Cover Peripheral Smart Contract Audit

tags: `Cover`

Introduction

General Provisions

_

Scope of the Audit

The scope of the audit includes the following smart contracts at:

https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol)
https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/Rollover.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/Rollover.sol)

https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/utils/Address.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/utils/Address.sol)
https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/utils/Ownable.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/utils/Ownable.sol)
https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/utils/ReentrancyGuard.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/utils/ReentrancyGuard.sol)
https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/utils/SafeERC20.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/utils/SafeERC20.sol)

https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/utils/SafeMath.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/utils/SafeMath.sol)

https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/IBFactory.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/IBFactory.sol)

https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/IBlacksmith.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/IBlacksmith.sol)

https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/IBPool.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/IBPool.sol)

https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/ICover.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/ICover.sol)

https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/ICoverERC20.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/ICoverERC20.sol)

https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/ICoverRouter.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/ICoverRouter.sol)

https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/IERC20.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/IERC20.sol)

https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/IProtocol.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/IProtocol.sol)

https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/IRollover.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/interfaces/IRollover.sol)

The audited commit identifier is `d5b37e34d47abec3252cdabd46e55e34a72421d4`

Security Assessment Principles

Classification of Issues

- CRITICAL: Bugs leading to Ether or token theft, fund access locking or any other loss of Ether/tokens to be transferred to any party (for example, dividends).

- MAJOR: Bugs that can trigger a contract failure. Further recovery is possible only by manual modification of the contract state or replacement.

- WARNINGS: Bugs that can break the intended contract logic or expose it to DoS attacks.

- COMMENTS: Other issues and recommendations reported to/ acknowledged by the team.

Security Assessment Methodology

Two auditors independently verified the code.

Stages of the audit were as follows:

- “Blind” manual check of the code and its model
- “Guided” manual code review
- Checking the code compliance with customer requirements
- Discussion of independent audit results
- Report preparation

Report

CRITICAL

Not found

MAJOR

1. It is possible to carry out attacks to manipulate pools within one transaction using an flash loan

Description

In contracts https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol) and https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/Rollover.sol (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/Rollover.sol), any user can exchange tokens with a contract. Any user can add and remove liquidity. An attacker can take an flash loan and perform multiple liquidity manipulations within a single transaction. These manipulations can lead to loss of funds for other users.

Recommendation

It is recommended to add protection against token manipulation with flash loans.
Here’s some sample code:

```
mapping(address => uint256) private _lastSwapBlock;

function some() external {
   _preventSameTxOrigin();
   ....
   some logic
   ...
 }

function _preventSameTxOrigin() private {
   require(block.number > _lastSwapBlock[tx.origin], "SAME_TX_ORIGIN");
   _lastSwapBlock[tx.origin] = block.number;
 }
```

Status

Fixed at https://github.com/CoverProtocol/cover-peripheral/commit/492741bc6a4179231f389505b13624535126ba20 (https://github.com/CoverProtocol/cover-peripheral/commit/492741bc6a4179231f389505b13624535126ba20)

2. Ability to steal tokens from contract balance

Description

Method `addCoverAndCreatePools` defined at https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L128 (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L128) accepts `_protocol` and `_collateral` addresses as arguments, then call `_addCover` that make approve for `_protocol` an unlimited amount of `_collateral` tokens and call `_protocol.addCover`. There are no checks of `_protocol` and `_collateral` validity, so attacker can pass malicious `_protocol` and get unlimited approval of `_collateral` tokens:

```
  if (_token.allowance(address(this), _spender) < _amount) {
    _token.approve(_spender, uint256(-1));
  }
```

According contract logic in optimistic flow contract shouldn't have any tokens in balance, but that invariant not fully checked, e.g. if `_protocol.addCover` at https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/Rollover.sol#L75 (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/Rollover.sol#L75) fails and return false then transaction will executed successfully, but user funds will be leave at CoverRouter balance.

Recommendation

We recommend to not use unlimited approve and add particular checks to keep zero balance invariant.

