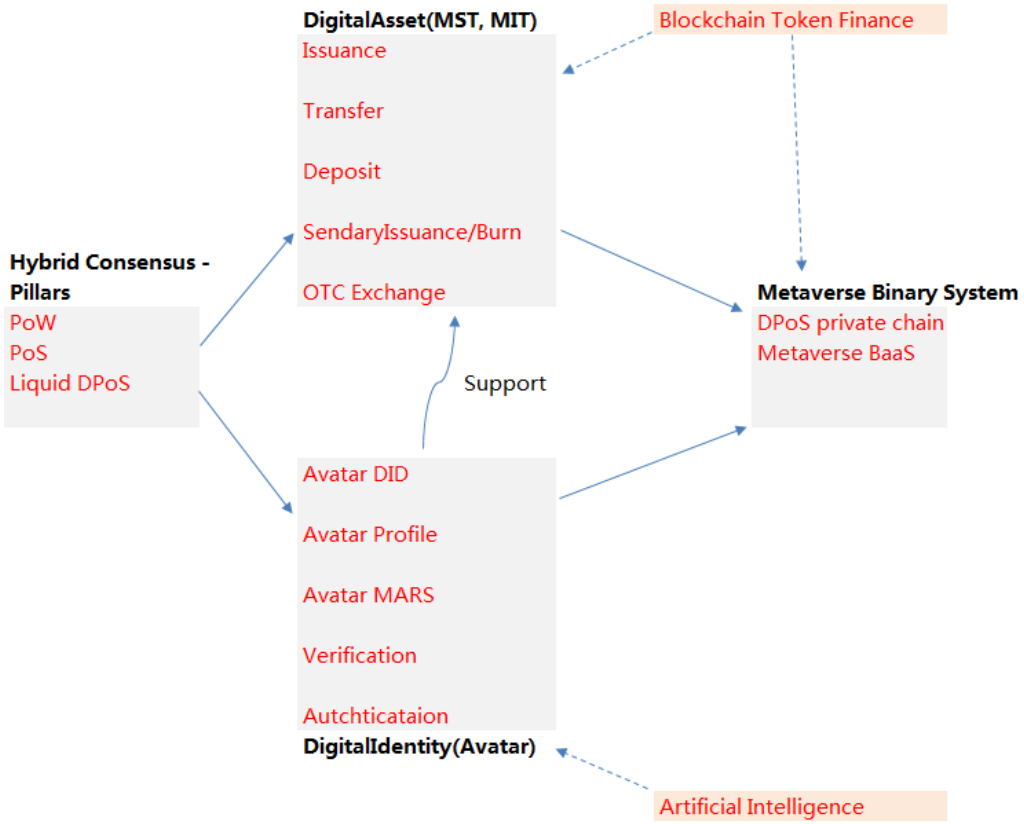
元界创世之柱技术白皮书

关键词(Key words)

混合共识，双链架构，MST可挖矿，解决51%攻击，解决nothing at stake，函数式智能合约，去中心化信用系统

摘要(Abstract)

元界创世之柱MPC（Metaverse Pillars Of Creation）是继SuperNova之后的下一个大版本，将支持混合共识算法（PoW+PoS+DPoS）三算法同时生效的共识机制，这种共识机制可以避免PoW的51%攻击，也可以避免PoS的nothing at stake攻击，在支持混合共识的同时，这些共识算还可以支持MST可挖矿。同时我们将开放全新的基于元界数字身份的MARS系统（Metaverse Avatars Reputation System），MARS系统是基于元界的开放式的去中心化的社会化的信用系统。在初版白皮书（Draft）中，我们描述了一种智能资产，智能资产的实现将基于可被验证的智能合约模板，我们也将在MPC中实现这种可编程的智能资产。设计和实现上述功能的同时，我们将为元界提供Binary-Port-Chain，用于提供标准数字身份服务和基于元界函数式智能合约的容器服务，帮助企业快速对接元界主链，属于双链系统，我们称之为双子星系统（Metaverse Binary System）。



我们将MPC将分为两个子阶段，第一子阶段我们主要升级共识算法、提供MST挖矿以及MARS系统，第二子阶段我们主要实现可编程的智能资产和双子星系统（Metaverse Binary System）。

前言

元界核心团队将元界的发展大致分为四个阶段，
第一阶段是First Release（2017.Feb~2018.Jul），发行了ETP，提供了基础数字资产功能；
第二阶段是SuperNova（2018.Jul~2019.Mar），升级了数字资产的功能，新增了数字身份；
第三阶段是PillarsOfCreation（2019.Mar~2020），提升TPS，升级数字身份的，新增智能资产；
第五阶段Galaxy（2020~），微通胀的宏观经济模型，面向区块链数据，为人工智能提供数字身份和智能资产标准服务协议。

MPC第一阶段：
Feb. 14, 2019　发布0.9.0全节点安装包
Mar. 1, 2019　激活创世之柱协议，目标块高1924000
MPC第二阶段：
Jul.2019

混合共识Pillars

概述

纯的PoW或PoS的TPS并不能满足海量应用的需要，就目前业界的区块链扩容方案来看，主要分为两个类型：

1. 修改共识算法本身来提升TPS
2. 修改交易的结构来提升TPS

方法1比方法2更底层，是优先考虑的方案，在MPC阶段，我们将考虑采用混合共识机制来提升TPS。考虑MVS的主网已经采用了PoW共识算法，所以我们在MPC阶段会完全兼容现有的PoW共识算法的（即目前ETP挖矿完全不受影响）。

混合共识机制

常见的混合共识机制可分为两类：PoW+PoS以及 PoW+BFT。
PoW+BFT（或其他改进）：

1. PoW产生区块，然后通过BFT进行最终化。
2. PoW 确定 Leader，Leader 负责写入key-block和micro-block，主要解决selfish-mining： Bitcoin-NG 为代表，包括 Credo、Hcash；

PoW + PoS 又分为三种情况：

1. PoW和PoS 并行竞争出块：UBTC；
2. PoW 打包出块，PoS 最终化：以 Decred 为代表，包括 Casper、Espers ；
3. PoW和PoS联合出块，固定出块的比例和顺序。

分析PoS

在讨论混合共识之前，我们先从PoS开始分析，PoS有多个变种，包含**Pure PoS1.0/ PoS2.0/ PoS3.0，Dynamic PoS，Liquid PoS，Lease PoS，Forging PoS**，而他们都基于以下逻辑：

On PoS protocol, blocks are separated into two distinct types: PoW blocks and PoS blocks. The PoS in the new type of blocks is a special transaction called coinstake (named after PoW special transaction coinbase). In the coinstake transaction, block owner pays himself thereby consuming his coin(or coin age), while gaining the privilege of generating a block for the network and minting for PoS. The first input of coinstake is called kernel and is required to meet a specific hash target protocol, thus making the generation of PoS blocks a stochastic process similar to PoW blocks. However, a significant difference is that the hashing operation is done over a limited search space (more specifically one hash per unspent wallet–output per second) instead of an unlimited search space as in PoW. Thus no significant consumption of energy is involved (King & Nadal, 2012).

