MOVEBIT
Securing the Move Ecosystem

Starcoin-Poly Bridge STC Contracts **Audit Report**

https://twitter.com/movebit_

contact@movebit.xyz

Starcoin-Poly Bridge STC Contracts Audit Report

1 Executive Summary

1.1 Project Information

Type	Bridge
Auditors	MoveBit
Timeline	2022-09-21 to 2022-10-09
Languages	Move
Methods	Architecture Review, Unit Testing, Formal Verification, Manual Review
Specification	
Source Code	github <https://github.com/Elements-Studio/poly-stc-contracts/tree/master/sources> , git version 0ae06a7b3f8a2fef4eb9b4740428790cfaf28169

1.2 Issue Statistic

Item	Count	Fixed	Pending
Total	16		16
Minor	14		14
Medium	2		2
Major			
Critical			

1.3 Issue Level

- **Minor** issues are general suggestions relevant to best practices and readability. They don't post any direct risk. Developers are encouraged to fix them.
- **Medium** issues are non-exploitable problems and not security vulnerabilities. They should be fixed unless there is a specific reason not to.
- **Major** issues are security vulnerabilities. They put a portion of users' sensitive information at risk, and often are not directly exploitable. All major issues should be fixed.
- **Critical** issues are directly exploitable security vulnerabilities. They put users' sensitive information at risk. All critical issues should be fixed.

1.4 Issue Status

- **Fixed:** The issue has been resolved.
- **Pending:** The issue has been acknowledged by the code owner, but has not yet been resolved. The code owner may take action to fix it in the future.

2 Summary of Findings

The Starcoin-Poly cross-chain bridge provides the ability to move STC, ETH, and USDT assets between Starcoin and Ethereum. Our team read the design documents, took a review of the framework architecture, and then tried the service on the testnet to move assets bidirectionally. Besides focusing mainly on the code review and formal verification with the Move Prover, our team also had a glance at the relayer codes and smart contract codes on Ethereum. Our team has been in close contact with the developing team for the past few days. As a result, our team found a total of 16 issues. The teams have discussed them together, and the developing team plans to address them soon.

3 MoveBit Audit BreakDown

MoveBit aims to assess repositories for security-related issues, code quality, and compliance with specifications and best practices. Possible issues our team looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Integer overflow/underflow
- Number of rounding errors
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting
- Unchecked CALL Return Values
- The flow of capability
- Witness Type

4 Methodology

The security team adopted the "**Testing and Automated Analysis**", "**Code Review**" and "**Formal Verification**" strategy to perform a complete security test on the code in a way that is closest to the real attack. The main entrance and scope of security testing are stated in the conventions in the "Audit Objective", and that can expand to contexts beyond the scope according to the actual testing needs. The main types of this security audit include:

(1) Testing and Automated Analysis

Items to check: state consistency / failure rollback / unit testing / value overflows / parameter verification / unhandled errors / boundary checking / coding specifications.

(2) Code Review

Refer to **Appendix 1** for Code scope.

(3) Formal Verification

Perform formal verification for key functions with the Move Prover.

(4) Audit Process

- Carry out relevant security tests on the testnet or the mainnet;
- If there are any questions during the audit process, communicate with the code owner in time. The code owners should actively cooperate (this might include providing the latest stable

source code, relevant deployment scripts or methods, transaction signature scripts, exchange docking schemes, etc.);

- The necessary information during the audit process will be well documented for both the audit team and the code owner in a timely manner.

5 Findings

5.1 Invalid condition `LESS_THAN == EQUAL` in if statement

Severity: Medium
Status: Pending
Descriptions: There is an invalid condition in `IF` statement in module `SafeMath`. The invalid condition is `if (cmp_order == LESS_THAN || LESS_THAN == EQUAL)`. `LESS_THAN == EQUAL` is a comparison between two constants, it's always false. The correct code here should be `cmp_order == EQUAL`.

