矩阵元区块链系统

DApp 开发手册

（v1.0）

文档版本号：	1.0	文档编号：	001
文档密级：	公开	归属部门/项目：	区块链研发组
产品名：	矩阵元区块链	系统名：	矩阵元区块链 1.0
编写人：	区块链项目组	编写日期：	2017-9-5

修订记录：

版本号	修订人	修订日期	修订内容

V1.0	区块链项目组	2017.9.5	初稿完成

所有权声明

目 录

引言

1. 读者对象

本文从矩阵元提供的一个区块链 Demo 入手，描述了基于矩阵元区块链进行 Dapp 开发的具体过程与步骤。以方便在矩阵元区块链上进行 Dapp 开发的的开发者进行参考。

2. 缩略语和术语

缩略语/术语	全 称	说 明
DApp	Decentralized Application	去中心化的应用

3. 读者预备知识

相关人员在进行安装配置时，必须先具备如下知识：

- Linux 操作系统的基本配置及操作管理知识；
- 了解矩阵元区块链系统的基本功能特性；
- 对基于区块链的智能合约编写有一定的经验；
- 对基于区块链的 Web 应用开发有一定的经验；

4. 使用约定

编写时统一使用一些特定的符号或格式，定义如下：

1. **黑体/黑斜体**：表示安装目录
2. 正常斜体：表示安装软件时需检查确认的部分
3. 红色字体：表示需重点注意的地方
4. 灰底字体：表示执行指令或者代码

1. 开发环境搭建

1.1. 区块链平台搭建

参考**《云平台 V1.3.0 用户手册-v0.6》**

1.2. 发布环境搭建

环境要求
+ Ubuntu16.04
+ NodeJS 6.0+ recommended.
+ npm 安装
+ truffle2.1.1 安装

1.2.1. git 安装

```
$ sudo apt-get install git
```

1.2.2. nodejs 安装

请确保安装的 nodejs 版本在 6.0+，此处通过离线包方式安装

```
$ sudo apt-get remove nodejs
$ sudo mkdir -p /opt/package && cd /opt/package
$ sudo wget https://nodejs.org/dist/v6.11.3/node-v6.11.3-linux-x64.tar.xz
$ sudo tar -xvf node-v6.11.3-linux-x64.tar.xz && mv node-v6.11.3-linux-x64 /usr/local/nodejs
$ sudo ln -s /usr/local/nodejs/bin/node /usr/bin/node
$ sudo ln -s /usr/local/nodejs/bin/npm /usr/bin/npm
```

1.2.3. npm 安装

```
# 配置国内镜像源，用于提速
$ sudo npm config set registry https://registry.npm.taobao.org
$ sudo npm config list
```

如图：正确安装 npm 的姿势，请确保镜像地址配置成功，后续安装速度很重要

```
juzhen@blockchain9:~$ npm -v
3.5.2
juzhen@blockchain9:~$ sudo npm config set registry https://registry.npm.taobao.org
juzhen@blockchain9:~$ sudo npm config list
; cli configs
user-agent = "npm/3.5.2 node/v4.2.6 linux x64"

; userconfig /home/juzhen/.npmrc
registry = "https://registry.npm.taobao.org/"

; builtin config undefined
globalconfig = "/etc/npmrc"
globalignorefile = "/etc/npmignore"
prefix = "/usr/local"

; node bin location = /usr/bin/nodejs
; cwd = /home/juzhen
; HOME = /home/juzhen
; "npm config ls -l" to show all defaults.
```

1.2.4. solc 安装

```
$ sudo npm install -g solc@0.4.11
$ sudo ln -s /usr/local/lib/node_modules/solc/solcjs    /usr/bin/solc
$ solc --version
```

1.2.5. truffle 安装

```
$ sudo mkdir -p /opt/package && cd /opt/package
$ sudo wget https://github.com/zjsunzone/truffle/releases/download/v2.1.1/truffle-2.1.1.tar.gz
$ sudo tar -zxvf    truffle-2.1.1.tar.gz
$ sudo mkdir -p /usr/local/lib/node_modules
$ sudo cp -rp /opt/package/truffle /usr/local/lib/node_modules/
$ sudo ln -s /usr/local/lib/node_modules/truffle/cli.js /usr/local/bin/truffle
$ sudo ln -s /usr/local/lib/node_modules/truffle/exec.js /usr/local/bin/truffle-exec
```