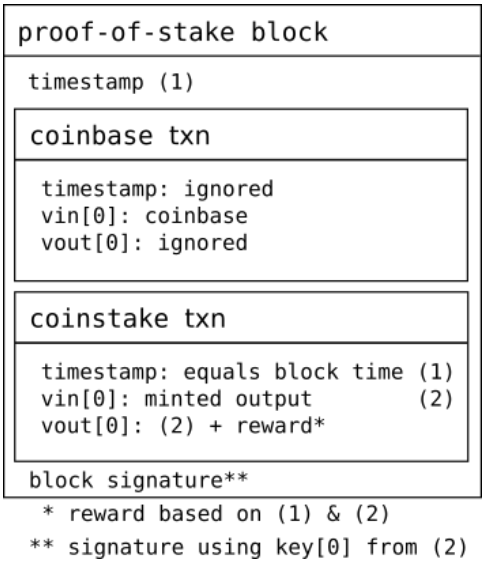
也就是说，PoS本身是包含了PoW的，只是在Pure PoS算法中，强化了PoS部分，弱化了PoW部分。我们一共分析了点点币PPCoin、新星币NovaCoin、雅币YaCoin、比特存储、黑币，我们发现早期都采用POW+POS混合的模式。

在PoS中出块必须满足以下条件:

hash(stake_modifier, current_time, UTXO) < coin(UTXO) * difficulty

1. 用户在每一秒时间（current_time），遍历自己所有的UTXO，代入上述公式中，看是否能满足不等式条件；如果满足，就把相应的UTXO记录在block中，并发布block（见4）
2. stake_modifier是对前一个block中部分字段hash后的值，加入这一项是为了防止用户提前预知自己何时有权挖矿
3. difficulty会根据近期的block产出时间动态调整，保证block产出时间间隔稳定
4. 由于每秒只需要完成和自己UTXO数量相等的hash计算，所以需要的算力较低
5. 从不等式可以看出，持有的UTXO越多、UTXO中token数额越大（coin(UTXO)）、UTXO持有时间越长（age(UTXO)，或称之为币龄），不等式越容易成立，越容易进行挖矿
6. 生成block的奖励设置为了coin(UTXO) * age(UTXO)，即UTXO数额越大持有时间越长，奖励越高
7. 为了将符合条件的UTXO记录进block，并且兼容原本的PoW模式，Peercoin设计了coinstake的逻辑：保留原本第一个transaction为coinbase，但要求输入数量必须等于1，且输入的prev.out字段必须置空值，输出数量必须大于等于1；令第二个transaction为coinstake，要求输入数量大于等于1，且第一个输入为满足条件的UTXO，输出数量大于等于2，且第一个输出必须置空值，第二个输出为block奖励。

一个PoS块的结构如下：



在某些版本的PoS设计中，采用的是如下公式：

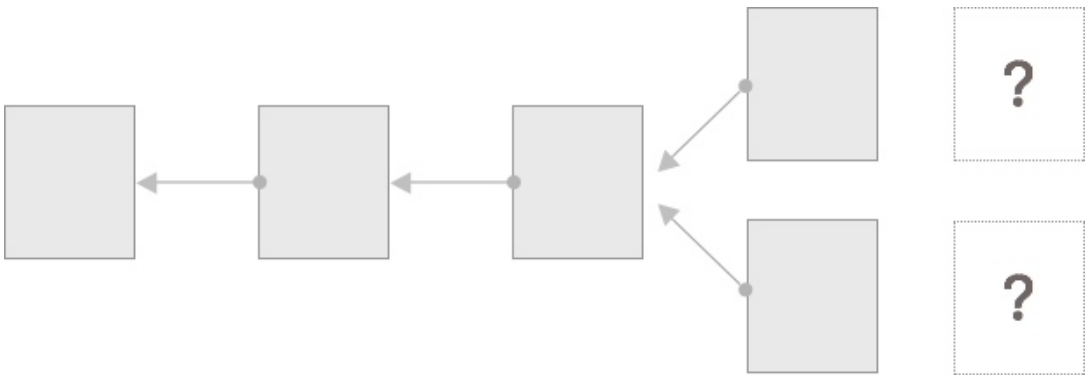
hash(stake_modifier, current_time, UTXO) < coin(UTXO) * age(UTXO) * difficulty

里面多了时间作为参数，由于引入了时间因素，所以容易出现了攒币龄攻击（ Coin Age Accumulation Attack ）。即关闭节点，直到age(UTXO)足够大的时候再启动节点挖矿，从而节省电力，这样引起了在线节点数太少系统脆弱的问题。当然，设置一个age上限似乎也是可以的，但这样也失去了age作为调节变量的意义。综上，元界的混合共识Pillars将不考虑币龄这种方案。

PoS仍还面临着一些问题：

无利益关系攻击（ Nothing at Stake Attack ）

在 PoW 机制中，当账本出现分叉时，对 PoW 这种算力敏感的算法，矿工必须选择一个方向进行挖矿以保持跟随难度最大的链。而在 PoS 这种算力不敏感的时候，PoS 矿工往往会多个方向都挖，以争取实现利益最大化。随着时间推移，链趋向于发散而不是收敛，所以当多数矿工都在多个分叉链上一起挖矿的时候，就会很容易出现双重支付攻击，那么这条链的账本几乎不可用。



长程攻击（ Long-Range Attack ）

PoS 中，产生每个 Block 的速度相对 PoW 快了很多。因此，少数不怀好意的节点会想着把整个区块链共识账本全部重写。这在 PoW 中是经典的 51% 问题，即：当某一个节点控制了 51% 及以上算力，就有能力篡改 逆向账本至多6个块，这种追溯随着逆向的块越多陡然增大，所以即使坐拥51%以上的算力，逆向篡改6个以上区块也是非常困难的。而在 PoS 中缺

乏物理算力的约束，那么就逆向篡改账本是可以达到任意块高，从这个角度来说PoS没有PoW安全。

贿赂攻击（Bribe Attack）

贿赂攻击流程如下：

1. 攻击者购买某个商品或服务
2. 商户开始等待网络确认这笔交易
3. 此时，攻击者开始在网络中首次宣称，对目前**相对最长的不包含这次交易的主链进行奖励**。
4. 当主链足够长时，**攻击者开始放出更大的奖励，奖励那些在包含此次交易的链条中挖矿的矿工。**
5. 六次确认达成后，放弃奖励。
6. 货物到手，同时放弃攻击者选中的链条。

因此，只要此次贿赂攻击的成本小于货物或者服务费用，此次攻击就是成功的。相比之下，PoW 机制中贿赂攻击就需要贿赂大多数矿工，因此成本极高，难以实现。

虽然纯的PoS有着诸多的问题，但相对PoW有更好的可塑性，所以我们将考虑在元界的Pillars混合共识中去解决这些问题。

元界的混合共识Pillars

虽然PoW算法简单粗暴，但是PoW依然面临着可扩展性不足、51%攻击、自私挖矿（Selfish-mining）等问题。通过上文分析PoS，我们发现可以在保留并完全兼容PoW的基础上，选择PoW+PoS竞争出块的模式，最终还可以通过元界的MARS系统来激活DPoS。
我们将混合共识Pillars划分为两个阶段。

Pillars@Phase1

→ 激活点 – 1924000
→ 平均出块时间控制在 25.02 s，PoW 占比y = 90%，PoS 占比 z = 10%

调整PoW难度调整

- 哈希算法完全兼容
- 根据理论和实际出块时间计算得当前PoW出块时间约33.5秒/块，**微调为28秒/块**
- 连续W个区块必须包含一个PoS块，**W = 30；**
- 难度调整为下述算法，使PoW出块更平稳：当然算力大幅增长时，增长速度和以前一致，当算力撤出时，算力下降比之前快很多（对矿工有利）。

```
bigint const interval = (bigint)(_bi.timestamp-_parent.timestamp);
bigint const adjustvalue= max<bigint>(2 - interval /10 ,-99);
target = _parent.bits + _parent.bits/2048*adjustvalue;
```