Code Location: sources/common/SafeMath.move, line 90

SafeMath.move

```
public fun sqrt_u256(y: U256): u128 {
    let u128_max = U256::from_u128(U128_MAX);
    let cmp_order = U256::compare(&y, &u128_max);
    if (cmp_order == LESS_THAN || LESS_THAN == EQUAL){
        let z = Math::sqrt(U256::to_u128(&y));
        (z as u128)
    } else {
        let z = copy y;
        let one_u256 = U256::from_u128(1u128);
        let two_u256 = U256::from_u128(2u128);
        let x = U256::add(U256::div(copy y, copy two_u256), one_u256);
        while (U256::compare(&x, &z) == LESS_THAN) {
            z = copy x;
            x = U256::div(U256::add(U256::div(copy y, copy x), copy x), copy two_u256);
        };
        U256::to_u128(&z)
    }
}
```

Suggestion: Change `LESS_THAN == EQUAL` to `cmp_order == EQUAL`.

5.2 Long vector<u8> argument may cause overflow to abort

Severity: Medium
Status: Pending

Descriptions: The arguments for `bytes_to_u64`, `bytes_to_u128` and `bytes_reverse_to_u128` are of type `vector<u8>`. If the arg's length is greater than 8, it will cause `bytes_to_u64` to abort. If the arg's length is greater than 16, it will cause `bytes_to_u128` and `bytes_reverse_to_u128` to abort.

Code Location: sources/common/Bytes.move, line 59,74,88

Bytes.move

```
public fun bytes_to_u64(data: &vector<u8>): u64 {
    let number: u64 = 0;
    let i = 0;
    let len = Vector::length(data);
    while (i < len) {
        let slice = *Vector::borrow(data, i);
        let bit = (len - (i + 1) as u8);
        //BitOperators::lshift return only u64
        number = number + BitOperators::lshift((slice as u64), bit * 8);
        i = i + 1;
    };
    number
}

// big endian cast
public fun bytes_to_u128(data: &vector<u8>): u128 {
    let number: u128 = 0;
    let i = 0;
    let len = Vector::length(data);
    while (i < len) {
        let slice = *Vector::borrow(data, i);
        let bit = (len - (i + 1) as u8);
        number = number + lshift_u128((slice as u128), bit * 8);
        i = i + 1;
    };
    number
}

// little endian cast
public fun bytes_reverse_to_u128(data: &vector<u8>): u128 {
    let number: u128 = 0;
    let len = Vector::length(data);
    if (len > 0){
        let i = len - 1;
        loop {
            let slice = *Vector::borrow(data, i);
            let bit = (i as u8);
            number = number + lshift_u128((slice as u128), bit * 8);
            if( i == 0){
                break
            };
            i = i - 1;
        };
    };
    number
}
```

Suggestion: Add assertions for `vector<u8>` arguments to make sure they have valid lengths.

5.3 Unnecessary assertion in function `SMTProofUtils::path_bits_to_bool_vector_from_msb`

Severity: Minor

Status: Pending

Descriptions: There are two assertions in function `SMTProofUtils::path_bits_to_bool_vector_from_msb`. They are duplicated, and the second one can be derived from the first.

Code Location: sources/smt/SMTProofUtils.move, line 15

SMTProofUtils.move

```
public fun path_bits_to_bool_vector_from_msb(path: &vector<u8>): vector<bool> {
    let path_len = Vector::length<u8>(path);
    assert!(path_len == SMTreeHasher::path_size(), Errors::invalid_argument(ERROR_INVALID_PATH_BYTES_LENGTH));
    let result_vec = SMTUtils::bits_to_bool_vector_from_msb(path);
    assert!(Vector::length<bool>(&result_vec) == SMTreeHasher::path_size_in_bits(), Errors::invalid_state(ERROR_INVALID_PATH_BITS_LENGTH));
    result_vec
}
```

Suggestion: Remove this line of code `assert!(Vector::length<bool>(&result_vec) == SMTreeHasher::path_size_in_bits(), Errors::invalid_state(ERROR_INVALID_PATH_BITS_LENGTH));`.

5.4 Operator `&` in expression is not clear in `Bytes::get_bit`

Severity: Minor

Status: Fixed

Descriptions: The operator `&` in Move has different precedence from that in common languages like C/C++. It may cause ambiguity, especially when used in long expressions.

