hexens

POLYGON TECHNOLOGY SMART CONTRACT AUDIT REPORT

27.10.2022

CONFIDENTIAL

CONFIDENTIAL

CONTENTS

SUMMARY

SEVERITY	NUMBER OF FINDINGS
CRITICAL	0
HIGH	0
MEDIUM	2
LOW	2
INFORMATIONAL	2

TOTAL: 6

SCOPE

The analyzed contracts are located in:
https://github.com/maticnetwork/pos-portal/commit/a70d8b6dbd867cd45c485746b2b235f593b7494e

CONFIDENTIAL

LIMITATIONS ON DISCLOSURE AND USAGE OF THIS REPORT

WEAKNESSES

This section contains the list of discovered weaknesses.

1. MISSING INCORRECT SIGNER CHECK IN PERMIT FUNCTION

SEVERITY: Medium

REMEDIATION: add a zero address check in the permit function

STATUS: fixed

DESCRIPTION:

The builtin **ecrecover()** function on the line 47 in the file **UChildDAI.sol** returns **address(0)** if the signature is malformed, thus there is a possibility to approve any amount for the zero address. As there is no impact with the current **ERC20** implementation that is being used by **UChildDAI,** the severity is lowered.

```
require(holder == ecrecover(digest, v, r, s), "UChildDAI: INVALID-PERMIT");
        require(expiry == 0 || now <= expiry, "UChildDAI: PERMIT-EXPIRED");
        require(nonce == nonces[holder]++, "UChildDAI: INVALID-NONCE");
        require(msg.sender != address(this), "UChildDAI:
PERMIT_META_TX_DISABLED");
        uint wad = allowed ? uint(-1) : 0;
        _approve(holder, spender, wad);
```

2. DISCREPANCY IN PARSING OF LISTS IN RLPREADER AND BOR

SEVERITY: Medium

REMEDIATION: make sure that short lists and long lists are parsed in the same manner in RLPReader and Bor

STATUS: acknowledged

DESCRIPTION:

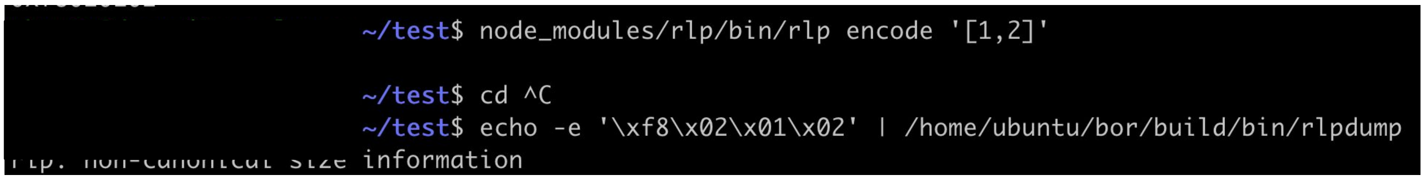
There are two ways of encoding lists in **RLP**:

- If the total payload of a list is 0-55 bytes long, the **RLP** encoding consists of a single byte with value `0xc0` plus the length of the list and **RLP** encoded items
- If the total payload of a list is more than 55 bytes long, the **RLP** encoding consists of a single byte with value `0xf7` plus the length in bytes of the length of the payload in binary form, followed by the length of the payload, followed by **RLP** encoded items

It is possible to encode a short list (<55 bytes long) using an encoding for long lists:

```
'[1,2]' → 0xc20102   // short list encoding
'[1,2]' → 0xf8020102 // long list encoding
```

RLPReader used in Polygon smart contracts recognises short lists encoded as long lists and returns correct results. However, RLP parser in Bor does not parse such payloads:



```
[redacted]~/test$ node_modules/rlp/bin/rlp encode '[1,2]'
[redacted]
[redacted]~/test$ cd ^C
[redacted]~/test$ echo -e '\xf8\x02\x01\x02' | /home/ubuntu/bor/build/bin/rlpdump
rlp: non-canonical size information
```

This discrepancy may lead to unexpected behaviours when the same transaction coming from an adversary is parsed with RLPReader and Bor.

3. MISSING EXPLICIT HANDLING OF ALL CONTROL FLOW BRANCHES

SEVERITY: Low

REMEDIATION: add an explicit revert or return false after the "for" statement

STATUS: fixed

DESCRIPTION:

The path of reaching the code after the for-loop on line 94 in the file **MerklePatriciaProof.sol** should be unreachable and it is better not to depend on the compiler/optimizer cleaning up the stack correctly for the function to be guaranteed to return false in that case.

```
pathPtr += traversed;
                nodeKey = bytes32(RLPReader.toUintStrict(currentNodeList[1]));
            } else {
                return false;
            }
        }
    }
```

4. MISSING LOGIC

SEVERITY: Low

REMEDIATION: add the logic for more consistent security review

STATUS: fixed

DESCRIPTION:

The **_processMessageFromRoot()** implementation for child and root tunnels on line 6 in the files **ChildTunnel.sol** and **RootTunnel.sol** are missing, although the function is part of a crucial mechanism for message transmission between the chains.

```
contract ChildTunnel is BaseChildTunnel {
  function _processMessageFromRoot(bytes memory message) internal
override {
    // implement your core logic here
  }
}
```

5. GAS OPTIMISATION

SEVERITY: Informational

REMEDIATION: use a regular increment instruction

STATUS: fixed

DESCRIPTION:

As the nonces are initialized with 0 values, the increment operation is practically impossible to overflow, since it will take 2^256 calls for every signer on line 50 in the file NativeMetaTransaction.sol.

```
nonces[userAddress] = nonces[userAddress]. add(1);
```

6. UNUSED RETURN VALUE

SEVERITY: Informational

REMEDIATION: consider removing the variable if there is no usage for it

STATUS: fixed

DESCRIPTION:

The variable createdAt that is being fetched from **_checkpointManager.headerBlocks** is being returned by the **_checkBlockMembershipInCheckpoint()** function on line 399 and 423 in the file **RootChainManager.sol**, although the return value is never being used.

```
_checkBlockMembershipInCheckpoint(
        payload.getBlockNumber(),
        payload.getBlockTime(),
        payload.getTxRoot(),
        payload.getReceiptRoot(),
        payload.getHeaderNumber(),
        payload.getBlockProof()
    );
```

hexens