EOS调研结果

> Author: huziang
>
> Date：2018年5月13日 星期天 下午9:48

EOS的背景调研：

1. EOS是block.one开发的一款区块链操作系统，和比特币，以太坊的底层实现不同。目前进行到dawn3.0版本。

   block.one官网：https://block.one/

   EOS官网：https://eos.io/

   github地址：https://github.com/EOSIO/eos/

2. EOS上开发的Dapp基本全部属于设计和研发阶段，因此EOS目前的作用仅限于EOS货币。

   下图是EOS上开发的一些Dapp：（基本都没查到相关信息，有些官网都改了）

序号	DAPP名称	项目内容简介	项目进展	主理人	地点位置	网站
1	Everipedia	去中心化维基百科	少许进展	Mahbod Moghadam	加利福尼亚，美国	everipedia.io
2	Iryo	去中心化医疗保健服务	未知	Tjaša Zajc	斯洛文尼亚	medium.com/iryo-network
3	scatter	EOS扩展钱包	概念验证	Nathan James	洛杉矶，美国	scatter-eos.com
4	OracleChain 欧链	OracleChain作为全球第一个直面区块链生态Oracle（预言机）需求的基础应用，将区块链技术服务和现实生活中的多种需求场景直接高效对接，深耕这个百亿美金估值的巨大市场。	截止目前，欧链科技累计向国家知识产权局完成35项区块链发明专利的申请提交。	赵微	北京，中国	oraclechain.io
5	EOS tracker	eos实时查看器	少许进展	Cesar Rodriguez	西班牙	eostracker.io
6	Tokenika	可替换的EOS命令行界面 Alternative EOS command line interface	少许进展	Tomasz Michalski	华沙，波兰	tokenika.io
7	SFEOS	基于科幻小说的多人在线游戏	少许进展	未知	未知	billionairetoken.com
8	Billionare Token	去中心化的赌博游戏	已经部署在其他链	DAREN IOTT	未知	billionairetoken.com
9	Mithril Coin	基于游戏数据的去中心化游戏广告平台	白皮书概念验证	NJ Kim	韩国，新加坡	https://mithrilcoin.io/
10	GoodGame	去中心化的竞技游戏和支付平台	仅有概念	Michael Paulson	美国	未知
11	Vanilla RTB	去中心的和透明的程序化广告平台	仅有概念	Vladimir Venediktov	纽波特比奇，加利福尼亚，美国	未知
12	Credit Consent	区块链上的“生命锁”，利用网络影响和舆论	仅有概念	David Larry	弗洛里达，美国	creditconsent.io
13	Awoo! DAC	Awoo！是一个允许使用者完成任务赚取代币以换取其劳动价值的合作众筹平台	仅有概念	Troi Bryan	越南	未知
14	ONEPAY	onepay是销售系统的革命性创新，它允许全球的商业机构接受客户支付的各种主流加密货币，我们将会在零售pos 业带来巨大改变——无论是对于公司还是顾客。零售商的利益不是由传统支付系统。持有我们代币的持有人会享有我们收益的50%。	少许进展	未知	纽约	one-pay.io
15	Project H (TBA)	主要应用于体育，游戏，电子竞技，团队管理和社区管理，货物，服务，相亲，数据验证，价格自动评估	白皮书	Chris Dunn-Birch	加利福尼亚，苏黎世	未知
16	Tokenmarket	下一代电子商务和拍卖平台	在其他链开发，计划部署在eos上	Martin Rerak / Adam Levine	主要在南美	tokenly.com
17	EOS Commander	EOS智能合约开发者工具	进行中	NJ Kim	韩国，	play.google.com/store/apps/details?id=io.mithrilcoin.eoscommander&hl=en
18	butterfly	允许品牌和行业有影响力大佬相互合作，构思和执行有影响力的市场项目，建立公平透明的价值标尺，使用代币来交换价值，提供无偏见的归因和有效性度量	白皮书	Mel Kantor	洛杉矶，加利福尼亚	buttrfly.net
19	UnlimitedIP (UIP)	基于EOS打造的一款文娱版权智能交易平台,未来版权致力于为全球泛娱乐IP版权持有者提供包括极低成本的版权存证、版权认证、版权登记、版权交易、在线维权、内容孵化等于一体的一站式文娱版权智能服务。	进行中	迟静超	中国	http://www.unlimitedip.io
20	cybex					
21	MORE	MORE首先是个EOS钱包，是整个EOS生态系统的入口，可以在其中观测整个生态的发展。	开发中	Einstein	未知	more.top
22	Bystake	Bystake是一个提供高质量公共联动的平台。	Beta	未知	未知	bystake.com
23	Coinerworld	数字货币智能投资平台，通过大数据，机器学习，以及其他技术给使用者提供专门化的数字货币投资信息。	开发中	未知	未知	coiner.world