Code Location: sources/common/Bytes.move, line 56

sources/proof/StarcoinVerifier.move, line 67

Bytes.move

```
public fun get_bit(data: &vector<u8>, index: u64): bool {
    let pos = index / 8;
    let bit = (7 - index % 8);
    (*Vector::borrow(data, pos) >> (bit as u8)) & 1u8 != 0
}
```

Suggestion: Add brackets to clarify operator precedence, change to `((*Vector::borrow(data, pos) >> (bit as u8)) & 1u8) != 0`. In addition, the function `get_bit` are duplicated in two modules, remove one.

5.5 Unnecessary calls of `Vector::borrow` in nested while statement in `CrossChainLibrary.move`

Severity: Minor

Status: Pending

Descriptions: The call of `Vector::borrow(signers, i)` can be extracted into the outer while statement, which can reduce unnecessary performance loss.

Code Location: sources/library/CrossChainLibrary.move, line 322

CrossChainLibrary.move

```
public fun contain_m_addresses(keepers: &vector<vector<u8>>, signers:
                               &vector<vector<u8>>, m: u64): bool{
        let cm:u64 = 0;
        let i = 0;
        let signers_len = Vector::length(signers);
        while ( i < signers_len){
            let j = 0;
            let keepers_len = Vector::length(keepers);
            while ( j < keepers_len) {
                let signer = Vector::borrow(signers, i);
                let keeper = Vector::borrow(keepers, j);
                ...
                j = j + 1;
            };
            i = i + 1;
        };
        return cm >= m
    }
```

Suggestion: Extract `Vector::borrow(signers, i)` to the outer while statement

```
public fun contain_m_addresses(keepers: &vector<vector<u8>>, signers:
                               &vector<vector<u8>>, m: u64): bool{
        let cm:u64 = 0;
        let i = 0;
        let signers_len = Vector::length(signers);
        while ( i < signers_len){
            let j = 0;
            let keepers_len = Vector::length(keepers);
            let signer = Vector::borrow(signers, i);
            while ( j < keepers_len) {
                let keeper = Vector::borrow(keepers, j);
                ...
                j = j + 1;
            };
            i = i + 1;
        };
        return cm >= m
    }
```

5.6 Record events are generated twice when `AssetHashMap` does not exist in `LockProxy::bind_asset_hash`

Severity: Minor

Status: Pending

Descriptions: When a resource `AssetHashMap<TokenT, ToChainType>` does not exist, it generates two `BindAssetEvent` .

Code Location: sources/cross-chain/LockProxy.move, line 257

LockProxy.move

```
public fun bind_asset_hash<TokenT: store, ToChainType: store>(signer: &signer,
                    to_chain_id: u64, to_asset_hash: &vector<u8>)
    acquires LockEventStore, AssetHashMap, LockTreasury {
    ...
    if (!exists<AssetHashMap<TokenT, ToChainType>>(genesis_account)) {
        ...
        inner_emit_asset_hash_event<TokenT>(to_chain_id, to_asset_hash);
    };
    ...
    inner_emit_asset_hash_event<TokenT>(to_chain_id, to_asset_hash);
}
```

Suggestion: Remove the first call of `inner_emit_asset_hash_event` in `IF` statement.

LockProxy.move

```
public fun bind_asset_hash<TokenT: store, ToChainType: store>(signer: &signer,
                    to_chain_id: u64, to_asset_hash: &vector<u8>)
    acquires LockEventStore, AssetHashMap, LockTreasury {
    ...
    if (!exists<AssetHashMap<TokenT, ToChainType>>(genesis_account)) {
        ...
    };
    ...
    inner_emit_asset_hash_event<TokenT>(to_chain_id, to_asset_hash);
}
```

5.7 There are test codes having no relation with bridge modules

Severity: Minor

Status: Pending

Descriptions: Module `Bridge::BCSTest` is a test module, but there is no relevant test code for the `Bytes` module inside.

Code Location: sources/common/Bytes.move, line 136.