1.2.6. nginx 安装

温馨提示：Nginx 的安装会较耗时，请耐心等待，如果没有目录的特别习惯，可以直接拷贝命令一句一句执行。

依赖说明：
gzip 模块需要 zlib 库
rewrite 模块需要 pcre 库
ssl 功能需要 openssl 库

1.1. 安装 pcre

```
$ sudo mkdir -p /opt/package && cd /opt/package
$ sudo wget https://jaist.dl.sourceforge.net/project/pcre/pcre/8.41/pcre-8.41.tar.gz
$ sudo tar -zxvf pcre-8.41.tar.gz && cd pcre-8.41
$ sudo ./configure --prefix=/usr/local/pcre
$ sudo make
$ sudo make install
```

1.2. 安装 openssl

```
$ sudo mkdir -p /opt/package && cd /opt/package
$ sudo wget https://www.openssl.org/source/openssl-1.0.2l.tar.gz
$ sudo tar -zxvf  openssl-1.0.2l.tar.gz && cd openssl-1.0.2l
$ sudo ./config --prefix=/usr/local/openssl
$ sudo make
$ sudo make install
```

1.3.安装 zlib

```
$ sudo mkdir -p /opt/package && cd /opt/package
$ sudo wget https://jaist.dl.sourceforge.net/project/libpng/zlib/1.2.11/zlib-1.2.11.tar.gz
$ sudo tar -zxvf  zlib-1.2.11.tar.gz && cd zlib-1.2.11
$ sudo ./configure --prefix=/usr/local/zlib
$ sudo make
$ sudo make install
```

1.4.安装 nginx

```
$ sudo mkdir -p /opt/package &&cd /opt/package
$ sudo wget https://nginx.org/download/nginx-1.12.1.tar.gz
$ sudo tar -zxvf  nginx-1.12.1.tar.gz && cd nginx-1.12.1
$ sudo ./configure --prefix=/usr/local/nginx --with-pcre=/opt/package/pcre/pcre-8.41
$ sudo make
$ sudo make install
```

说明：若安装时找不到上述依赖模块，使用--with-openssl=、--with-pcre=、--with-zlib=指定依赖的模块目录；

至此，对 web 支持的服务器部署完成，如果仅测试 demo 使用默认配置即可;

推荐配置：/usr/local/nginx/conf/nginx.conf

```
worker_processes  4;
worker_cpu_affinity 0001 0100 1000 0010;
error_log  logs/error.log  notice;
pid        nginx.pid;
```

```
events {
    worker_connections  1024;
}

http {
    include       mime.types;
    default_type  application/octet-stream;
    log_format    main  '$remote_addr - $remote_user [$time_local] "$request" '
                        '$status $body_bytes_sent "$http_referer" '
                        '"$http_user_agent" "$http_x_forwarded_for"';
    access_log    logs/access.log  main;

    sendfile        on;
    keepalive_timeout  65;
        send_timeout 10s;
        client_max_body_size 8m;



        # define upstream
        #upstream proxy_monitorWebEngine {
        #       server 192.168.9.18:10002;
        #}

    server {
        listen       80;
        server_name  localhost;
        charset utf-8;
        access_log  logs/host.access.log  main;

        #location /monitor_web {
        #    proxy_pass  http://proxy_monitorWebEngine/monitor_web;
        #    proxy_redirect              default;
        #    proxy_set_header            Host $host:$proxy_port;
        #    proxy_set_header            X-Real-IP $remote_addr;
        #    proxy_set_header            X-Forwarded-For $proxy_add_x_forwarded_for;
        #}

        location / {
            root   html;
            index  index.html index.htm;
        }

        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
```

```
            root    html;
        }
    }
}
```

2. Demo 说明

参考资料：
《矩阵元股权交易产品用户手册》
https://github.com/Juzix/wiki/blob/master/矩阵元股权交易产品用户手册.pdf

2.1. 代码获取

继续下列操作时，请确保已完成环境搭建，参考：章节 1.开发环境搭建

* DAPP 合约代码获取

```
$ git clone https://github.com/Juzix/BlockChain-Demo.git
```

说明：推荐使用 gitbash 进行 git 代码管理

* DAPP-Web 代码获取

```
$ wget https://github.com/Juzix/BlockChain-Demo/releases/download/v0.1.0/dapp-token.zip
```

说明：web 代码为纯前端 `html+js` 组成，结合 web3.js 与节点进行交互

* 移动 APP 下载 二维码

```
Url: https://github.com/Juzix/BlockChain-Demo/releases/download/v0.1.0/Juzix-app-token.apk
```