经过计算，由于出块，PoW矿工相比升级之前，收益提升**7%**左右。

设计PoS

- 根据z=10%，调整为**252秒/块**，初始块奖励0.3*0.95^3，每50万衰减

- 当PoW难度低于一定值时，PoS获得50%收益（待定）；
- PoS挖矿的钱包必须具有至少一个inbound 连接

```
bigint const interval = (bigint)(_bi.timestamp-_parent.timestamp);
bigint const adjustvalue= max<bigint>(18 - interval /10 ,-99);
target = _parent.bits + _parent.bits/2048*adjustvalue;
```

Pillars@Phase2

→ 可能的激活点 – 2541142

→ 平均出块时间控制在 16.35 s，PoW 占比y = 15/32，PoS 占比 z = 1/16，DPoS 占比 v = 15/32

调整PoW难度调整

- 哈希算法完全兼容
- 根据理论和实际出块时间计算得当前PoW出块时间约33.5秒/块，**微调为34.9秒/块**
- 难度调整为下述算法，使PoW出块更平稳，根据前向区块的类型，快速调整难度

```
bigint const interval = (bigint)(_bi.timestamp-_parent.timestamp);
bigint const adjustvalue= max<bigint>(2 - interval /10 ,-99);
if (prevs_header.is_pow_version() || prevs_header.is_pos_version()) {
  adjustvalue *= 2;
  target = _parent.bits + _parent.bits/2048*adjustvalue;
} else {
  target = _parent.bits + _parent.bits/2048*adjustvalue;
}
```

设计PoS（MPC@Phase1）

- 根据z=2/32，调整为**268秒/块；**
- 当PoW难度大于xxx时或低于xxx时，PoS获得50%收益；
- PoS挖矿的钱包必须具有至少一个inbound 连接；
- 当PoS挖矿消耗任意UTXO以后（1000ETP UTXO），该UTXO的恢复期块高g(z)满足下述函数：

$$g(z)=\frac{1}{1+e^{-z}}$$

其中z是PoS挖矿难度；

设计DPoS（MPC@Phase2）

目标：

1. 通过模式设计，让DPOS共识更稳定出块。
2. MARS分数是出块的核心评价，考虑节点的质量。从节点是否拥有证书，stake是否足够，以及是否积极出块三个方面进行评价；证书体现基金会的管理效率，规范性和对节点质量的保证；stake体现节点对网络的影响力；是否积极出块体现了节点在网络运行过程中是否尽到了应尽的义务。
3. 在每个节点更替周期的初期，对发展下属节点的一级节点，也有节点管理激励，考虑节点的数量。

证书属性：
证书有如下状态：

- 激活状态
- 失活状态

激活状态的充分条件：发展至少23个二级节点
失活状态的必要条件：经历相对200万块则失活
失活状态下可转移给其他人
激活状态下不可转移
不可主动注销失活

设计内容：

- 根据v=15/32，调整为**34.9秒/块**（释放曲线待定，上凸可积函数，备选sigmoid）
- 初始块生成23个见证人证书；
- 拥有一级证书则可以设立23~46个二级证书；
- 一级至少设23个以后才能开启挖矿（二级证书接收人必须足够10万ETP且在注册列表前100的用户），一级证书持有者再获得15.8万ETP（挖矿释放，不挖没有）；
- 普通投资投资者可选择证书持有人，将手中的stake租借给证书持有人，从中获得收益；
- DPOS见证人被选举的可能性完全由MARS值决定：

$$MARS\ 值 = F(NodeCert, NodeStake, NodeBlockMiss)$$

NodeCert：即上文解释的证书
NodeStake：即持有ETP的Stake
NodeBlockMiss：Miss 出块的统计数值

1.节点数量、权利和义务概述

表1

节点类型	数量	权利	义务
一级节点	23	1. 持有一级证书（两个月有效） 2. 生成最多46张二级证书 3. 激活一级证书后提升MARS值，200万块高内有效 4. 根据出块情况获得块奖励	1. 获得一级证书之后发展至少23个二级节点参与共识，激活证书有效性 2. 积极参与出块，维持MARS值，避免被冻结、吊销证书
二级节点	最多1058个	1. 持有二级证书，提升MARS值，200万块高内有效 2. 根据出块情况获得块奖励	1. 积极参与出块，维持MARS值

2.MARS设计
MARS受到证书、stake和出块miss率三个因素的影响：

$$MARS(Node) = Cert(NodeCert) + Stake(NodeStake) + BlockMissRate(NodeBlockMiss)$$

（1）持有有效状态的一级证书或二级证书，cert分数均有30分

$$Cert(NodeCert) = \begin{cases} 30, & NodeCert = true, \\ 0, & NodeCert = false. \end{cases}$$

（2）stake数量分数：例，10万ETP对应30.9分，公式如下：

$$Stake(NodeStake) = \log(NodeStake) \times 6.18$$

（3）BlockMissRate（出块错失率）分数：定义–节点Node第EpochNum次被选为epoch的候选出块节点，23名代表节点轮流出230个块，则每个节点理论出块数出10个块；如果轮到Node出块，而Node因为网络原因等错过了出块，则BlockCount<10。当前epoch结束的时候，会计算Node节点第EpochNum次出块的BlockMissRate分数，公式如下：

$$BlockMissRate(NodeBlockMiss) = 30 + NodeBlockMiss \times \begin{cases} 0, & NodeBlockMiss < -0.2 \\ 15, & 0 > NodeBlockMiss \geq -0.2 \\ 5, & NodeBlockMiss \geq 0 \end{cases}$$

其中，NodeBlockMiss是Node节点从第一次成为候选人至今的出块错失率，递归计算公式如下，且定义NodeBlockMiss(0,BlockCount)=0：

$$NodeBlockMiss = NodeBlockMiss(EpochNum, BlockCount) = \frac{NodeBlockMiss(EpochNum-1, BlockCount') \times (EpochNum-1) + BlockCount}{EpochNum},$$
$$EpochNum \in N+$$