Bytes.move

```
#[test_only]
module Bridge::BCSTest {
    //use StarcoinFramework::Vector;
    use StarcoinFramework::Debug;
    //use StarcoinFramework::BitOperators;
    //use StarcoinFramework::Hash;
    use StarcoinFramework::BCS;
    use StarcoinFramework::STC;
    use StarcoinFramework::Token;
    //use Bridge::LockProxy;

    struct CrossChainFeeLockEvent has store, drop {
        from_asset: Token::TokenCode,
        sender: address,
        to_chain_id: u64,
        to_address: vector<u8>,
        net: u128,
        fee: u128,
        id: u128,
    }

    #[test]
    public fun test_bcs_serialize() {
        let cc_fee_event = CrossChainFeeLockEvent{
            from_asset: Token::token_code<STC::STC>(),
            sender: @Bridge,//Signer::address_of(signer),
            to_chain_id: 11,
            to_address: x"18351d311d32201149a4df2a9fc2db8a",//*to_address,
            net: 111,
            fee: 222,
            id: 333,
        };
        let bs = BCS::to_bytes<CrossChainFeeLockEvent>(&cc_fee_event);
        Debug::print<vector<u8>>(&bs);
    }
}
```

Suggestion: Move module `Bridge::BCSTest` into its corresponding file or into a separate file.

5.8 The check `CrossChainGlobal::require_not_freezing()` is not placed in the first line of function `CrossChainManager::cross_chain_with_param_pack`

Severity: Minor

Status: Pending

Descriptions: `CrossChainGlobal::require_not_freezing()` is not placed in the first line of function `CrossChainManager::cross_chain_with_param_pack`, it is not a good practice.

Code Location: sources/cross-chain/CrossChainManager.move, line 271

CrossChainManager.move

```
public fun cross_chain_with_param_pack(signer: &signer,
                                       lock_parameters: CrossChainProcessCombinator::Lo
ckToChainParamPack)
acquires EventStore {
    let (
         to_chain_id,
         to_contract,
         method,
         tx_data
     ) = CrossChainProcessCombinator::unpack_lock_to_chain_parameters(lock_parameters);

     // Check global freezing switch has closed
     CrossChainGlobal::require_not_freezing();
     ......
}
```

Suggestion: `CrossChainGlobal::require_not_freezing()` ought to be placed in the first line of the function.

5.9 TODO labels exist in codes

Severity: Minor
Status: Pending
Descriptions: There are two `TODO` labels in the project. Our team inspected them, and it seems they both need some improvements.

Code Location: sources/library/ZeroCopySource.move, line 143, sources/library/CrossChainLibrary.move, line 181

Suggestion: The dev team needs to search the `TODO` in the code, review all the labels, and make sure all the work is done.

5.10 Variable assignments are ambiguous in `SMTreeHasher.move`

Severity: Minor
Status: Pending
Descriptions: There are ambiguous or optimizable assignment steps. The assignment to variables `start` and `end` is confusing. They could be improved and be more clear.

Code Location: sources/smt/SMTreeHasher.move, line 34-39, line 71-76

SMTreeHasher.move

```
public fun parse_node(data: &vector<u8>): (vector<u8>, vector<u8>) {
    ......

    let start = 0;
    let end = prefix_len;
    _ = start;//let prefix = SMTUtils::sub_u8_vector(data, start, end);

    start = end;
    end = start + path_size();
    ......
}

public fun parse_leaf(data: &vector<u8>): (vector<u8>, vector<u8>) {
    ......
    let start = 0;
    let end = prefix_len;
    _ = start;//let prefix = SMTUtils::sub_u8_vector(data, start, end);

    start = end;
    end = start + path_size();
    ......
}
```

Suggestion: Could change the codes as below.

SMTreeHasher.move

```
let start = prefix_len;
let end = start + path_size();
```

5.11 Repeated calculation within loop statements

Severity: Minor

Status: Pending

Descriptions: The calculation in the loop condition can be calculated in advance to avoid repeated calculation every time the loop runs. This issue exists in a few functions.

Code Location: sources/proof/EthStateVerifier.move, line 51

sources/proof/MerkleProofStructuredHash.move, line 24

sources/smt/SMTUtils.move, line 25,53

```
fun decode_children(data: &vector<u8>, offset: u64, child_offset: u64, length: u64): (v
ector<vector<u8>>, u64) {
    let result = Vector::empty();
    while (child_offset < offset + 1 + length) {
    ...
}


public fun create_literal_hash(word: &vector<u8>) : vector<u8> {
    let word_length = Vector::length(word);
    assert!(word_length <= SMTreeHasher::path_size(), ERROR_INVALID_HASH_LENGTH);
    if (word_length < SMTreeHasher::path_size()) {
        let result = Vector::empty();
        let idx = 0;
        while (idx < SMTreeHasher::path_size()) {
    ...
}


public fun count_common_prefix(data1: &vector<u8>, data2: &vector<u8>): u64 {
    let count = 0;
    let i = 0;
    while ( i < Vector::length(data1)*8) {
    ...
}


public fun bits_to_bool_vector_from_msb(data: &vector<u8>): vector<bool> {
    let i = 0;
    let vec = Vector::empty<bool>();
    while (i < Vector::length(data)*8) {
    ...
}
```

Suggestion: It is recommended to calculate in advance to avoid repeated calculations in each loop.

