GoToken 合约及操作记录

目录

1 介绍

本文记录了 GoToken 合约情况，以及多签操作记录。个人的操作行为不在本文记录的范围内。

2 工厂合约及地址

1. 0x38c931c4064459468d4F6ED0Ae16F4Ba433Aa03F，多签工厂合约，用于生成多签合约，已废弃。
2. 0xe6D887f548b48bF30b06a68a0aaE06F62A525032，多签工厂合约，用于生成多签合约。
3. 0x98D5EC13d08E5796De7191c1719D6D5bDec4a74c，可信列表工厂合约，用于生成一个可信的列表。
4. 0x5004e5C6b2D2009eB273bBaB45E5a756e2f729E5，ERC20 代币工厂合约，用于生成一个 ERC20 代币。
5. 0xfcD4bfFd7B76092c59Fb9791ff3F9B5405D4010A，投资及分发工厂合约，业务相关。
6. 0x5b0D747aF4AF70d8022b1Cbe4Fa27010B3C3AD6d，初始锁仓工厂合约，用于生成一个锁仓 ERC20 的合约。
7. 0x649C0bA9D1454fA22331B2e801505a234504B8BE，GT 项目工厂合约，用于生成一个 GT 项目的合约。

3 实例合约及地址

注意，本节仅描述了构造时使用的参数，因此随着后面的操作，本节描述的地址可能发生变化，以链上地址为准

GoToken 的合约由三个合约组成，分别为

1. 0x9FDB24Df185b4e6c42846c2f1355cA0A2BB7e043，多签合约，用于管理 GTToken 合约的信任合约列表;已变更为0x8e61F120beAe3CF91cC9a017AC48844348e3b34A，构造参数为

```
(['0xe855B4cb17eA22CAE1be5FeB7fCDC0Ef67DCa84D',
  '0x3e6F107Fd4A95AF86108c1F44E502A6136AD386e',
  '0x57955d7AA271DbDDE92D67e0EF52D90c6E4089cA'])
```

2. 0xBEb0FAE8c75c79e5f92c1C23C6435bD0509c276e，GTToken 的信任合约列表，使用多签操作；构造参数为

```
([],
 0x9FDB24Df185b4e6c42846c2f1355cA0A2BB7e043) //多签地址，已变更
```

3. 0x353214343Ee192AD8a58C62961B972F4d5a6877E，GTToken 合约，这是一个标准的 ERC20 合约，用来完成代币的管理、转账等操作；构造参数为

```
('0x0000000000000000000000000000000000000000',
 0, "GoToken", 6, "GOO", true,
 0x9FDB24Df185b4e6c42846c2f1355cA0A2BB7e043, //多签地址，已变更
 0xBEb0FAE8c75c79e5f92c1C23C6435bD0509c276e) //信任列表地址
```

GTToken 合约的增发及销毁必须通过信任合约列表中的合约进行，目前在信任列表中存在着两个合约，

1. 0xAA9a51f48834924B2F79639Af74FefE9BFF74529 FundAndDistribute 合约，用于完成铸币及代币分发操作；构造参数为

```
(0x353214343Ee192AD8a58C62961B972F4d5a6877E, //GTToken地址
 "USDT for GOO", "Only for Funders",
 0xdAC17F958D2ee523a2206206994597C13D831ec7,  //USDT地址
 0x9FDB24Df185b4e6c42846c2f1355cA0A2BB7e043,  //多签地址，已变更
 0x735451974A28f63b0593cCe91696659c1B900380)  //信任列表地址
```

注意，此处的信任地址列表为投资人白名单，虽然内容上与 GTToken 的信任合约列表一样，但是构造参数并不同，其构造参数为:

```
(['0xe855B4cb17eA22CAE1be5FeB7fCDC0Ef67DCa84D',
  '0x3e6F107Fd4A95AF86108c1F44E502A6136AD386e',
  '0x57955d7AA271DbDDE92D67e0EF52D90c6E4089cA'], //投资人列表
 0x9FDB24Df185b4e6c42846c2f1355cA0A2BB7e043) //多签地址
```

2. 0x491c8a58DDb5a6380FeA7fE8909788f681453B00 InitLock 合约，用于完成初始锁仓功能；构造参数为

```
(0x353214343Ee192AD8a58C62961B972F4d5a6877E, //GTToken地址
 9932664,  //解锁区块高度，大约锁仓6个月
 ['0xe855B4cb17eA22CAE1be5FeB7fCDC0Ef67DCa84D',
  '0x3e6F107Fd4A95AF86108c1F44E502A6136AD386e',
  '0x57955d7AA271DbDDE92D67e0EF52D90c6E4089cA'], //解锁分配地址
 [30000000000, 10000000000, 10000000000],   //解锁分配数量
 0x9FDB24Df185b4e6c42846c2f1355cA0A2BB7e043) //多签地址
```

不难发现，上述所有合约使用的多签合约为同一个合约地址，这仅仅是为了方便初期的管理，并不是为了限制。每个合约都可以实用 `transfer_multisig` 方法修改其对应的多签合约。