说明：APP 为移动端原生 APP，承载 dapp-web 的装载，使用 app 可完成对 dapp-web 的操控；

2.2. 代码解读

功能说明：
* 实现了一个基本的股权交易合约
* 基于该合约能够实现基本的股权登记与非交易过户等功能特性
* 采用零知识证明与同态加密算法实现用户股权账户的隐私保护
* 初步呈现了矩阵元联盟链在交易执行、业务查询、隐私保护等方面的功能

合约代码

app/	- truffle 测试 WEB 程序
contracts/	- 合约目录
- interfaces/	- 合约接口定义（抽象合约）
- IJuzixTokenManager.sol	- 接口:股权交易
- IRegisterManager.sol	- 接口：注册中心
- IRoleFilterManager.sol	- 接口：角色过滤器
library/	- Libray 库目录
- LibTokenPailler.sol	- Lib:合约信息 Struct
- LibTokenRecord.sol	- Lib:交易记录 Struct
nizk/	- 同态加密 Lib 库集
- LibNIZK.sol	
- LibNizkParam.sol	
sysbase/	- 系统合约目录
- BaseModule.sol	- 基础模块合约，所有模块需要继承该合约
- OwnerNamed.sol	- 所有业务合约都需继承该合约
token/	- ERC2.0 接口标准
- BasicToken.sol	
- ERC20.sol	
- ERC20Basic.sol	

```
- StandardToken.sol
utillib/                                    - 系统工具合约
- LibDB.sol
- LibDecode.sol
...
JuzixTokenManager.so                        - 股权交易主合约
TokenModuleManager.sol                      - 股权 DAPP 对应模块
truffle.js                                  - truffle 配置文件
package.json                                - npm 包配置文件
install.js                                  - 合约发布程序
```

说明：

以上为 github 下载的代码目录结构，重点介绍如下：

* interfaces

    合约接口定义目录，主要用于存放定义合约的接口；

* library

    struct 的库解析目录，所有业务 struct 结构的解析存放于此；

* nizk

    主要存放了封装零知识证明的工具函数库；

* sysbase

    矩阵元联盟链合约编写规范的合约定义及规范，根据场景继承使用；

* utillib

    基于矩阵元联盟链的工具库封装，主要包含字符串操作、整形数据操作、编解码、对 levelDB 操作、JSON 数据格式解析等；

* token

    ERC2.0 标准定义

* JuzixTokenManager.sol

    股权交易的主逻辑合约

* TokenModuleManager.sol

    股权交易模块（一个模块==DAPP）

2.3. 使用说明

在正事开始使用前请确保已完成如下操作：

* 已成功部署云平台，参考**《云平台 V1.3.0 用户手册-v0.6》**
* 下载移动 APP 应用程序，并注册一个用户，参考**《移动 APP 使用手册-v1.0》；**
* 注册成功后，获取新注册用户的钱包地址；（后续步骤需要使用此地址）

2.4. 应用部署

提示：合约部署前请确认已完成基本的环境搭建

2.4.1. DAPP 部署

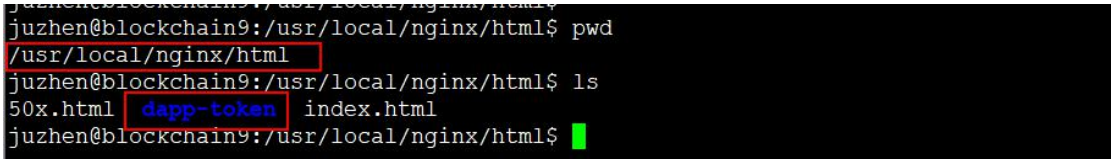
说明：应用采用 app+web 混合应用方式使用，移动 APP 加在在线的 web 应用，好比浏览器访问 web 页面进行操作。以下为 Web 的部署方式；

Nginx 部署完成后，使用默认配置即可。将下载的 dapp-web 代码放置在/usr/local/nginx/html 目录，启动 nginx 则表示发布成功；

```
$ cd /usr/local/nginx/sbin
$ ./nginx                    # 启动 nginx
```

说明：默认 DAPP-Web 的文件名为：dapp-token ,目录结构如下：

```
html/
- dapp-token
   - index.html
   - static/
      - css
      - js
      - fonts
      - images
```



则访问地址为：**http://xxx.xxx.xxx.xxx:port/dapp-token/#**

Nginx 常用命令介绍

```
$ cd /usr/local/nginx/sbin
$ ./nginx -s stop            # 停止
$ ./nginx -s reload          # 平滑重载配置文件
```

