanism	SlowMist confirms ComplianceToken is not issued on multiple chains, hence no cross there is no cross-chain demand.
Privacy risks	Smart contracts do not store users' private data, such as Tokenholder's date of birth.

3.4 Information of the Wallet Management

According to the requirements of Circular on intermediaries engaging in tokenised securities-related activities, Part A, (b)(iv), the wallet private key must be managed properly. The following is the basic information on managing wallets:

In a multi-signature wallet, a threshold is the number of signatures required to approve a transaction. For example, a 3-of-5 multi-signature wallet requires 3 out of 5 signatures to approve a transaction.

Basic information of the Gnosis safe multisig			
Wallet address	Wallet key management	Personnel information	Threshold value
address_1	Use the ledger hardware wallet for management	op1	3/5
address_2	Use the ledger hardware wallet for management	op2	
address_3	Use the ledger hardware wallet for management	op3	
address_4	Use the ledger hardware wallet for management	op4	
address_5	Use the ledger hardware wallet for management	op5	

3.5 Information of the DLT Network

According to the requirements of Circular on intermediaries engaging in tokenised securities-related activities, C.9, E.20.(d), Part A.(b).(ii), the type and robustness of DLT networks need to be explained. The following is the basic information of DLT networks:

Basic information of DLT networks		
Robustness	Consensus	Archetypes of DLT networks

Setup by OP-Stack, maintained by HashKey Cloud who is a leading node provider and is partnered with many experienced RaaS platforms.	Optimistic Rollups and Ethereum PoS	public-permissionless

HashKey Chain leverages Optimistic Rollups technology to reduce transaction costs and increase throughput while inheriting Ethereum's security and maintaining EVM compatibility for seamless smart contract migration.

3.6 Information of the Tokenisation Arrangement

In accordance with the requirements of Circular on intermediaries engaging in tokenised securities-related activities, C.10, E.20.(a)(b)(e), the material information on the tokenisation arrangement needs to be explained.

Basic information of the tokenisation arrangement			
Settlement	Transfer restrictions	DLT-related events	Custodial arrangement
The Master Registrar of Tokenholders kept by the Trustee (XPert) will be the source of truth and considered final and irreversible. The Master Register of Token holders will contain 4 parts to ensure accuracy and comprehensiveness on the token level: (i) On-chain record of the issuance of the new Tokens minted according to the smart contract to whitelisted addresses of Xpert; (ii) Off-chain record of transactions among whitelisted address of Xpert; (iii) On-chain record from whitelisted XPert address to whitelisted addresses of HBL; and (iv) Off-chain record of	Transfer restrictions need to be triggered based on whitelisting and other AML or risk management strategies.	Data validation through on-chain and off-chain data checks can help prevent DLT-related events from affecting transaction data.	Allow users to authorize their funds to a designated wallet for custody.

Token ownership of users of HBL's platform			

3.7 Information of the Anti-money laundering

In accordance with the requirements of Circular on intermediaries engaging in tokenised securities-related activities, C.10, PartA.(f), the following is an explanation of the anti-money laundering:

Basic information of the Anti-money laundering	
AML	KYT
Will adopt KYC/AMLProcedure of the VA Exchange	Will adopt the KYT mechanism of the VA Exchange

4 Code Overview

4.1 Contracts Description

Audit Version:
https://github.com/HashKeyChain/hbs-token-contracts/src
commit: dc5c5573cd0c27928343d955f9e998fb8d7a055a
Fixed Version:
https://github.com/HashKeyChain/hbs-token-contracts/src
commit: 2a64db4df79257a753181e336ad29610e43cb50f

The main network address of the contract is as follows:
The code was not deployed to the mainnet.

4.2 Visibility Description

The SlowMist Security team analyzed the visibility of major contracts during the audit, the result as follows:

ComplianceToken			
Function Name	Visibility	Mutability	Modifiers
<Constructor>	Public	Can Modify State	ERC20

initialize	Public	Can Modify State	initializer
name	Public	-	-
symbol	Public	-	-
forceTransfer	Public	Can Modify State	onlyChairman
transferChairman	Public	Can Modify State	onlyChairman
updateWhitelister	Public	Can Modify State	onlyChairman
updateMinter	Public	Can Modify State	onlyChairman
updateDepositAddr	Public	Can Modify State	onlyChairman
updateWithdrawAddr	Public	Can Modify State	onlyChairman
updateDeadAddr	Public	Can Modify State	onlyChairman
updateWhitelist	Public	Can Modify State	onlyWhitelister
mint	Public	Can Modify State	onlyMinter
burn	Public	Can Modify State	onlyMinter
transferFrom	Public	Can Modify State	-
transfer	Public	Can Modify State	-
detectTransferRestriction	Public	-	-
messageForTransferRestriction	Public	-	-