4 操作记录

2019-10-28

- `add_multi_trusted(`

  `1,`

  `['0xAA9a51f48834924B2F79639Af74FefE9BFF74529', //FundAndDistribute合约`

  `'0x491c8a58DDb5a6380FeA7fE8909788f681453B00'  //InitLock合约`

  `])`

  在 GTToken 合约中增加两个信任合约。

- `transfer(`

  `1,`

```
  0x3e6F107Fd4A95AF86108c1F44E502A6136AD386e,
  1000000)
```

实验性的将一个 GoToken 打出（已经在事后打回并销毁）。

2019-11-17

- 对新的多签工厂合约调用，
  0xe6D887f548b48bF30b06a68a0aaE06F62A525032

  ```
  createMultiSig(['0xe855B4cb17eA22CAE1be5FeB7fCDC0Ef67DCa84D',
    '0x3e6F107Fd4A95AF86108c1F44E502A6136AD386e',
    '0x57955d7AA271DbDDE92D67e0EF52D90c6E4089cA'])
  ```

  返回地址为0x8e61F120beAe3CF91cC9a017AC48844348e3b34A，为 GT 系统新的多签合约；
  0x31288e10fc62a675c76e864b09b72146f156387ab60a990215c1260b81db498e

- 修改 GTToken 的信任列表合约的多签地址，
  0xBEb0FAE8c75c79e5f92c1C23C6435bD0509c276e

  ```
  transfer_multisig(1, 0x8e61F120beAe3CF91cC9a017AC48844348e3b34A)
  ```

- 修改 GTToken 合约的多签地址，
  0x353214343Ee192AD8a58C62961B972F4d5a6877E

  ```
  transfer_multisig(2, 0x8e61F120beAe3CF91cC9a017AC48844348e3b34A)
  ```

- 修改 FundAndDistribute 合约的多签地址，
  0xAA9a51f48834924B2F79639Af74FefE9BFF74529

  ```
  transfer_multisig(3, 0x8e61F120beAe3CF91cC9a017AC48844348e3b34A)
  ```

- 修改 FundAndDistribute 白名单合约的多签地址，
  0x735451974A28f63b0593cCe91696659c1B900380

  ```
  transfer_multisig(4, 0x8e61F120beAe3CF91cC9a017AC48844348e3b34A)
  ```

2019-11-28

- 创建新的多签合约，用于管理 FundAndDistribute 白名单，与其他多签合约分离，

```
0xe6D887f548b48bF30b06a68a0aaE06F62A525032
    .createMultiSig(
        ['0xe855B4cb17eA22CAE1be5FeB7fCDC0Ef67DCa84D',
        '0x3e6F107Fd4A95AF86108c1F44E502A6136AD386e',
        '0x57955d7AA271DbDDE92D67e0EF52D90c6E4089cA',
        '0x0aCA6913289d7887983F26d4CBF7ACF4B104D03f'])
```

返回的合约地址为0x86880095C6A4Dccef7e8dBdB8A864d417A37b7EA;

- 修改 FundAndDistribute 白名单合约的多签地址，

```
0x735451974A28f63b0593cCe91696659c1B900380
    .transfer_multisig(
        1,
        0x86880095C6A4Dccef7e8dBdB8A864d417A37b7EA)
```

- 在 FundAndDistribute 白名单中增加地址，

```
0x735451974A28f63b0593cCe91696659c1B900380
    .add_trusted(
        1,
        0x0aCA6913289d7887983F26d4CBF7ACF4B104D03f)
```

- 修改 FundAndDistribute 的多签地址，

```
0xAA9a51f48834924B2F79639Af74FefE9BFF74529
    .transfer_multisig(
        2,
        0x86880095C6A4Dccef7e8dBdB8A864d417A37b7EA)
```

附录 A　操作帮助

本节对于常见的操作给出通用的合约操作方法，本节描述的方法仅针对直接操作合约，以 EtherScan 为例。

A.1　查看多签时使用的 id

在进行多签操作时，通常需要一个 id，此 id 用于标识试图通过达成的共识的编号，该编号有以下特点：

1. 该 id 必须是从 1 开始连续递增的，不能跳过编号；
2. 一个多签人仅能对同一个 id 进行一次表决；
3. 该 id 的编号对于不同的函数是独立的，互相不影响；
4. 该 id 的编号是由多签合约提供的，因此多签合约更换时，相应的 id 需要重新开始编号。

综合上述特征，不难发现，在多签操作前，取得正确的 id 是很重要的。

对于任一非多签合约，应该都可以在 EtherScan 中找到其对应的多签合约（合约中的 multisig_contract 地址），在多签合约中，调用 (read contract)get_unused_invoke_id，填入要进行多签的函数名，就可以得到相应的 id。

A.2　如何进行铸币

铸币的前提是铸币人需要在铸币白名单中。

铸币操作分为两步：

1. 调用 USDT 合约的 approve 方法，需要注意的是，其中的 spender 必须为 FundAndDistribute 合约的地址，数量必须大于需要铸币的数量，例如

```
0xdAC17F958D2ee523a2206206994597C13D831ec7
    .approve(
        0xAA9a51f48834924B2F79639Af74FefE9BFF74529,
        10000000000)
```

其中，调用的地址为 USDT 合约的地址。

2. 调用 FundAndDistribute 合约的 `fund` 方法，需要注意的是，由于该方法需要进行多次转账，因此需要适当调高 gas 费用。