3.MARS相关激励，以及发展节点激励（**草案**）

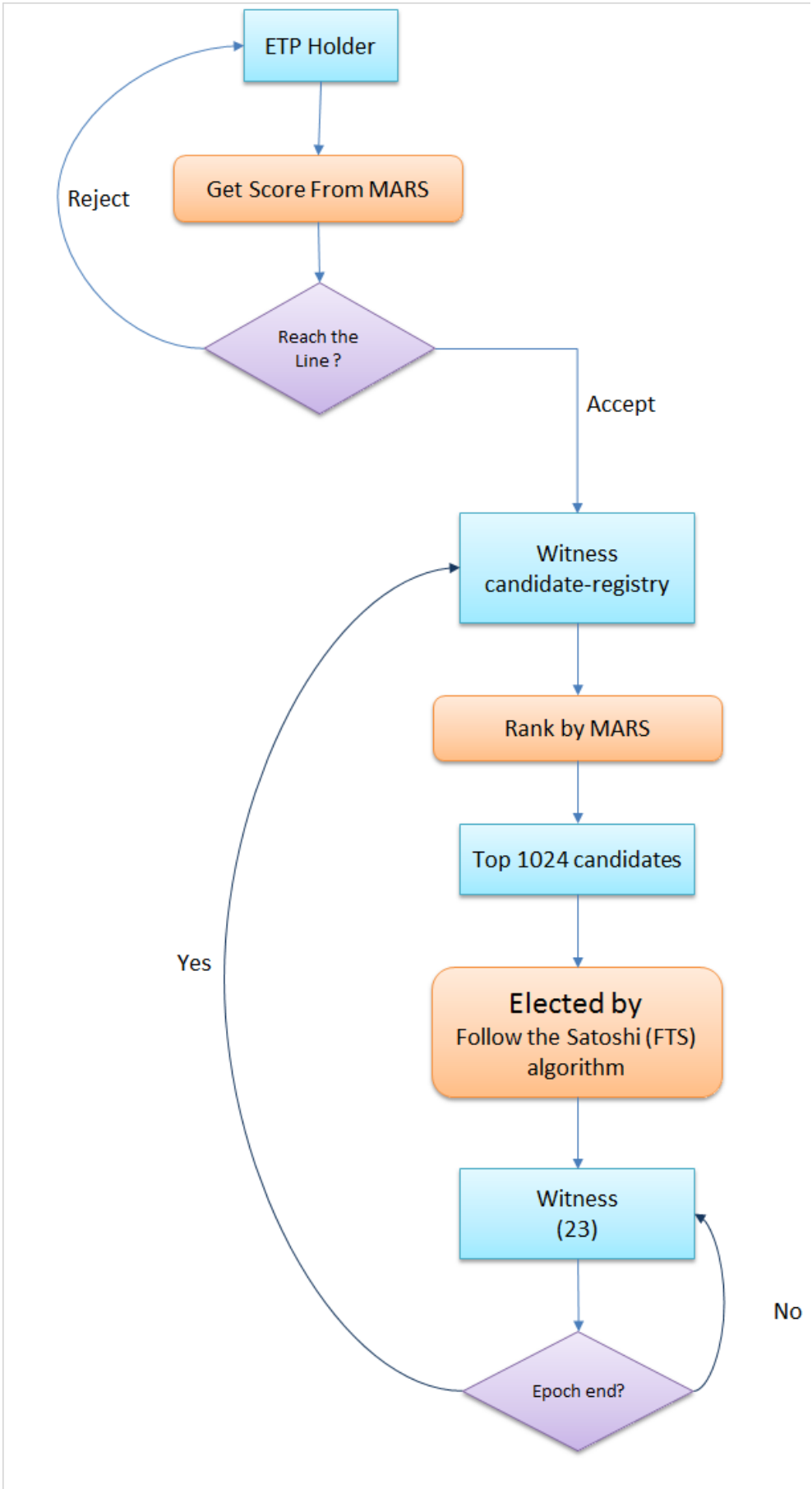
表2

节点类型	动作	激励
一级节点	获得基金会核发的一级证书	获得**18000**ETP，分12个月等量释放 增加MARS分，更易出块
	两个月内发展23个二级节点，激活证书	获得**158000**ETP，分12个月等量释放 增加MARS分，更易出块
	出块	0.5　ETP：块奖励 下属二级节点数*0.01　ETP：节点管理奖励 x ETP：当前块手续费 MARS值稳定，更易出块
	一级证书被吊销	降低MARS分，更难出块 剩余锁仓ETP没收回归基金会
	一级证书到期失效	降低MARS分，更难出块
	发展二级节点失败	一级证书被吊销，降低MARS分，更难出块

	MARS分长期过低	冻结一级证书，及其下属二级证书 停止发放锁仓ETP，直到节点与基金会取得联系 第三次冻结后吊销证书，及其下属二级证书
二级节点	获得一级节点和基金会核发的二级证书	获得**6000**锁仓ETP，分6个月等量释放 增加MARS分，更易出块
	出块	0.5 ETP：块奖励 x ETP：当前块手续费 MARS值稳定，更易出块
	二级证书被吊销	降低MARS分，更难出块 剩余锁仓ETP没收回归基金会
	二级证书到期失效	降低MARS分，更难出块

4. 算法基本流程

算法基本流程：

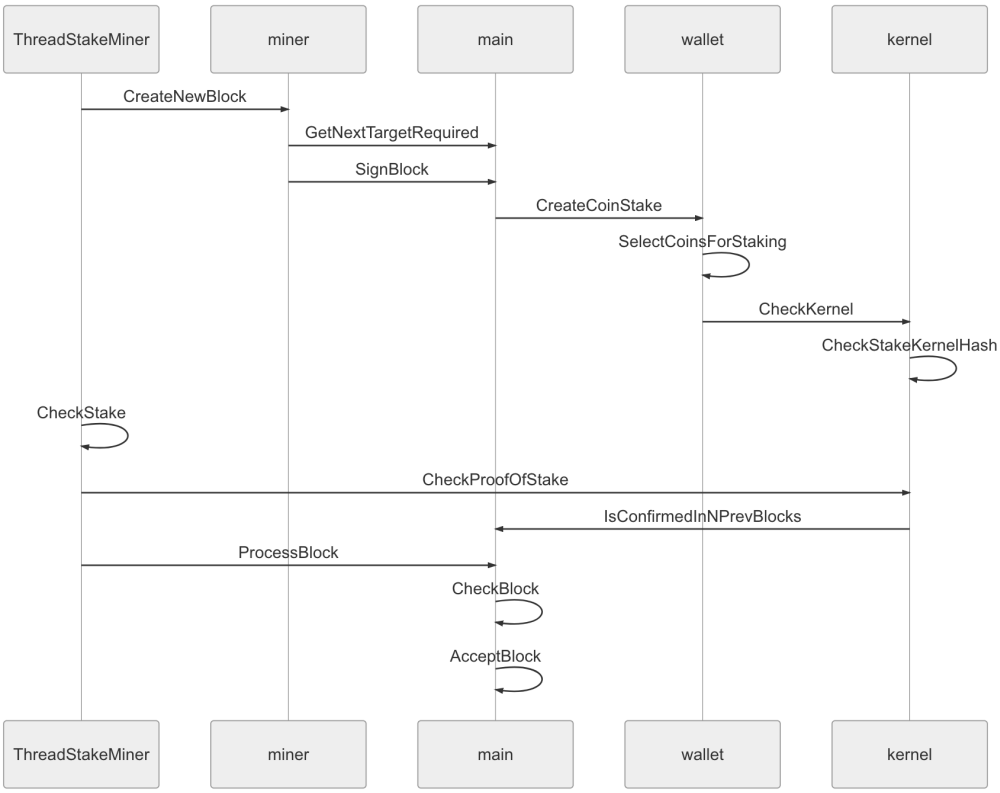


Witness–MARS 结合了中本聪随机数算法，MARS分作为权重影响被选概率；
1024不足则按实际人头选；
超过1024则按MARS排名选前1024名，MARS的影响因子只有stake时则退化为POS；