4.3 Vulnerability Summary

N1 [Medium] Risk of excessive authority

1. The chairman role has the ability to forcefully transfer tokens from one address to another through the forceTransfer function.
Code location:
src/ComplianceToken.sol#

```
    function forceTransfer(address from, address to, uint256 amount) public
onlyChairman {
        require(balanceOf(from) >= amount, "insufficient balance");
```

```
        _transfer(from, to, amount);
        emit ForceTransfer(from, to, amount);
    }
```

2. The chairman role can add or remove the minter role through the updateMinter function, and the minter role can arbitrarily mint tokens through the mint function with no upper limit on the token amount.
Code location:
src/ComplianceToken.sol#L158-L167, L226-L231

```
    function updateMinter(address addr, bool isMinter) public onlyChairman {
        require(addr != address(0), "new address cannot be zero address");
        if (isMinter) {
            minter[addr] = true;
            emit AddMinter(addr);
        } else {
            delete minter[addr];
            emit RemoveMinter(addr);
        }
    }

    function mint(uint256 amount, uint256 validUntil) public onlyMinter {
        require(amount > 0, "mint amount must be greater than zero");
        require(block.timestamp <= validUntil, "this transaction has
expired");
        _mint(depositAddr, amount);
        emit TokensMinted(depositAddr, amount);
    }
```

Solution:
In the short term, transferring owner ownership to multisig contracts is an effective solution to avoid single-point risk. But in the long run, it is a more reasonable solution to implement a privilege separation strategy and set up multiple privileged roles to manage each privileged function separately. And the authority involving user funds should be managed by the community, and the authority involving emergency contract suspension can be managed by the EOA address. This ensures both a quick response to threats and the safety of user funds.

N2 [Suggestion] Missing the event records

In the contract, the whitelister role can add or remove whitelist addresses through the updateWhitelist function, but there is no event log.

Code location:
src/ComplianceToken.sol#L

```
    function updateWhitelist(address[2][] memory addr, bool isWhitelist)
public onlyWhitelister {
        if (!isWhitelist) {
            // remove whitelist
            for (uint256 i = 0; i < addr.length; i++) {
                delete whitelistIndex[addr[i][0]];
                delete whitelist2Index[addr[i][1]];
            }
        } else {
            for (uint256 i = 0; i < addr.length; i++) {
                // add new whitelist
                whitelistIndex[addr[i][0]] = latestIndex + 1;
                whitelist2Index[addr[i][1]] = latestIndex + 1;
                latestIndex++;
            }
        }
    }
```

Solution:
It is recommended to record events when sensitive parameters are modified for self-inspection or community review.

N3 [Suggestion] Missing zero address check

The chairman role can change the chairman address through the transferChairman function, but the function does not check whether the newly changed address is the zero address. If erroneously transferred to the zero address, there is a risk of permission loss.
Code location:
src/ComplianceToken.sol#L138-L141

```
    function transferChairman(address addr) public onlyChairman {
        chairman = addr;
        emit ChairmanUpdated(addr);
    }
```

Solution:
It is recommended to add the 0 address check when sensitive parameters are modified.

5 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002502280001	SlowMist Security Team	2025.02.25 - 2025.02.27	Medium Risk

Summary conclusion: The SlowMist security team uses a manual and SlowMist team's analysis tools to audit the project, during the audit work we found 1 medium risk, and 2 suggestions. The code was not deployed to the mainnet.

6 Statement

SlowMist issues this report with reference to the facts that have occurred or existed before the issuance of this report, and only assumes corresponding responsibility based on these.
For the facts that occurred or existed after the issuance, SlowMist is not able to judge the security status of this project, and is not responsible for them. The security audit analysis and other contents of this report are based on the documents and materials provided to SlowMist by the information provider till the date of the insurance report (referred to as "provided information"). SlowMist assumes: The information provided is not missing, tampered with, deleted or concealed. If the information provided is missing, tampered with, deleted, concealed, or inconsistent with the actual situation, the SlowMist shall not be liable for any loss or adverse effect resulting therefrom. SlowMist only conducts the agreed security audit on the security situation of the project and issues this report. SlowMist is not responsible for the background and other conditions of the project.

Official Website

www.slowmist.com

E-mail

team@slowmist.com

Twitter

@SlowMist_Team

Github

https://github.com/slowmist