24	dStadia VR DACC	VR游戏平台，通过代币和投机形式展现“人类竞争”，通过专有区块链平台创造，发放收益。	开发中	未知	未知	dstadim.com
25	Kyndor	辅助学校，父母，当地商业，聊天平台		未知	未知	kyndor.com
26	Billionaire Token	去中心化游戏平台	开发中	未知	未知	billionairetoken.com
27	EOSfinex	全球首个高性能去中心化交易所，结合EOS.IO的速度和可扩展性以及Bitfinex的行业领先经验	开发中—Alpha	未知	未知	eosfinex.com
28	Koeos	人工智能框架	仅有概念	未知	未知	underconstruction.com
29	carmel	基于EOS的开源教育平台，帮助有经验的软件开发者通过代币化的平等挑战提升技能。	开发中	未知	未知	carmel.io
30	IIOTTEL	工业物联网两个组成，一个是分布式IIOT平台，另一个是无线网络LoRaWan	仅有概念	未知	未知	iiot.tel
31	SHOPXING	是一个去中心化的全球贸易市场，目前来说全球贸易存在收费高，地域限制和三方不信任的问题，shopxing提供商品服务的无现金点对点交易，并且为防止双重支付提供了加密码货币“XING”	仅有概念	未知	未知	shopxing.io
32	Bitjoy	使得程序开发人员在区块链上创建各种激励措施，还提供了一个端到端的内买系统。	仅有概念	未知	未知	bitjoy.io
34	invest digital	数字资产管理协议/工具箱	测试网络	Daniele Bernardi	未知	investdigital.info
35	augur	Augur是一个预测市场平台	Beta	Stephen Sprinkle	未知	augur.net
36	ONO	基于区块链的社交网络	近期上线	徐可	中国	未知
37	Plactal	基于区块链的移动游戏广告 韩国首个EOS Dapp	开发	未知	韩国	plactal.io
38	CryptoHouse	CryptoHouse是一个平台，希望在EOS平台上创建一个分散的应用程序，从而创建一个可以利用代币经济优势的社区： 1.拥有全球公共和私人空间的股份 2.金融房地产项目 3.在社区内创建一个实时投票和排名系统，仅为社区利益的最佳项目提供资金	Ver.0.1	未知	未知	3gatti.com/
39	Xenon	Xenon是来自EOS项目的替代区块链，它避免了公开发行衍生衍生品所面临的一些监管挑战，同时拥有比众筹EOS代币更广泛的分销渠道。	开发中	未知	未知	xenon.network
40	Kendra	我们正在构建一个独立的开源多平台应用程序，它将通过API“插入”传统服务提供商以及新的区块链“服务”通过API或直接连接到智能合约。Kendraio应用程序将为分割协商（本质上为投票），冲突解决，索赔请求（附有确认证据和第三方签名），	Ver.0.1	未知	未知	www.kendra.io
41	Uncloak	全球首个区块链3.0强大的网络威胁检测解决方案 拥有尖端人工智能的网络专业知识，致力于发现最新的网络威胁 Uncloak的数据库：提供了一个持续的实时网络威胁清单 企业可以保持领先并消除网络威胁，而不是依赖昂贵的第三方服务	Ver.1.0	未知	美国伦敦	uncloak.io
42	Weos	WEOS是一个社会生态系统，受到WeCash的激励，并受到社区的支配	概念	未知	美国	weosapp.com
43	antonym.life	ntonym.life是你生活的图书馆 - 一个使用数据驱动激励技术的平台 -	开发中	Louis Botha	毛里求斯	antonym.life
44	codum	Codum是一个基于EOS的基础架构和基于git的不同编程语言的代码市场	概念	UAB Teisingas kodas	立陶宛	codum.io
45	EOSbooks	基于eos的图书交换平台	概念	未知	北京，中国	无
46	Nebula	星云是一个分散的，精心策划的专业人士和就业机会清单	Ver.0.1			nebulaprotocol.com
47	SFEOS	SFEOS将是一款100%玩家运行的经济区块链游戏，以及一款完全由那些选择冒险，探索和开拓这部正在进行的史诗级游戏的人所拥有和运营的DAC	开发中	未知	未知	sfeos.io

3. 抛开EOS不谈，目前以区块链为基础的应用基本都与货币有关。

一种是生成货币，然后炒币。例如比特币，以太坊。

一种是生成货币，然后通过生成的虚拟货币（作为奖励）搭建去中心化平台。例如Steem（一个社交平台）。

总结：

EOS目前还是一个偏概念性的东西，虽然已经有编程实现，但是没人在上面开发出像模像样的成品。目前EOS的作用仅限于炒币。我个人认为EOS未来可能会出现像Steem之类的平台。

EOS的原理调研：

> PS: 本章内容大部分来自EOS白皮书。地址：
>
> https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md

底层实现：

EOS.IO借助WASM（WebAssembly，一种字节码技术，可将c/c++代码解释成二进制格式，也可以在json和二进制格式之间无缝转换）虚拟机，规范化了ABI格式，因此可以在不同的操作系统上进行通讯。