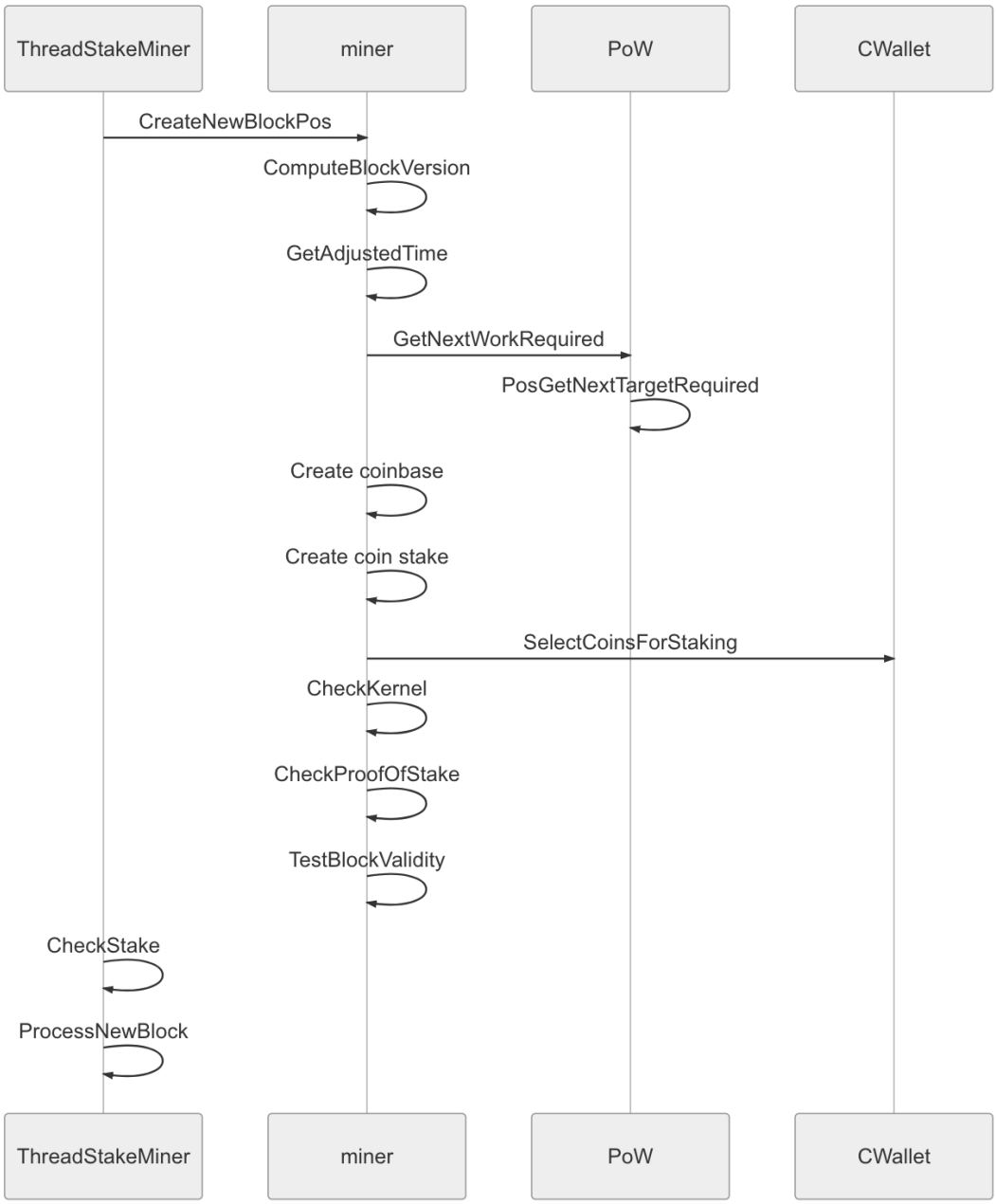
实现PoW和PoS竞争混合出块

首先考虑同时激活PoW和PoS共识，因为两者的块结构相似，并且基本出块逻辑一致，两者的区别只有PoS比PoW多出coinstake结构，所以我们只要在共识的验证部分，同时认可PoS区块和PoW的区块即可。

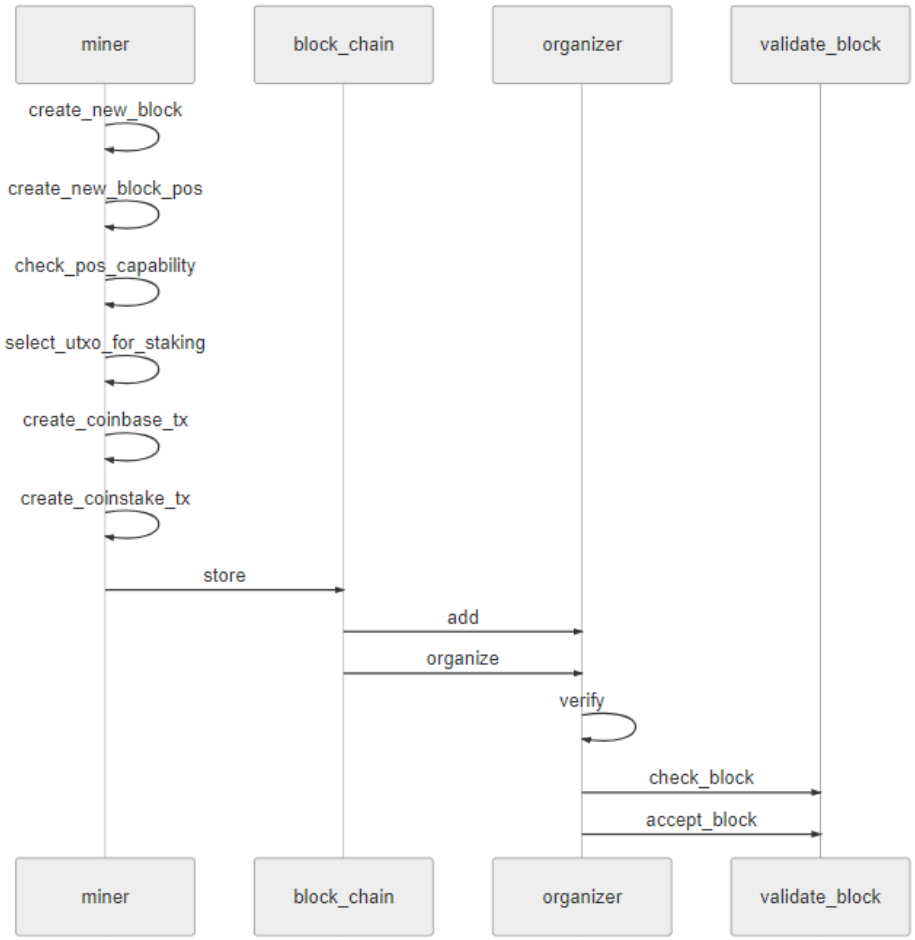
下图是**Blackcoin** 对应的PoS机制在代码调用过程中的流程：



我们再来参考UBTC的PoS流程：



上述两个币种为我们提供了PoW+PoS混合共识的设计思路。



我们可在元界的代码中作出了上述修改，如流程图所示，在 miner::create_new_block 中创建 pos 块，该 block 的 version 设置为 block_version_pos，同时在块的交易中添加了 coinstake 交易。如前所述， coinstake 交易是具有特定格式的交易：

- inputs 的第一项，为代表挖矿的 stake UTXO；
- outputs 的第一项为空；
- outputs 的第二项为 inputs 的第一项对应的 UTXO 输出；