此步骤结束后请牢记 dapp-token 的可访问地址，后续步骤中需要使用

2.4.2. 合约修改

在合约发布前需要将交易模块（TokenModuleManager）中的访问地址进行更改

```
43
44          string memory _json = "{";
45          _json = _json.concat("\"moduleId\":\"", sysModuleId, "\",");
46          _json = _json.concat("\"moduleName\":\"", moduleName, "\",");
47          _json = _json.concat("\"moduleText\":\"", moduleText, "\",");
48          _json = _json.concat("\"moduleVersion\":\"", moduleVersion, "\",");
49          _json = _json.concat("\"icon\":\"
50          _json = _json.concat("\"moduleEnable\":", "0", ",");    //模块开关：0-放权；1-控权【需要修改】
51          _json = _json.concat("\"moduleType\":", "1", ",");      // 模块类型 1 系统模块
52          _json = _json.concat("\"moduleUrl\":\"http://192.168.9.18/dapp-token/#\"", ",");
53          _json = _json.concat("\"moduleDescription\":\"", moduleName.concat(moduleVersion, "-Token代币模块"
54          _json = _json.concat("}");
55
56          uint ret = addModule(_json);
57          sysModuleId = ret.recoveryToString();
58          log("construct sysModuleId:",sysModuleId);
59
60      }
61
```

如图：更新 moduleUrl 的访问地址为 web 部署的地址：

http://xxx.xxx.xxx.xxx:port/dapp-token/#

注意：此地址为移动 APP 对在线 web 访问的地址。

2.4.3. 合约部署

在 git 上获取到代码如图：

```
juzhen@blockchain9:~/test$ ls
BlockChain-Demo
juzhen@blockchain9:~/test$
```

进入到 BlockChain-Demo 目录完成下列操作：

* 安装 nodejs 包依赖

```
$ npm install
```

* 编译合约

```
$ truffle compile
```

如果出现无法编译的情况，可使用 node 带参数设置方式:

```
$ node --max_old_space_size=2000000 /usr/local/bin/truffle
```

建议为以上命令设置别名，便于方便使用:

```
$ echo   "alias   mtruffle='node --max_old_space_size=2000000 /usr/local/bin/truffle'" >> ~/.bash_aliases
$ source ~/.bash_aliases
$ mtruffle version
```

新的合约编译方式为：

```
$ mtruffle compile
```

* 合约发布

```
$ node install.js --passwd 11111111 --url http://127.0.0.1:6789 --address
0xa5efb0582443bc5b77f707cea850ef099e3068cc
```

--passwd	临时钱包口令，用于发布合约
--url	矩阵元联盟链节点 JSON-RPC 访问地址
--address	APP 上注册成功的用户钱包地址

说明：

1、address 参数务必仔细填写，在合约发布完成后会自动重置合约的归属者，此地址会被设置会合约的 owner 地址；

2、发布合约前请务必先编译合约；

3. SDK 说明

3.1. 智能合约接口

~/BlockChain-Demo/contracts/sysbase 目录特别声明：

BaseModule.sol

此合约为模块基础合约，所有新模块的编写必须继承此合约。该合约主要完成了模块数据的新增，模块下合约的新增、模块角色新增以及权限的新增。新的模块数据如果要发送到矩阵联盟链则必须通过此合约完成；

OwnerNamed.sol

此合约为所有合约的基类，任何一个新增的合约都要继承该合约。OwnerNamed 主要提供合约公用的函数功能，主要包括对合约的注册、日志输出、鉴权数据同步等操作。如果您希望自定义实现一个合约并发布到链上，务必继承该合约；

~/BlockChain-Demo/contracts/utillib 目录特别声明：

该目录下为封装的函数 Lib 库，主要为工具类库，包含了对整数、字符串数据的转换处理，同时包含对 DB、日志、堆栈等数据的操作；

~/BlockChain-Demo/contracts/interfaces 目录特别声明：

此目录主要用于存储合约的接口定义，接口中定义的所有函数实现的合约必须全部实现，否则派生合约也为抽象合约；

~/BlockChain-Demo/contracts/nizk 目录特别声明：

LibNiz 封装了零知识证明的相关操作，包含生成零知识证明结构、密文数据相加、密文数据相减；

3.2. Java 开发接口

N/A

3.3. Javascript 开发接口

N/A

4. 附录

4.1. 安装问题

Nodejs

安装完成后执行命令：`node -v` 查看是否安装成功，如果没有找到 node 命令，可以使用：

```
$ ln -s /usr/bin/nodejs /usr/bin/node
```

4.2. 开发问题

4.3. 参考文档

* Solidity 智能合约开发指南【英文版】《https://solidity.readthedocs.io》
* Solidity 智能合约开发指南【中文版】《http://www.tryblockchain.org》