EOS.IO使用的数据库类似于mongodb，所有数据都被解析成json格式，然后通过WASM转换成二进制格式存储到数据库中。

和比特币，以太坊类似，EOS.IO基于石墨烯底层，实现高频交易。项目地址：https://github.com/cryptonomex/graphene

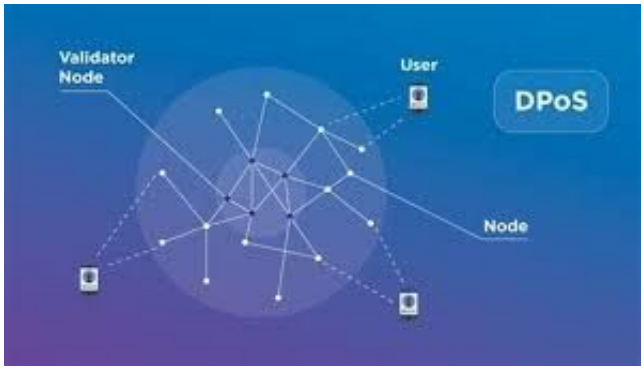
区块（block）：

PS：我个人认为的EOS.IO亮点之一

EOS.IO精确到0.5s生成一个块，同比特币一样，该块中记录了本时间段内交易的记录和上一个块的hash值。与比特币不同的是，新块中的hash值不做要求，新块的hash值不再需要满足类似0000000xxxxxxxxxx这种形式。

EOS.IO与比特币不同，EOS.IO通过BFT（拜占庭容错机制）和DPOS（股份授权证明机制）生成新区块，具体细节如下：

- **DPOS：股份授权证明机制**



> 简单来说，DPOS就是按股份管理公司制度，股东根据持股数投票选出董事会，董事会管理公司。

在EOS.IO中：

进行交易的货币被称为令牌（token），持有令牌的用户被称为股东。

每个股东根据自己持有的令牌数量，1:1获得选票，所有股东都可以自由投票给任意节点。最终，得票最多的21个节点成为区块生产者（见证人）。

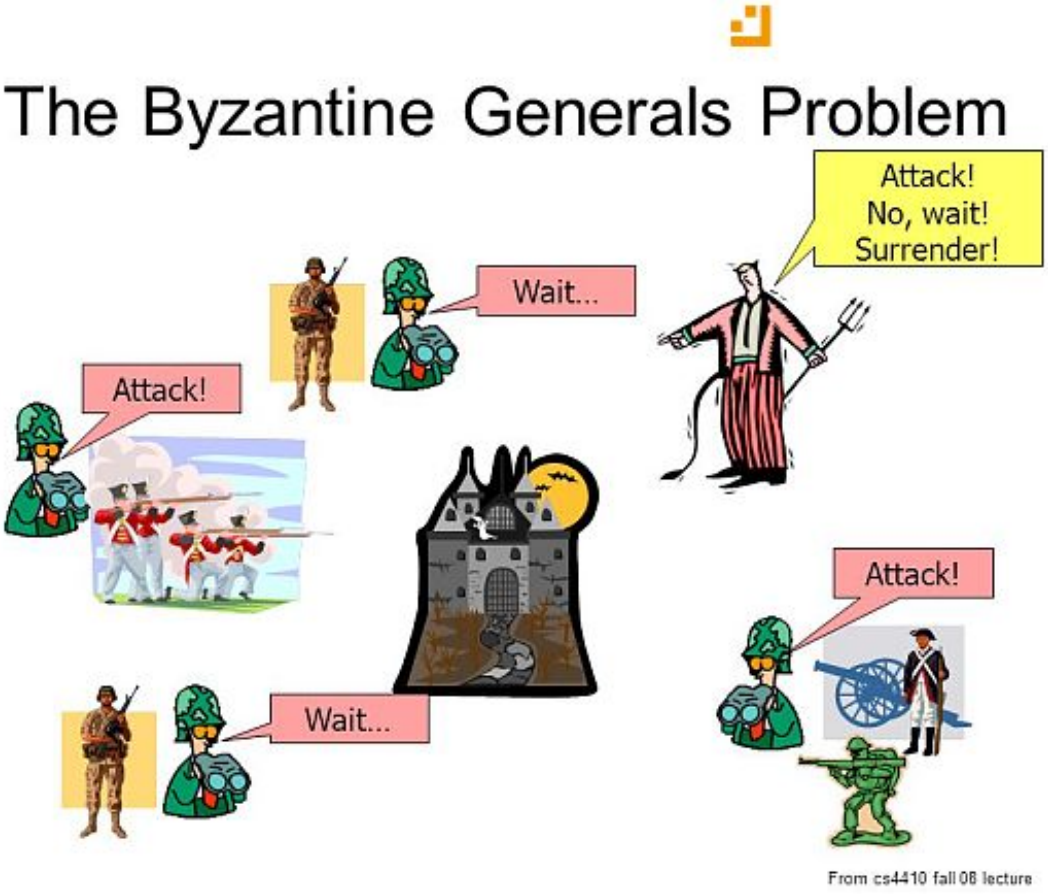
由这21个区块生产者，相互协作，按照一定的顺序，轮流进行记账。**出块间隔为3s一个的大区块**。此外，这21个区块生产者，不仅记账，还需要提供EOS全链所需要的计算和网络资源（包括CPU、内存、存储容量等等）。