同时，为了防止伪造身份出块以及块被篡改，PoS 块头中还包含用见证人的私钥对整个块的签名 blocksig。

要激活 PoS 需同时满足如下两个条件：

1. 锁仓一定数量的 ETP；
2. 持有一定数量可用于挖矿的 ETP；一个 UTXO 需满足一定块高的成熟度才可以用于挖矿；

为了归集小额 UTXO以及拆分大额 UTXO 便于挖矿，每一个 coinstake 交易自带小额归集以及大额拆分功能。

PoS 也必须满足MARS对应的分数才能挖矿。

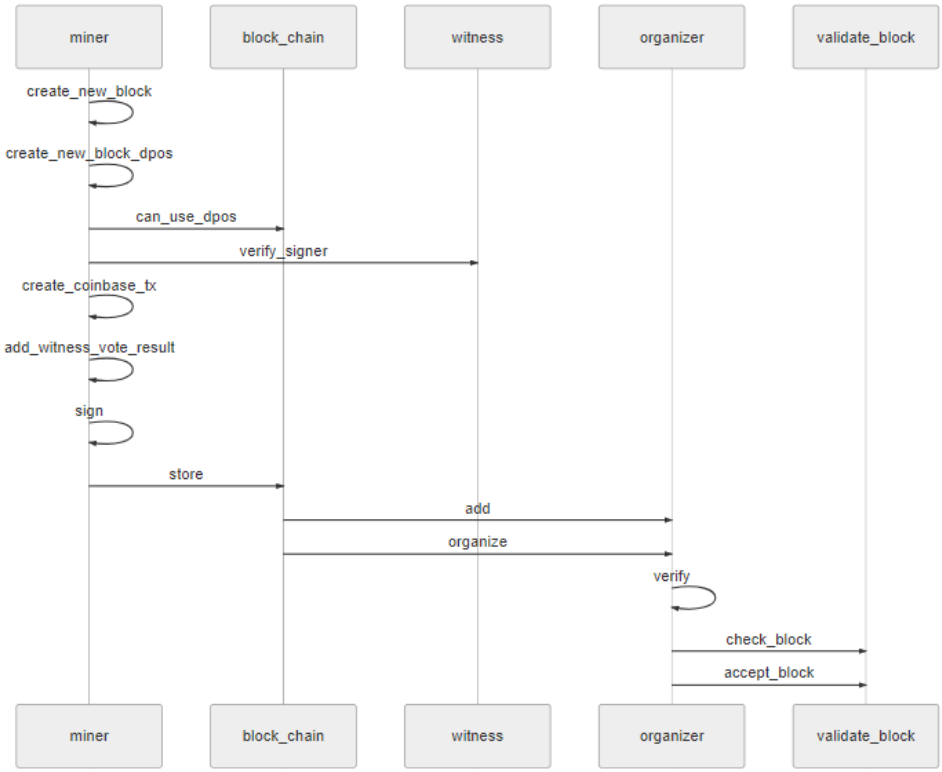
实现DPoS跟随出块

元界的混合共识还支持 DPoS 跟随出块，即一个 PoW 或 PoS 块之后可以跟随出一个 DPoS 块。首先生成一个DPOS共识表，该表作默克尔哈希登记到区块header上，记录哪些证书有效，哪些无效，我们根据DPOS登记簿的MARS分进行随机选举；

DPoS 实现流程如下：

1. DPoS 矿工登记为候选见证人；
2. 候选见证人锁仓一定数量的 ETP；
3. 每一个 epoch 内的第一个块根据候选见证人锁仓权益比例用追随中本聪算法随机选举出一组见证人；
4. 在一个 epoch 内，由选举出的见证人轮流出块；
5. 为了应对见证人不在线的情况，每个 epoch 内出块比例远低于平均水平的见证人将禁止参与下一轮选举。

DPoS 出块流程如下：



如流程图所示，在 miner::create_new_block 中创建 dpos 块，该 block 的 version 设置为 block_version_dpos，同时在块的 header 中添加了出块见证人的 public key。

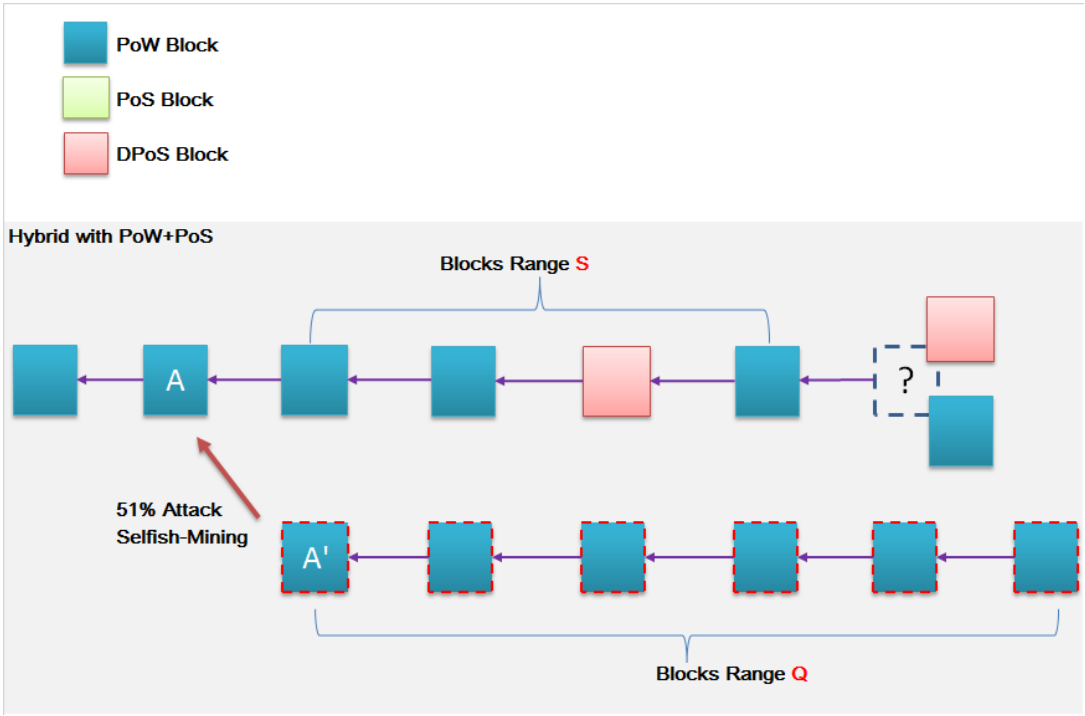
同时，为了防止伪造身份出块以及块被篡改，PoS 块头中还包含用见证人的私钥对整个块的签名 blocksig。