5.12 Duplicated call of `Signer::address_of` for `signer` in `CrossChainData`

Severity: Minor

Status: Pending

Descriptions: The address of `signer` is calculated at the first line in the function `CrossChainData::init_genesis`, and is stored in the variable `account`. It can be reused to reduce unnecessary performance loss.

Code Location: sources/cross-chain/CrossChaindata.move, line 65

CrossChaindata.move

```
// This function called from cross chain data
public fun init_genesis(signer: &signer) {
    ...
    // Repeate check
    assert!(!exists<SparseMerkleTreeRoot>(Signer::address_of(signer)),
        Errors::invalid_state(ERR_INITIALIZED_REPEATE));
    move_to(signer, SparseMerkleTreeRoot {
        hash: *&SMTreeHasher::placeholder()
    });
}
```

Suggestion: Suggested changes are as follows.

CrossChaindata.move

```
// Repeated check
assert!(!exists<SparseMerkleTreeRoot>(account), Errors::invalid_state(ERR_INITIALIZED_R
EPEATE));
move_to(signer, SparseMerkleTreeRoot {
    hash: *&SMTreeHasher::placeholder()
});
```

5.13 The type of the parameter and the returned value are inconsistent

Severity: Minor

Status: Pending

Descriptions: The return value of `get_eth_tx_hash_index` is of type `u128`, but the type of `get_eth_tx_hash's` parameter is `u64`. They are indices of `Ethereum cross-chain tx hash`, they should be of the same type. Otherwise, a typecast might be required when using it.

Code Location: sources/cross-chain/CrossChainData.move, line 98,111

CrossChaindata.move

```
public fun get_eth_tx_hash_index(): u128 acquires Consensus {
    let consesus = borrow_global_mut<Consensus>(CrossChainGlobal::genesis_account());
consesus.eth_to_poly_tx_hash_index
    }

public fun get_eth_tx_hash(eth_tx_hash_index: u64): vector<u8> acquires Consensus {
    let consesus = borrow_global<Consensus>(CrossChainGlobal::genesis_account());
let eth_to_poly_hash = Vector::borrow<vector<u8>>(&consesus.eth_to_poly_tx_hash, eth_tx
_hash_index);
*eth_to_poly_hash
}
```

Suggestion: It is recommended to use the same type.

5.14 Annotation error

Severity: Minor

Status: Pending

Descriptions: There are some code comment errors, and they do not match the codes. In `CrossChaiManager.move` , there are no return values for some functions, and the comments are incorrect. In `CrossChainScript.move` , the comment for `set_freeze` is incorrect.