如果某个节点出现故障，之后节点会跟上继续计块。

如果在过程中出现了分叉，DPOS仍然采用的是最长链原则，并且约定每个节点不能同时在两个链上出块（否则节点将被判定为违规，且失去资格），这使得当产生分叉之后，最多过一半见证人节点总数的高度之后（在EOS里是11个区块高度），就只会保留一条链了。

区块生产者进行一轮后（生产完21个大区块），股东将根据之前表现重新选择生产者，以此循环。

- **BFT：拜占庭容错机制**



> 简单来说，BFT就是少数服从多数原则，所有节点共同执行多数人听到的命令。

为了提高块确认的速度，EOS.IO又将大区块分解成**出块间隔0.5s一个的小区块**。小区块间不改变生产者。

区块生产者每生成一个小块之后，就会马上广播给其他区块生产者。其他区块生产者收到广播后，会立即对该区块进行确认，当有2/3的区块生产者（15个）对该区块进行确认后，该区块就会进入**不可逆状态**。加入到区块链中。

确定性并行执行：

- **区块分割**

为了最大限度的进行并行执行，减少平均通讯延迟，EOS.IO将：

区块（block）分成多个周期（cycles）。

一个周期分成多个并行执行的碎片（shards）。

一个碎片包含一个事务列表（transactions）。

一个事务列表包含一个或多个动作（actions）。

每发起一个交易，就会生成一个新的事务列表，该事务列表一般都会在一个block之内完成，这样，一个用户向另一个帐户发送操作并接收响应所需的时间就不会超过0.5s。

```
Block

  Region

    Cycles (sequential)

      Shards (parallel)

        Transactions (sequential)

          Actions (sequential)

            Receiver and Notified Accounts (parallel)
```

- **上下文无关操作**

上下文无关行动涉及仅取决于交易数据的计算，而不取决于区块链状态。例如，签名验证是一种只需要事务数据和签名以确定签署事务的公钥的计算。这是区块链必须执行的最昂贵的单个计算之一，但由于此计算是上下文无关的，因此可以并行执行。

- **只读操作**

某些帐户可能能够以通过/未通过的方式处理动作，而不必修改其内部状态。如果是这种情况，那么这些处理程序可以并行执行。

令牌（token）：

令牌是EOS.IO的货币。与比特币一样，生产者生产新区块后，会获得令牌作为奖励。

EOS.IO为了防止资源滥用，通过令牌分配区块链上的资源。通过链接上EOS.IO的区块链，应用程序可以使用三大类资源：

1. 带宽和磁盘
2. CPU
3. RAM

同时，用户的本地的可用资源也会被系统统计计算。

用户根据自身持有的令牌数量，使用区块链上的资源，自身拥有的令牌数量越多，可使用的区块链上的资源也越多。例如：如果一个账户持有根据该区块链可分配总代币的1%，则该账户有可能利用1%的RAM容量。

账号（account）：

EOS.IO通过公钥，私钥，ID来创建账号。EOS.IO为每个帐户提供了自己的专用数据库，只能由其自己的操作处理程序访问。其中，公钥，私钥储存在自己专用数据库中，避免泄漏。