要激活 DPoS 需同时满足如下两个条件：

1. 发起一笔登记为候选见证人的交易；
2. 锁仓一定数量的 ETP；

混合共识Pillars的优势

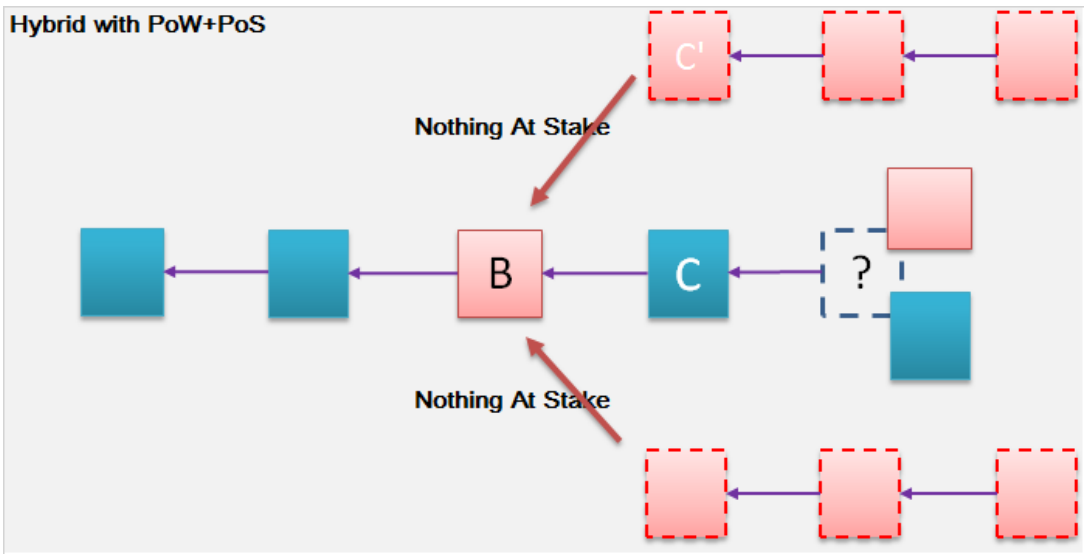
首先我们考虑PoW的51%攻击，如下图所示：



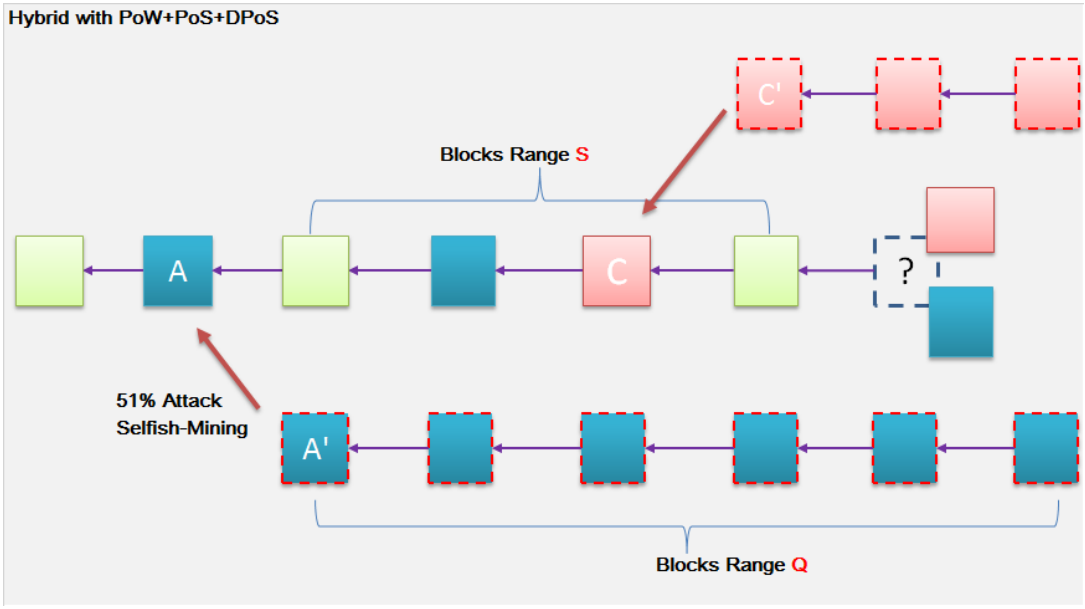
当攻击者发起51%攻击或者Selfish-mining时，在攻击点A处积攒了长度为Q的分叉链，当Q大于S时，攻击者释放Q链时，必须满足：

当Q大于等于W=30时，攻击者的Q链不被接受；

当Q小于等于W=30时，攻击者的Q链仍然处在未达到安全确认数的状态；



当PoS矿工发起Nothing At Stake时，总是会被后续的PoW区块C determine，PoS矿工无法维护较长的分叉链。



当我们进入三混合共识的阶段时，DPoS的区块是跟随PoW区块出现的，当攻击者选择A’或者C’进行攻击时，总是会被DPoS区块阻隔住；当DPoS试图发起攻击时，也会被PoW和PoS阻隔住；

女巫攻击–Sybil Attack

目前还没有出现能够抵抗Sybil Attack的共识算法，而通过Avartar 的 MARS值，通过扩展评价维度，可以提高Sybil Attack的攻击阈值。

可能的混淆攻击

1. 恶意篡改PoW区块版本，使之进入PoS的验证流程
   由于 PoS 块结构与 PoW 块结构不同，因此伪造成 PoS 的 PoW 块是无法通过共识验证的。一个合法的 PoS 块必须满足如下所有条件：
   a. 块的 version 为 block_version_pos；
   b. 块的第一个交易必须为 coinbase 交易
   c. 块的第二个交易必须为 coinstake 交易；
   d. coinstake 交易 inputs 的第一项，为代表挖矿的 stake UTXO；
   e. coinstake 交易 outputs 的第一项为空；
   f. coinstake 交易 outputs 的第二项为 inputs 的第一项对应的 UTXO 输出；
   g. 块头中包含见证人私钥对块的签名 blocksig；

只有当上述条件都满足时，一个PoS区块才被认可，所以混淆PoW区块为PoS是行不通的。

2. 恶意篡改PoS区块版本，使之进入PoW的验证流程
   由于 PoS 块结构与 PoW 块结构不同，因此伪造成 PoW 的 PoS 块是无法通过共识验证的。一个合法的 PoW 块必须满足如下所有条件：
   a. 块的 version 为 block_version_pow；
   b. 块的第一个交易必须为 coinbase 交易；
   c. 块的交易中一定没有 coinstake 交易；

只有当上述条件都满足时，一个PoS区块才被认可，所以混淆PoW区块为PoS是行不通的。

3. 时间戳攻击
   共识中对块的时间戳有严格的验证，一个合格的块的时间戳必须满足如下条件：

a. 为防止过去时间攻击，块的时间戳必须不早于前一个块的时间戳；
b. 为防止未来时间攻击，块的时间戳必须不迟于验证的时间加一个狭窄的时间窗口；

元界的时间窗口调整为38秒，PoW和PoS的时间戳验证是不同的，也无法通过混淆攻击相互干扰对方的时间戳。平均出块时间是25秒，也就是攻击者最多生产2个攻击区块，当攻击者妄图生成更多的攻击区块时，难度会快速上升导致出块变慢。

P2P网络的承载能力

P2P网络消化所有区块的极限是10秒左右，而元界的P2P网络在未经大幅度优化，并且没有在uncle block的情况下，暂不适合优化到15秒左右。

对闪电网络的支持（@MPC Phase1）

MPC第一子阶段将支持闪电网络；
The Lightning Network is dependent upon the underlying technology of the blockchain. By using real Bitcoin-like transactions and using its native smart-contract scripting language, it is possible to create a secure network of participants which are able to transact at high volume and high speed.
我们都知道元界所使用的交易结构与比特币类似，所以我们在也元界的网络上独立实现了Lightning Network。

原ETP的锁仓奖励调整（@MPC Phase1）

在MIP-2中我们分析了锁仓奖励是一个不合理的设置，在7~14年后可能会导致ETP总量突破1亿枚。
考虑到锁仓奖励与PoS奖励设置的相似性，所以我们决定将所有的锁仓奖励转换为PoS和DPoS的奖励，而之前过锁仓利息产生的ETP将永久保留，由于出块时间加速，之前的锁仓代币会提前释放。

数字资产协议升级

MST可挖矿（@MPC Phase 1）

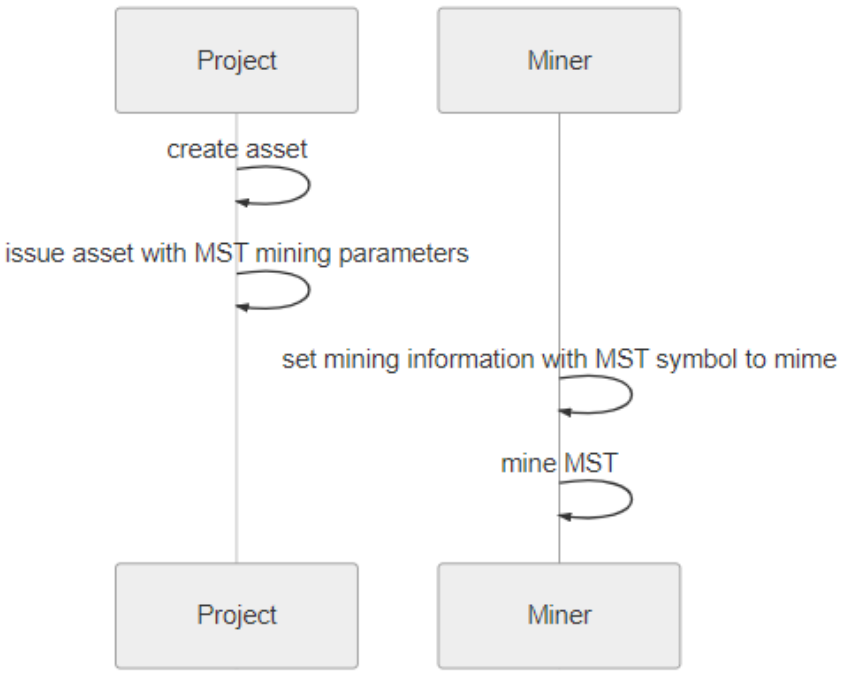
为了让 MST可共享元界共识算法的资产分布能力，MPC阶段一支持针对MST资产的PoW和PoS挖矿，在MPC第二阶段将支持PoW，PoS，DPoS三种共识挖矿。