Code Location: sources/cross-chain/CrossChainManager.move, line 68,140,232,333

sources/cross-chain/CrossChainScript.move, line 176

CrossChaiManager.move

```
/* @notice              sync Poly chain genesis block header to smart contrat
 *  @dev                this function can only be called once, nextbookkeeper of rawHea
der can't be empty
 *  @param rawHeader    Poly chain genesis block raw header or raw Header including swi
tching consensus peers info
 *  @return             true or false
 */
public fun init_genesis_block(signer: &signer,
                              raw_header: &vector<u8>,
                              pub_key_list: &vector<u8>) acquires EventStore {


/* @notice              change Poly chain consensus book keeper
 *  @param rawHeader    Poly chain change book keeper block raw header
 *  @param pubKeyList   Poly chain consensus nodes public key list
 *  @param sigList      Poly chain consensus nodes signature list
 *  @return             true or false
 */
public fun change_book_keeper(signer: &signer,
                              raw_header: &vector<u8>,
                              pub_key_list: &vector<u8>,
                              sig_list: &vector<u8>) acquires EventStore {


/* @notice              ERC20 token cross chain to other blockchain.
 *                      this function push tx event to blockchain
 *  @param toChainId    Target chain id
 *  @param toContract   Target smart contract address in target block chain
 *  @param txData       Transaction data for target chain, include to_address, amount
 *  @return             true or false
 */
public fun cross_chain_with_param_pack(signer: &signer,
                                       lock_parameters: CrossChainProcessCombinator::Lo
ckToChainParamPack)
acquires EventStore {


/* verifyHeaderAndExecuteTx
 * @notice             Verify Poly chain header and proof, execute the cross chain tx f
rom Poly chain to Ethereum
 *  @param proof        Poly chain tx merkle proof
 *  @param rawHeader    The header containing crossStateRoot to verify the above tx mer
kle proof
 *  @param headerProof  The header merkle proof used to verify rawHeader
 *  @param curRawHeader  Any header in current epoch consensus of Poly chain
 *  @param headerSig    The coverted signature veriable for solidity derived from Poly
chain consensus nodes' signature
 *                      used to verify the validity of curRawHeader
 *  @return             true or false
 */
public fun verify_header_with_param_pack(proof: &vector<u8>,
                                         raw_header: &vector<u8>,
```

```
                                        header_proof: &vector<u8>,
                                        cur_raw_header: &vector<u8>,
                                        header_sig: &vector<u8>)
: CrossChainProcessCombinator::HeaderVerifyedParamPack acquires EventStore {
```

CrossChainScript.move

```
// Set admin account by genesis account
public(script) fun set_freeze(signer: signer, switch: bool) {
    CrossChainConfig::set_freeze(&signer, switch);
}
```

Suggestion: Update the code comments.

5.15 Overflow judgment is useless

Severity: Minor

Status: Pending

Descriptions: If an arithmetic operation will cause overflow, it aborts in Move. So the judgments like `offset < offset + 1` are useless.

Code Location: sources/library/ZeroCopySource.move, line 48,66,77,89,102,115,128,142,156,167,180

ZeroCopySource.move

```
public fun next_bool(data: &vector<u8>, offset: u64) : (bool, u64) {
    assert!(((offset + 1) <= Vector::length(data)) && (offset < offset + 1 ),
            Errors::invalid_state(ERR_NEXT_BYTE_OFFSET_EXCEED));
    let data_slice = Bytes::slice(data, offset, offset + 1);
    ...
}
```

Suggestion: Could change codes as follows.

```
// define constant for max u64 number.
const U64_MAX:u64 = 0xFFFFFFFFFFFFFFFF;

public fun next_bool(data: &vector<u8>, offset: u64) : (bool, u64) {
    assert!(((offset + 1) <= Vector::length(data)) && (offset <= U64_MAX - 1),
            Errors::invalid_state(ERR_NEXT_BYTE_OFFSET_EXCEED));
    let data_slice = Bytes::slice(data, offset, offset + 1);
    ...
}

//other:
public fun [function](data: &vector<u8>, offset: u64) : (RETURN_TYPE, u64) {
    assert!(((offset + 1) <= Vector::length(data)) && (offset <= U64_MAX - n),
            Errors::invalid_state(ERR_NEXT_BYTE_OFFSET_EXCEED));
    let data_slice = Bytes::slice(data, offset, offset + 1);
    ...
}
```

5.16 Miss validation in `CrossChainLibrary::verify_sig`

Severity: Minor

Status: Pending

Descriptions: Parameter `sig_list` is missing validation. As a consequence, invalid signatures might pass in.

Code Location: sources/library/CrossChainLibrary.move, line 139

CrossChainLibrary.move

```
public fun verify_sig(raw_header: &vector<u8>, sig_list: &vector<u8>,
                      keepers: &vector<vector<u8>>, m: u64): bool{
        let hash = get_header_hash(raw_header);
        let sig_count = Vector::length(sig_list) / POLYCHAIN_SIGNATURE_LEN;
    ...
}
```

Suggestion: Add validation for parameter `sig_list` like `CrossChainLibrary::verify_pubkey` .