- **操作和处理程序**

每个帐户都可以将结构化的操作发送给其他帐户，并可以定义脚本以在收到操作时处理操作。这个结构化的操作由合约（contract）定义。

为了保证并行，用户在数据库中访问账户信息的时候没有冲突。

- **权限管理**

PS：我个人认为的EOS.IO亮点之二

在EOS.IO软件，帐户可以定义命名的权限级别，每个级别都可以从更高级别的命名权限中获取。一定或更高级别的权限可以执行对应级别的动作。

- **权限映射**

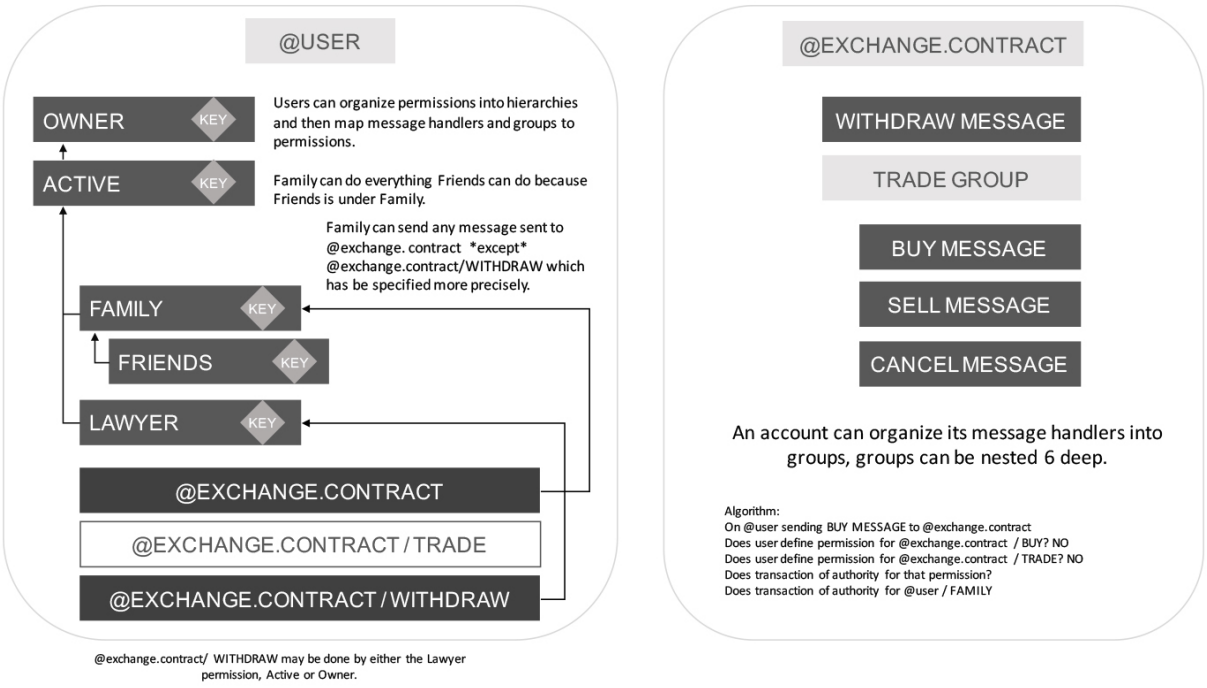
EOS.IO软件允许每个帐户定义任何其他帐户的合同或动作与其自己的命名权限级别之间的映射。例如：可以将账户alice与账户bob的交易行为映射到账户alice的"Friend"组，这样下次交易就不需要进行权限认证。

- **评估权限（没看太懂）**

假设alice和bob进行交易，交易类型叫做Action，评估先后顺序如下：

1. 查看bob的Action组是否存在权限映射
2. 遍历bob的Action组的父节点寻找是否有alice
3. 如果未找到，遍历alice父节点，重新执行1，2

如下图：



合约（contract）：

EOS.IO的合约包含一组操作和类型定义。其中，操作定义指定并实施合约的动作（即Action），类型定义指定了操作所需的内容和结构。

账户和合约之间可以通过动作进行通讯，中间过程类似于CS架构中的client端和server端。

PS：EOS.IO已经开放了自定义智能合约的api，用户可以生成自己的合约。

自治：

PS：我个人认为的EOS.IO亮点之三

自治分为三个步骤：

1. 用户对某个集体的主关问题达成共识。例如：生产者生产新区块获得的令牌过多，应该少一点。
2. 执行集体决定
3. 通过宪法改变规则

- **冻结账户和更改合约代码**

当出现异常用户或者异常程序时，生产者可以通过投票决定是否冻结账户或者替换合约代码。
2/3（15个）生产者同意则操作执行。

- **宪法**

宪法的内容规定了用户之间的义务，不能由法典完全执行，并通过确立管辖权和法律选择以及其他相互接受的规则来促进解决争端。在网络上广播的每一笔交易都必须包含宪章的散列作为签名的一部分，从而明确约定签署人与合同。

宪法是用户建立合约和交易时绑定签署的部分条约，所有用户都必须要遵守宪法。

宪法修改过程如下：

1. 区块生产者提议修改宪法，并获得2/3（15个）生产者批准。
2. 2/3（15个）生产者连续30天支持新宪法。
3. 所有用户都必须表明接受新宪法，作为未来交易的前提条件。
4. 区块生产者通过修改源代码来反映宪法的变化，并使用新宪法的散列将其提交给区块链。
5. 2/3（15个）生产者连续30天支持新代码。
6. 对代码的更改在7天后生效，在批准源代码后给予所有非生产节点1周的时间进行升级。
7. 所有不升级到新代码的节点都会自动关闭

总结：

EOS.IO构建的系统类似于互联网，添加合约类似于在互联网上添加服务端，添加账户类似于在互联网上添加客户端节点。

个人认为EOS.IO的优点和特点：

1. **支持高频并发交易**
2. **完善的账户权限系统**
3. **拥有交易不可抵赖性**
4. **可以分配整个区块链的资源**
5. **高度自治**
6. **去中心化**