Status

Fixed at https://github.com/CoverProtocol/cover-peripheral/commit/492741bc6a4179231f389505b13624535126ba20 (https://github.com/CoverProtocol/cover-peripheral/commit/492741bc6a4179231f389505b13624535126ba20)

WARNINGS

1. No validation of the address parameter value in contract constructor

Description

The variable is assigned the value of the constructor input parameter. But this parameter is not checked before this. If the value turns out to be zero, then it will be necessary redeploy the contract, since there is no other functionality to set this variable.

- At the line https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L36 (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L36) the `protocolFactory` variable is set to the value of the `_protocolFactory` input parameter.
- At the line https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L37 (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L37) the `bFactory` variable is set to the value of the `_bFactory` input parameter.

Recommendation

In all the cases, it is necessary to add a check of the input parameter to zero before initializing the variables.

Status

Fixed at https://github.com/CoverProtocol/cover-peripheral/commit/492741bc6a4179231f389505b13624535126ba20 (https://github.com/CoverProtocol/cover-peripheral/commit/492741bc6a4179231f389505b13624535126ba20)

2. It is possible to use Front-Running attack

Description

Since all transactions are visible in the mempool for a short while before being executed, observers of the network can see and react to an action before it is included in a block. An example of how this can be exploited is with a decentralized exchange where a buy order transsaction can be seen, and second order can be broadcast and executed before the first transaction is included.

At the line https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L64 (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L64), any user can execute the `rolloverAndAddLiquidityForAccount()` function. This function calls other functions and exchanges tokens between the contract and the `_account` address. Due to the fact that another user completes the transaction earlier, a profitable position in the trade may be lost and the user will lose his profit.

Recommendation

The best remediation is to remove the benefit of Front-Running in your application, mainly by removing the importance of transaction ordering or time. It will be possible to make use of the maximum or minimum acceptable price or amount range for the transaction, thereby limiting price slippage.

Status

Acknowledged

3. Need to check remaining tokens

In contract there is the method `_transferRem`. But in some cases there are no checks:

- https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L45 (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L45) (`addCoverAndAddLiquidity`)
- https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L115 (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L115) (`addLiquidity`)
- https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L158 (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L158) (`createNewPool`)

Recommendation

We recommend to check the all tokens and transfer remaining balance after every external method of the contract.

Status

Fixed at https://github.com/CoverProtocol/cover-peripheral/commit/492741bc6a4179231f389505b13624535126ba20 (https://github.com/CoverProtocol/cover-peripheral/commit/492741bc6a4179231f389505b13624535126ba20)

4. Unlimited approval

At this line `_token.approve(_spender, uint256(-1)` (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/Rollover.sol#L64 (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/Rollover.sol#L64)) we have approval unlimited access from CoverRouter.

Recommendation

We recommend either to revoke approval after execution or to approve only expect amount.

Status

Fixed at https://github.com/CoverProtocol/cover-peripheral/commit/492741bc6a4179231f389505b13624535126ba20 (https://github.com/CoverProtocol/cover-peripheral/commit/492741bc6a4179231f389505b13624535126ba20)

COMMENTS

1. No events for parameter changes

Basic features for `onlyOwner` don’t emit any events:

- https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L179 (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L179) (`setSwapFee`)
- https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L185 (https://github.com/CoverProtocol/cover-peripheral/tree/d5b37e34d47abec3252cdabd46e55e34a72421d4/contracts/CoverRouter.sol#L185) (`setCovTokenWeights`)

Recommendation

We recommend to create events: `SwapFeeUpdate` , `CovTokenWeightUpdate` .

Status

Fixed at https://github.com/CoverProtocol/cover-peripheral/commit/492741bc6a4179231f389505b13624535126ba20 (https://github.com/CoverProtocol/cover-peripheral/commit/492741bc6a4179231f389505b13624535126ba20)

CONCLUSION

Findings list

Level	Amount
CRITICAL	0
MAJOR	2
WARNING	4
COMMENT	1

Final commit identifier with all fixes: `cbf6f30cde5ca6830af0554f3fb5247ae3bdaf06`