CrossChainLibrary.move

```
public fun verify_sig(raw_header: &vector<u8>, sig_list: &vector<u8>,
                      keepers: &vector<vector<u8>>, m: u64): bool{
        assert!(Vector::length(sig_list) % POLYCHAIN_SIGNATURE_LEN == 0 , Errors::inval
id_state(ERR_SIGNATURE_LIST_LEN_ILLEGAL));
        let hash = get_header_hash(raw_header);
        let sig_count = Vector::length(sig_list) / POLYCHAIN_SIGNATURE_LEN;
    ...
}
```

6 Prover Formal Verification

Our team added formal specifications for most of the important functions, except for some functions that contain special elements that cannot be reasoned about. All the verification codes will be submitted as PR (pull request) to the code repository, and get merged by the developing team in later revisions.

The formal verification report of all files and modules is as follows.

Bytes

General Descriptions

This module is located in `sources/common/Bytes.move`.

This module provides simpler and more efficient utility functions related to bytes.

Formally Verified Properties

- Add `invariant` conditions to while loops in some functions.
- Get the value of `Vector` by `index`, which must be less than the length of `Vector`.
- After `concat()` is executed, the length of the returned result must be the sum of the lengths of `v1` and `v2`. When `v2` is empty, the last value of the returned result is equal to the last value of `v1`. When `v2` is not empty, the last value of the returned result is equal to the last value of `v2`.
- Use the `aborts_if false` expression for functions that never abort.
- In `left_padding()` and `right_padding()`, the length of `data` after the function is executed is greater than or equal to `total_len`.
- After `create_zero_bytes()` is executed, the length of the returned result is always equal to the parameter `length`.

SafeMath

General Descriptions

This module is located in `sources/common/SafeMath.move`.

This module is about math and provides more secure operations and comparison utilities.

Formally Verified Properties

- The divisor cannot be 0.
- In `safe_mul_div()`, the `x*y/z` calculation result must be greater than the u128 maximum value.

CrossChainLibrary

General Descriptions

This module is located in `sources/library/CrossChainLibrary.move`.

This module is cross-chain related and provides utility functions for Merkle root verification.

Formally Verified Properties

- In `verify_pubkey()`, the length of `pub_key_list` must be a multiple of 67, and the length of `pub_key_list` must be greater than 67.
- Add invariant conditions to while loops in some functions.

ZeroCopySink

General Descriptions

This module is located in `sources/library/ZeroCopySink.move`.

Wrappers over encoding and serialization operation into bytes from basic types in Move for PolyNetwork cross-chain utility.

Encode basic types in Move into bytes easily.

Formally Verified Properties

- Make sure the conversion result is correct.

ZeroCopySource

General Descriptions

This module is located in `sources/library/ZeroCopySource.move`.

Wrappers over decoding and deserialization operation from bytes into basic types in Move for PolyNetwork cross-chain utility.

Decode into basic types in Move from bytes easily.

Formally Verified Properties

- Verified abort condition.

SMTProofs

General Descriptions

This module is located in `sources/smt/SMTProofs.move`.

Sparse Merkle Tree proof for non-membership.This is a concrete realization of the Sparse Merkle Tree.

Formally Verified Properties

- Create membership proof side nodes from non-membership proof. Added loop Invariants check.

- Compute root hash added loop Invariants check.
- Other parts of this module have been verified in SMTProofUtils, SMTreeHasher and SMTUtils.

SMTProofUtils

General Descriptions

This module is located in `sources/smt/SMTProofUtils.move`.

Split sibling nodes data from concatenated data. Concatenate all vectors in `vector<vector<u8>>` into one `vector<u8>`.

Formally Verified Properties

- Added loop Invariants check.
- Make sure the length of the `side node data` is correct.
- Make sure the size of the `path` parameter is correct.

SMTreeHasher

General Descriptions

This module is located in `sources/smt/SMTreeHasher.move`.

Partial implementation of Sparse Merkle Tree. Digest leaf and node data. Parse leaf and node data.

Formally Verified Properties

- The length of the parsed leaf data should be greater than the leaf prefix, and the prefix is `x"00"`.
- The length of the parsed node data should be greater than the node prefix, and the prefix is `x"01"`.
- The length of the digested leaf data should be greater than the leaf prefix, and the prefix is `x"00"`.
- The length of the digested node data should be greater than the leaf prefix, and the prefix is `x"00"`.
- The length of `left_data` and `right_data` of the digested node should be the same.