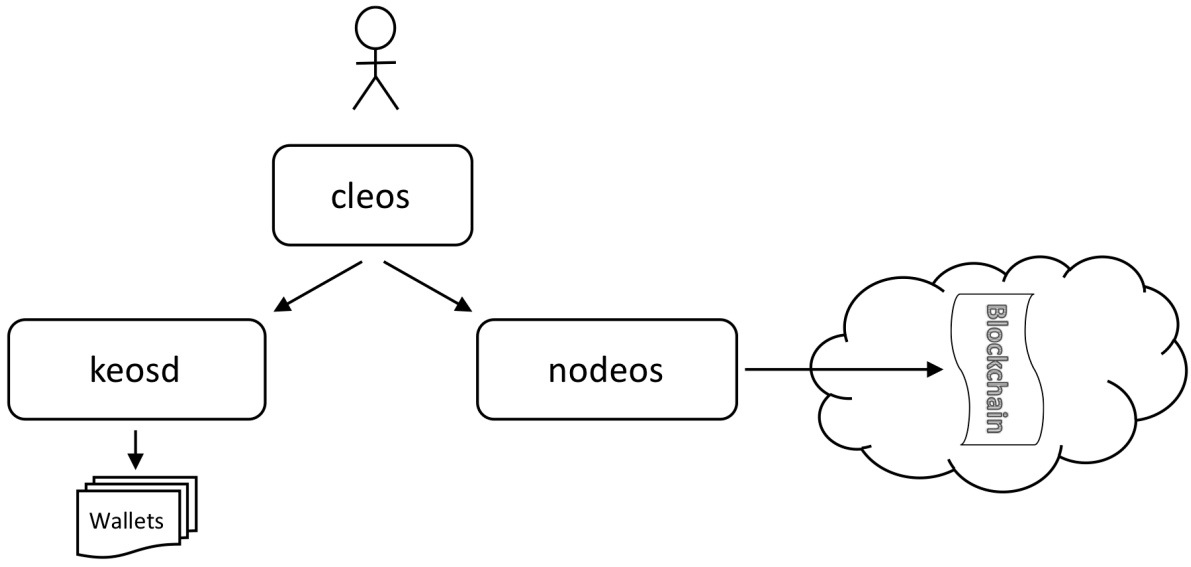
虽然EOS.IO可以在多个计算机之间搭建统一平台，但是我还未找到对应的安装教程和资料，可能还未实现。

EOS的程序实现：

> PS：本章内容大部分来自于EOSwiki，地址：
>
> https://github.com/EOSIO/eos/wiki

PS：官网上的EOS.IO项目目前还是私有链测试版。

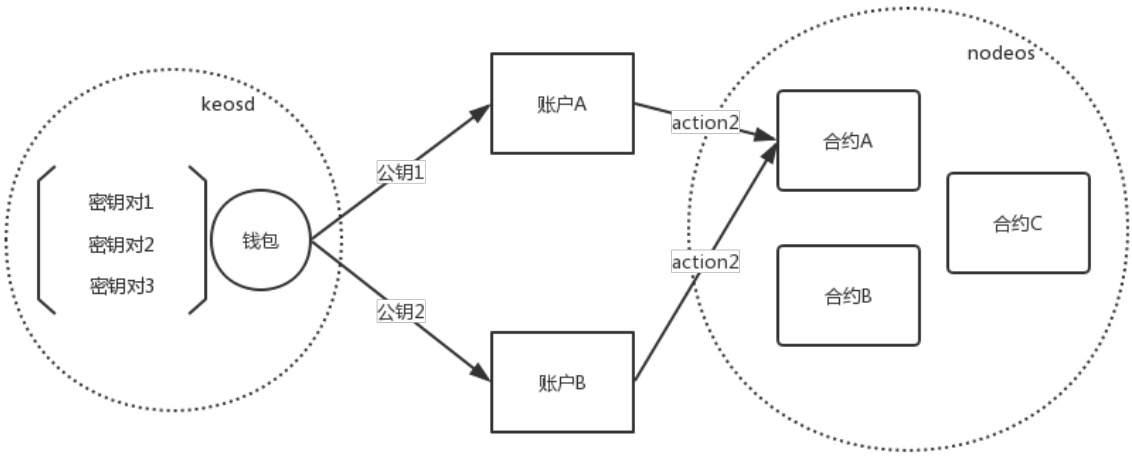
EOS.IO程序分成三个部分，如图所示：



- nodeos：服务端。建立在本机的私有区块链，接口为Restful模式，运行后每0.5s生成一个块。通过nodeos可以进行账户和合约操作。
- cleos：客户端。用户输入指令访问keosd和nodeos进程，执行命令。
- keosd：管理wallet的守护进程，wallet包含私钥和公钥，储存在本地保证安全。

钱包，密钥，账户，合约的关系如下：

- 钱包：储存在本地数据库，一个钱包可包含多个密钥对，只有钱包打开后，用该钱包中密钥创建的账户才能进行交易。
- 密钥：可生成任意个，生成后可放入钱包内，用于账户创建的时候设置Owner和Active权限。
- 账户：由钱包中的公钥创建，一个钱包可以创建多个账户，账户可以通过合约与其他用户进行交易。账户是进行操作的基本单位。
- 合约：类似于服务端，运行在区块链上。账户可以通过合约指定的Action接口与合约进行交互。账户和合约类似于CS架构。



EOS的程序部署：

> PS：本章内容大部分来自于EOSwiki，地址：
>
> https://github.com/EOSIO/eos/wiki/Local-Environment

配置环境：macOS High Sierra 10.13.4

安装编译源程序：

- 从github上下载指定版本，本次下载dawn-v3.0.0版本，如果需要其他下载版本可以去github查看。

```
$ git clone https://github.com/EOSIO/eos.git --recursive
```

