

Auditing System by ISUNCLOUD

Currently, smart contracts are deployed using Remix, and data is input directly into the deployed smart contracts via Etherscan or Remix.

1. System Deployment:

Deploy the smart contracts in the following sequence:

1. parser.sol
2. reports.sol
3. TransactionContract.sol (parser address)
4. Handlers Contracts (TransactionContract address, parser address)
5. Calculating Contracts (TransactionContract address, Parser address , report address)
6. getTransactionTimeSpan.sol (transaction address, parser address, functioning contracts addresses)

2. Operation Flow:

1. Users start by registering handlers using the `registerHandlers` function in the **"TransactionContract"**, inputting the **"TransactionType"** (of type `bytes32`) and the **"handler's"** address (of type `address`).

A screenshot of a debugger's call stack. The top frame is titled 'registerHandler' and shows two arguments: 'transactionType' with the value '0x4530303031303030310000' and 'handler' with the value '0xc99afd3f66AA5b2C16Ea24d2'. Below the arguments are two buttons: 'Calldata' and 'Parameters'. At the bottom right is an orange button labeled 'transact'.

2. Record data using a **bytes32** array in the **addRecord** function of the “**TransactionContract.sol**” contract, where each element has been multiplied by 10^{18} . The first element must be the eventID, and the second should specify the event type. Users must omit the timestamp column to prevent fraudulent events; the system will automatically record the current time.”

For example, for the following transaction, the format of the array should be:

[illegible]

	Events
出入金	E00010001 EP001 用戶入金 10000 USDT EP002 內扣手續費 10 USDT EP003 外扣手續費 1 USDT EP004 交易時間 t1 EP005 交易時匯率 1 USDT = 1.01 USD

Which in decimal:

[
"first",	(eventID)
"E00010001",	(transactionType)
1000000000000000000000,	(EP001)
1000000000000000000000,	(EP002)
1000000000000000000000,	(EP003)
1010000000000000000000	(EP005)
]	

, notice that every number has been multiplied by 10^{18} .

```
0x0000000000000000000000000000000000000000000dbd2fc137a30000,  
0x000000000000000000000000000000000000000000056bc75e2d63100000,  
0x0000000000000000000000000000000000000000000000000000000006590,  
0x66697273745f7265706f727440000000000000000000000000000000000000
```

```
[
99000000000000000000,      (SP002)
1600000000000000000000,    (SP003)
2600000000000000000000,    (SP004)
"first_report"                (reportID)
]
```

setRate

_SP002:

bytes32

_SP003:


bytes32


_SP004:

bytes32

_reportID:

bytes32

 Calldata

 Parameters


transact


filterTransactionsInRange

startTime: 1699497948

endTime: 1699498949

_reportID: 0x66697273745f7265706f7274

 Calldata

 Parameters

transact

4. Then the system will pass those transactions which had been filtered in a time span to calculating functions (use eventID to determine which one), the calculating functions first use “**lparser.sol**” function to change **bytes32** into string or **int256** and then calculate data with planned formula.
5. After calculating calculating get a 3D array from “**lreports.sol**”, and then add results into the respective column.
6. We can check the numbers is correct or not by calling the function **getValue(reportID, reportType, reportColumn)** in reports.sol. There are three options to fill in the reportType, ‘**balanceSheet**’, ‘**comprehensiveIncome**’, ‘**cashFlow**’. Then input the ‘**reportColumn**’ to check the respective column.

getValue

reportID:

first_report

reportType:

balanceSheet

reportColumn:

assets.details.cryptocurrency.total

Calldata

Parameters

call

0: int256: 9900990000000000000000

getValue

reportID:

first_report

reportType:

comprehensiveIncome

reportColumn:

income.details.depositFee.weighte

Calldata

Parameters

call

0: int256: 1111000000000000000000

getValue

reportID:

first_report

reportType:

cashFlow

reportColumn:

supplementalScheduleOfNonCasl

Calldata

Parameters

call

0: int256: 1008990000000000000000

3. Interact with smart contracts on Ethereum and checking results

1. First, users should download and deploy the “hardhat” environment locally .
2. Run “***npm hardhat run transformReportAPI.js***”.
3. This program would help users to interact with reports.sol on ethereum by getting reports data.
4. The program will parse the raw data into planned API format.

```
{
  "success": true,
  "code": "00000000",
  "reason": "ERROR_MESSAGE.SUCCESS",
  "data": {
    "id": "first_report",
    "date": 1699411588439,
    "totalAssetsFairValue": 9900.99,
    "totalLiabilitiesAndEquityFairValue": 9900.99,
    "assets": {
      "totalAmountFairValue": 9900.99,
      "details": {
        "accountsReceivable": {
          "totalAmountFairValue": 0,
          "weightedAverageCost": "?",
          "breakdown": {
            "ETH": {
              "currencyType": "CRYPTOCURRENCY",
              "name": "ETH",
              "amount": "?",
              "weightedAverageCost": "?",
              "fairValue": "?"
            },
            "BTC": {
              "currencyType": "CRYPTOCURRENCY",
              "name": "BTC",
              "amount": "?",
              "weightedAverageCost": "?",
              "fairValue": "?"
            },
            "USDT": {
              "currencyType": "CRYPTOCURRENCY",
              "name": "USDT",
              "amount": 0,
              "weightedAverageCost": "?",
              "fairValue": 0
            },
            "USD": {
              "currencyType": "FIAT",
              "name": "USD",
              "amount": "?",
              "weightedAverageCost": "?",
              "fairValue": "?"
            }
          }
        }
      }
    }
  }
}
```

4. Future Plans:

1. Develop a standardized interface for calculation functions and store it in the interfaces file.
2. Code each function up to Column 7, encompassing all types of deposits and withdrawals.
3. Implement Mistake Proofing mechanisms, such as require statements.
4. Enhance calculation accuracy, given that Solidity lacks a float variable type.
5. Optimize gas fees through careful data structure design.
6. Create an interface for reports.sol and import only the interface.