SMTUtils

General Descriptions

This module is located in `sources/smt/SMTUtils.move`.

Utility functions for Sparse Merkle Tree. Include getting the bit at an offset from the most significant bit. Concatenate two vectors.

Formally Verified Properties

- Added loop Invariants check.

- The `data` length to get the bit at an offset from the most significant bit can not be `0`.
- The length of `data` should be less than `position / 8`.
- Count the number of prefixes that are the same for both vectors to ensure the same length.

Appendix 1 - Files in Scope

This audit covered the following files:

Files	SHA-1 Hash
sources/asset/erc20/XUSDT.move	c4f0fb82e245f82f0e4bd5fb0f82bebfee51c816
sources/asset/erc20/XETH.move	9be3f9313468edf1058a9f6cf9b2ada58395cb9
sources/cross-chain/LockProxy.move	ffc384362f243a09c1271312509e1137aa0aff10
sources/cross-chain/CrossChainScript.move	5a379f5a9c5596c1d30acf4afa769efc8bba6c2f
sources/cross-chain/CrossChainRouter.move	311ec47834480173de337031ffce6f37d6611af3
sources/cross-chain/CrossChainGlobal.move	6b73f3416a0e64fc14de95ce7216649d9648089
sources/cross-chain/CrossChainProcessCombinator.move	452462f382e5aef717dcbc50839e0f04a8cb2ea
sources/cross-chain/CrossChainConstant.move	99cf64c7748bfc9561dcdbee3b9285e063b12f8
sources/cross-chain/CrossChainManager.move	883e4b7f5f1956195bb1d8077c7252da8ba7d3
sources/cross-chain/CrossChainConfig.move	daa934797f9be4e3a22e8fc9af9e0797ab147de
sources/cross-chain/CrossChainData.move	838e67093ea7950a11ebacda0afcb3fce6d5a07
sources/proof/EthStateVerifier.move	a4ce01e8e2427c520a2befeb5861158d892224
sources/proof/StarcoinVerifier.move	633c5554af8d1f0c2e84c1ed8cfa311db5efe875
sources/proof/CrossChainSMTProofs.move	a9ea0af9f2241fda866a4b77f149c5b136a3d012
sources/proof/MerkleProofStructuredHash.move	27340ec6f808a2cd9e8b41b46933fa4da0bf839
sources/common/Bytes.move	b2f95b5286d62d3620e55dd58666e714bc25d5
sources/common/Address.move	d74e965e5f61d1da872eb9ccb5021ae967ca68
sources/common/SafeMath.move	dea3966b8c8ea0002f9f31df6fa3579c5cace5d4
sources/smt/SMTreeHasher.move	1fa109e783ff7e2bf878fb817a3380e7de71e796
sources/smt/SMTUtils.move	e08b666fd74edc4e51a11b1d2c095cd6204614
sources/smt/SMTProofUtils.move	acff0f41b7f9b80544758dffe17a8616231680a2
sources/smt/SMTProofs.move	f6b1b9783ede093dec3e6defce3a8a4cfef050f6
sources/smt/SMTHash.move	e1d8afb196e8bd0b92a5c109c66c39e45c2d0b
sources/upgrade/UpgradeScript.move	cf2beb442412e7a585b01bfde21e6fd10068bc2
sources/library/CrossChainLibrary.move	1b0dd7a406282d2731fcf73da0bb2e9e6e85d3
sources/library/ZeroCopySource.move	b839456122efd59a98b880123222430d182363
sources/library/ZeroCopySink.move	3e7571393096c69b6e64e2bc29f6a06a0eb4c9c

Appendix 2 - Disclaimer

This report is based on the scope of materials and documents provided, with a limited review at the time provided. Results may not be complete and do not include 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 own risk. A report does not imply an endorsement of any particular project or team, nor does it guarantee its security. These reports should not be relied upon in any way by any third party, including for the purpose of making any decision to buy or sell products, services, or any other assets. TO THE FULLEST EXTENT PERMITTED BY LAW, WE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, IN CONNECTION WITH THIS REPORT, ITS CONTENT, RELATED SERVICES AND PRODUCTS, AND YOUR USE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NOT INFRINGEMENT.

MOVEBIT

Securing the Move Ecosystem

https://twitter.com/movebit_

contact@movebit.xyz