必须要添加—recersive，整个时间较长。

- 进入目录文件夹，运行脚本自动安装依赖环境。

```
$ cd eos
$ ./eosio_build.sh
```

EOS依赖环境如下，如果brew运行不了，可以自行安装：

1. make
2. cmake
3. boost
4. openssl
5. secp256k1-zkp
6. wasm （**PS：一定要4.0.0版本**）

- 进入build文件夹，进行编译。

```
$ mkdir build && cd build
$ cmake -DBINARYEN_BIN=~/binaryen/bin -DWASM_ROOT=~/wasm-compiler/llvm -
DOPENSSL_ROOT_DIR=/usr/local/opt/openssl -
DOPENSSL_LIBRARIES=/usr/local/opt/openssl/lib ..
$ make -j4
```

编译过程较长，可能需要近一个小时。

- （可选）如果想选择安装程序包，直接make install即可。

```
$ make install
```

运行nodeos节点：

- 修改config.ini文件

```
$ vim ~/Library/Application\ Support/eosio/nodeos/config/config.ini
```

添加或修改一下信息：

```
# Load the testnet genesis state, which creates some initial block
producers with the default key
genesis-json =  "~/Library/Application
Support/eosio/nodeos/config/genesis.json"
# Enable production on a stale chain, since a single-node test chain is
pretty much always stale
enable-stale-production = true
# Enable block production with the testnet producers
producer-name = eosio
# Load the block producer plugin, so you can produce blocks
plugin = eosio::producer_plugin
# Wallet plugin
plugin = eosio::wallet_api_plugin
# As well as API and HTTP plugins
plugin = eosio::chain_api_plugin
plugin = eosio::http_plugin
```

（PS：eos的nodeos信息存放在~/Library/Application Support/eosio/nodeos/文件夹下）

- 修改完毕配置信息后，即可运行（前提将mongodb数据库开启）

```
$ cd /path/eos/build/programs/nodeos
$ ./nodeos
```

- 如输出以下信息，则运行成功。

```
2056511ms thread-0   producer_plugin.cpp:239       block_production_loo ]
eosio generated block bcdc4795... #208537 @ 2018-05-15T06:34:16.500 with 0
trxs, lib: 208536
2057011ms thread-0   producer_plugin.cpp:239       block_production_loo ]
eosio generated block 7d512d05... #208538 @ 2018-05-15T06:34:17.000 with 0
trxs, lib: 208537
2057510ms thread-0   producer_plugin.cpp:239       block_production_loo ]
eosio generated block c795360e... #208539 @ 2018-05-15T06:34:17.500 with 0
trxs, lib: 208538
2058011ms thread-0   producer_plugin.cpp:239       block_production_loo ]
eosio generated block be4026cd... #208540 @ 2018-05-15T06:34:18.000 with 0
trxs, lib: 208539
2058513ms thread-0   producer_plugin.cpp:239       block_production_loo ]
eosio generated block bbfa3c34... #208541 @ 2018-05-15T06:34:18.500 with 0
trxs, lib: 208540
```

PS：整个安装过程需要时间较长。

EOS的合约构建和常见API使用：

> PS：本章tic.tac.toe合约见github。
>
> https://github.com/EOSIO/eos/wiki/Tutorial-Tic-Tac-Toe
>
> PS：合约源码在eos/contract/tic_tac_toe也能找到。

- 创建新钱包mywallet：

```
$ cleos wallet create -n mywallet
Creating wallet: mywallet
Save password to use in the future to unlock this wallet.
Without password imported keys will not be retrievable.
"PW5HzsvBkDc3BWHwPydDfa9YEzeneHCCcTvLUUYUaxMSxXnMJWbtJ"
```

会生成对应钱包的密钥，请记住。

- 输入刚刚的密码，解锁钱包mywallet：

```
$ cleos wallet unlock -n mywallet
password: Unlocked: mywallet
```

- 生成一对密钥：

```
$ cleos create key
Private key: 5J425KHYu4XN6sxWxNRA83hCVEiWJ22HHLiQjq7sXpVYNS3Sdjb
Public key: EOS6GxfQNPKs13U3aXubDmEXocqjnWaNhksSojVw2Q9UWknvpqBi2
```

- 将密钥加入钱包mywallet：

```
$ cleos wallet import 5J425KHYu4XN6sxWxNRA83hCVEiWJ22HHLiQjq7sXpVYNS3Sdjb -
n mywallet
imported private key for:
EOS6GxfQNPKs13U3aXubDmEXocqjnWaNhksSojVw2Q9UWknvpqBi2
```