元界的所有资产结构均是UTXO-based，我们针对Coinbase作了outputs扩展，具体如下：

```
{
    "hash" : "bf4ca43a2c23c8d5b06",
    "inputs" :
    [
        {
            "previous_output" :
            {
                "hash" : "0000000000000000000",
                "index" : 4294967295
            },
```

```
            "script" : "[ b71f0f ]",
            "sequence" : 0
        }
    ],
    "outputs" :
    [
        {
            "address" : "Miner's Address'",
            "attachment" : // nomarl coinbase
            {
                "type" : "etp"
            },
            "index" : 0,
            "value" : 94338400
        },
        {
            "address" : "Miner's Address",
            "attachment" : // new MST output for coinbase
            {
                "quantity" : 300000000,
                "symbol" : "TEST001.MINING",
                "type" : "asset-transfer"
            },
            "index" : 1,
            "value" : 0
        }
    ],
}
```

MST 挖矿的流程如下图所示：



具体步骤如下：

1. 创建MST资产时指定可挖矿；

2. 主网登记MST资产，同时指定 MST 挖矿奖励的参数；
3. 矿工指定挖何种 MST；
4. 矿工像往常一样开始挖矿，不久就会在在挖到的块的coinbase奖励中包含 MST 奖励。

资产定价置换Swap（@MPC Phase 2）

Refers to MIP-15

数字身份协议升级

开放的MARS标准（@MPC Phase1）

Refers to MIP-16
在MPC Phase2实现引用MARS分数值的DPoS算法

兼容OIDC的统一登录入口（@MPC Phase 2）

当开启元界钱包时，元界Avatar继承OIDC服务，可提供去中心化的OIDC身份服务，这将使得Avatar的作用范围拓展到传统互联网应用。

元界标准身份服务（@Galaxy）

精准空投服务
个人成就证书服务
开放式的公链数据挖掘（AI友好，Public blockchain data mining）

元界双链系统（Binary-System@Galaxy）

区块链的应用落地一个比较大的挑战无法平衡TPS和去中心化，当提升TPS时则无法提供良好的去中心化能力，所以我们可以考虑双链架构。将目前元界的主链作为基础链，可以提供良好的去中心化能力，而第二层链提供高TPS的传输能力，可以和主链上的DPoS保持同步关系。第二层链面对的问题是如何适配现有的系统。

主链上的微观经济体

To be written...

双链上的双向适配（Binary-Port）

（以下内容Copy自白皮书Aggregation部分）
架构模式有很多种，我们仅讨论单点应用、分层架构模式、事件驱动的架构模式和微服务架构模式。我们将讨论MBaaS应当处在这些架构模式的什么位置。

MBaaS与钱包的关系

首先MBaaS是一类服务的集合，在系统中的表现形式是一类服务进程，它们通常是由钱包程序运行而产生的。

目前我们有两种模式可以可以操作：

1. **钱包程序分离模式**：从钱包程序分离出功能，做成多进程模式，每个进程提供了轻量级的MBaaS；
2. **钱包程序统一模式**：钱包程序提供有所MBaaS，但是可以形成主从关系，做成内部分布式网络，而不是连接到公网。统一模式对钱包的优化和稳定性提出了更高的要求。

分离模式

元界至少提供以下基本模块分离：

- P2P网络
- 交易验证和解析
- 私钥管理
- 区块持久化存储

轻钱包是分离模式的第一个案例，从全节点钱包。

统一模式

钱包程序至少提供内部高速同步的功能，内部节点从最终一致性转变为强一致性，要求当区块链分叉时，支持内部节点可以达成强一致。

分离模式和统一模式并不是绝对的，在实际场景中可能存在混合用。我们接下来根据架构模式进行探讨。

单体应用（Monolith Appicatlions）

单点应用分为客户端单点应用和服务端单点应用。

服务端单点应用例如wordpress，如果我们想让wordpress支持MST，最快速的方法是在wordpress的后端同时搭建元界钱包，然后修改后端代码去调用MST相关的API，最终界面显示MST的token即可。这种情况适合统一模式，快速部署。

单点应用比较常见地是使用微内核架构（Microkernel Architecture）模式。例如在Eclipse IDE中植入Metaverse Avatar，这就要求元界轻量级钱包作为插件植入到Eclipse中。这种情况适合分离模式，例如轻钱包。

分层架构模式（Layered Architecture）

分层架构既适合分离模式，也适合统一模式，取决于这个分层架构的规模大小。分离模式适合大规模的分层架构，例如一个SOA架构中，统一模式显然胜任不了。

考虑分离模式，分离模式下的MBaaS适合放在分层模式的业务层，成为一个个普通组件，它只要求钱包的API与业务层的其他模块尽可能兼容。如果存在持久层，可能需要区块链结构化存储，否则可以直接使用钱包本身替代区块存储的功能。

考虑统一模式，统一模式下的MBaaS适合小规模的应用，MBaaS的搭建可以参考服务端单点应用。

事件驱动的架构模式（Event-driven Architecture）

事件驱动的架构模式适合分离模式。

事件驱动的架构模式关注的是对事件的分发和处理，如果我们分析区块链的逻辑，我们可以发现区块链都是基于交易的，交易本身就是一个事件，所以在事件驱动的架构模式中，分离模式是最适合的。

在Mediator模式中，交易类型与交易数据需要被解析再分发，所以Mediator必须具有解析交易的能力。如果是账户状态模型，还需要具备读取账户状态的能力；在Broker模式中，每个

Processor具有解析和判断交易的能力即可，不涉及Broker的变更。

上面的流程作为事件输入的，而需要交易输出的时候，我们可以可以把钱包看成一个processor，只处理和区块链相关的业务，但这里可能会遇到这个processor演变成中央processor的问题，因为在任意一个核心业务流的最终目的都是支付，钱包processor会成为验证、签名、广播交易的集合体，会遇到明显的性能瓶颈。

所以分离模式下的网络模块、交易验证模块可以设置为水平扩展的。分离模式要求元界提供比较完备的SDK来支持事件分发和处理。

微服务架构模式（Micro-services Architecture）

微服务架构既适合统一模式，也适用于分离模式。

考虑统一模式，让钱包成为微服务组件，只要求钱包功能足够内聚。例如钱包可在组件A中扮演支付的角色，在组件B中扮演交易验证的角色。这就要求钱包的行为尽可能地贴合微服务架构中微服务，并且提供足够丰富的查询和验证API。

考虑分离模式，分离模式似乎与微服务的架构思路非常契合，那么提供标准化的元界微服务组件就成了我们首先要考虑的，这似乎不是很难。

智能合约（@Galaxy）

Standard Template Library of Smart Property
Standard Template Library of Avatar
Functional Language for Smart Contract
Code Template Upgrade System Manager

References

- 《PoW, PoS, & Hybrid protocols: A Matter of Complexity?》https://arxiv.org/pdf/1805.08674.pdf
- 《2-hop Blockchain: Combining Proof-of-Work and Proof-of-Stake Securely》https://eprint.iacr.org/2016/716.pdf
- https://github.com/mvs-org/mips/blob/master/mips/mip-2.md
- https://github.com/Nevacoin/nevacoin
- https://github.com/dashpay/dash/issues/2268
- https://github.com/mvs-org/mips/blob/master/mips/mip-16.md
- http://newmetaverse.org/white-paper/Metaverse-whitepaper-v3.0-EN.pdf