- 根据刚刚的密钥创建两个账户，alice和bob：

```
$ cleos create account eosio alice
EOS6GxfQNPKs13U3aXubDmEXocqjnWaNhksSojVw2Q9UWknvpqBi2
EOS6GxfQNPKs13U3aXubDmEXocqjnWaNhksSojVw2Q9UWknvpqBi2
executed transaction:
00af7518e9cfcdd3e6781a63faf5df1c4a9d6a8858510fdfed310e286e105d2b  352 bytes
 102400 cycles
#         eosio <= eosio::newaccount
{"creator":"eosio","name":"alice","owner":{"threshold":1,"keys":
[{"key":"EOS6GxfQNPKs13U3aXubDmEXocq...
$ cleos create account eosio bob
EOS6GxfQNPKs13U3aXubDmEXocqjnWaNhksSojVw2Q9UWknvpqBi2
EOS6GxfQNPKs13U3aXubDmEXocqjnWaNhksSojVw2Q9UWknvpqBi2
executed transaction:
918122fe176eb2cd3785f6886500c77cfafbb1da0e546678175c203b128fdd71  352 bytes
 102400 cycles
#         eosio <= eosio::newaccount
{"creator":"eosio","name":"bob","owner":{"threshold":1,"keys":
[{"key":"EOS6GxfQNPKs13U3aXubDmEXocqjn...
```

eosio是系统默认的账户。我们使用刚刚创建的公钥创建账户。

其中"eosio <= eosio::newaccount"即代表执行了一个事务，该事物由一个action组成。

- 创建合约tic.tac.toc，由于eos给定的合约缺少.wast文件，我们先编译，再创建合约。

```
$ cd /path/eos/contracts/
$ eosiocpp -o tic_tac_toe/tic_tac_toe.wast tic_tac_toe/tic_tac_toe.cpp
$ cleos set contract tic.tac.toe tic_tac_toe/
Reading WAST/WASM from tic_tac_toe/tic_tac_toe.wast...
Assembling WASM...
Publishing contract...
executed transaction:
98499896fa16bad6d7adb55de2cf2e1296deaccde6b466b7ed3141b87878933e  4848
bytes  2200576 cycles
#         eosio <= eosio::setcode
{"account":"tic.tac.toe","vmtype":0,"vmversion":0,"code":"0061736d010000000
16a1260000060027e7e006001...
#         eosio <= eosio::setabi
{"account":"tic.tac.toe","abi":{"types":[],"structs":
[{"name":"game","base":"","fields":[{"name":"ch...
```

- alice账户创建新棋局（执行create Action）：

```
$ cleos push action tic.tac.toe create '{"challenger":"bob",
"host":"alice"}' -p alice@active
executed transaction:
4e10c6a8f257681ff64ca0937f83833a68e5e976d4edbdb507882467fdef5ab6  240 bytes
 104448 cycles
#   tic.tac.toe <= tic.tac.toe::create
{"challenger":"bob","host":"alice"}
```

其中-p意思是给定此Action的权限。

- alice和bob各走一步（执行move Action）：

```
$ cleos push action tic.tac.toe move '{"challenger":"bob", "host":"alice",
"by":"alice", "mvt":{"row":0, "column":0} }' -p alice@active
executed transaction:
12711be167e84b1c3eb191c2c170c2c68dd0feed039767a1abb5cc45656808c9  256 bytes
 107520 cycles
#   tic.tac.toe <= tic.tac.toe::move
{"challenger":"bob","host":"alice","by":"alice","mvt":{"row":0,"column":0}}
$ cleos push action tic.tac.toe move '{"challenger":"bob", "host":"alice",
"by":"bob", "mvt":{"row":1, "column":1} }' -p bob@active
executed transaction:
0b6c28e9d3ecbea4077f78a20db1225ea798afadb15c4baca9954338c25fc94a  256 bytes
 107520 cycles
#   tic.tac.toe <= tic.tac.toe::move
{"challenger":"bob","host":"alice","by":"bob","mvt":{"row":1,"column":1}}
```

- 棋局情况：

```
$ cleos get table tic.tac.toe alice games
{
  "rows": [{
      "challenger": "bob",
      "host": "alice",
      "turn": "alice",
      "winner": "none",
      "board": [
        1,
        0,
        0,
        0,
        2,
        0,
        0,
        0,
        0
      ]
```

```
    }
  ],
  "more": false
}
```

- alice关闭棋局（执行move Action）：

```
$ cleos push action tic.tac.toe close '{"challenger":"bob",
"host":"alice"}' -p alice@active
executed transaction:
74b592bb33f6bc9e7c23800dc2fd607969ba74f7ba1de43ef0794a4607842207  240 bytes
 105472 cycles
#   tic.tac.toe <= tic.tac.toe::close
{"challenger":"bob","host":"alice"}
$ cleos get table tic.tac.toe alice games
{
  "rows": [],
  "more": false
}
```

目前进展：

1. 已经了解了EOS的基本原理构造和运作方式
2. 成功的在本地部署了了EOS.IO程序
3. 可以较为熟练的使用cleos中的api
4. 已经开发出了教程中的tic.tac.toe合约，并且成功运行

存在的问题：

1. 目标需求不明确：目前我有两个进一步研究的方向。
    - 研究EOS具体底层实现方式。例如：研究石墨烯项目，了解高频并发如何做到。这一点可能对公司有些贡献。
    - 进行EOS合约开发。例如：开发xiaozong之前说的工资系统，实现每个月发工资。
2. EOS很多具体细节，例如如何进行节点间通讯，怎么实现DPOS，我到现在还是没有搞懂，可能需要进一步研究。