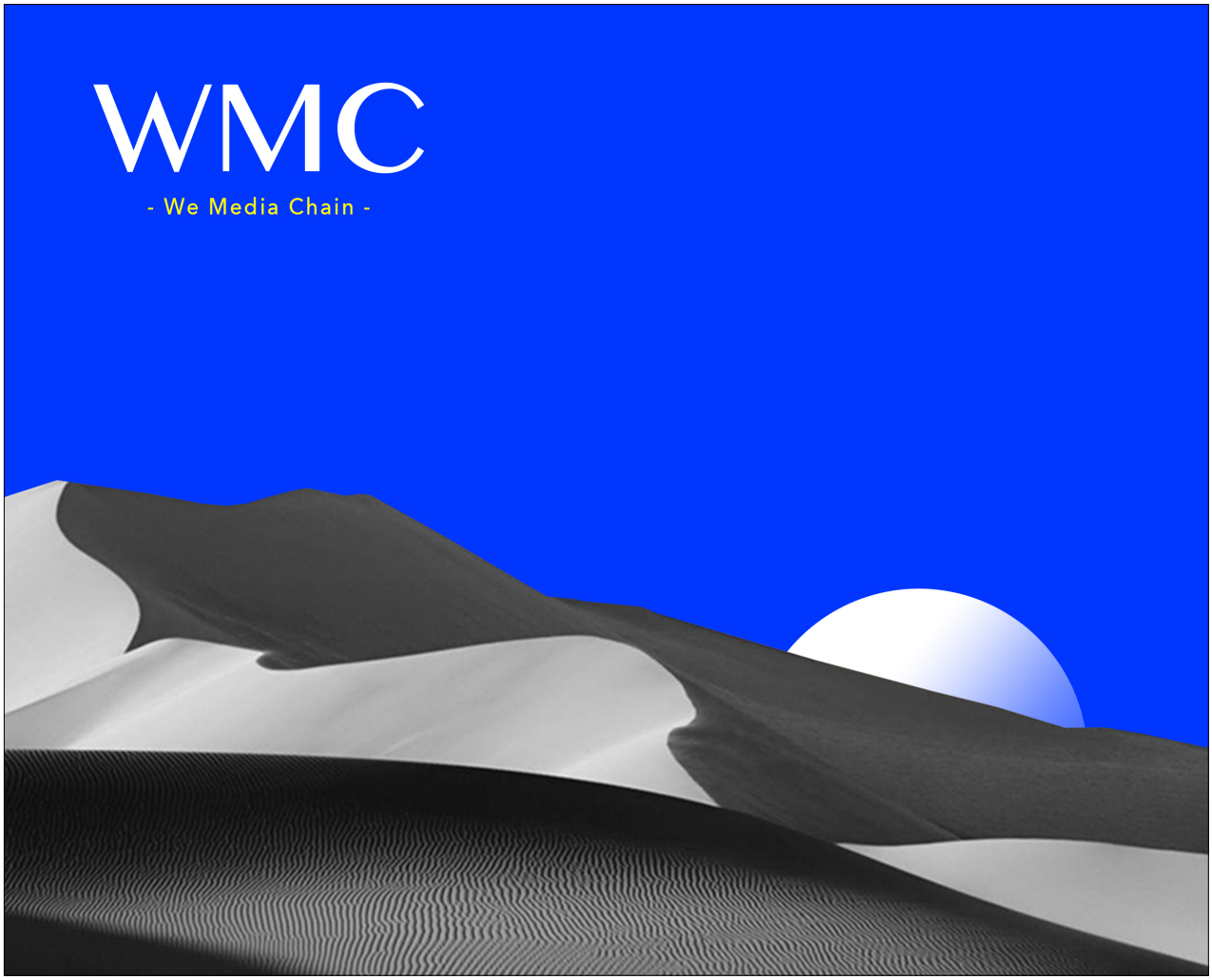


We Media Chain

自媒区块网络（海姆达尔协议）白皮书 0.8

修订日期: 2018年5月18日

第一章、行业背景

以比特币为代表的第一代区块链技术，通过去中心账本，解决了资产中心发行的问题，为我们带来了全新的货币避险及流通方案，保护了货币持有者的利益，避免了发行机构对个人财富的掠夺。

以以太坊为代表的第二代区块链技术，通过智能合约，提供了去中心证券的可能性，为我们带来了全新的资产证券化及设计金融规则的可能性，极大地提升了早期项目证券化的效率，以去中心的区块链网络为信用背书，为早期项目投资及证券规则设计提供了全新的思路及工具。

那么，仅仅资产或者价值上链，是否是区块链的终极形态，是否有更大的领域也需要利用区块链来降低高昂的信任成本，提升信任效率？让“代码成为宪法”，创建出新的社会组织形态。

我们认为，

由价值（货币）组成的区块链网络（经济体），

将会被，

由数据（人）组成的区块链网络（社会），

所容纳，所替代，

由“算力投票”构成的“工作量证明PoW”，One CPU One Vote（算力决定控制权，代表持币者利益）

将会被，

由“数据投票”构成的“数据个体选举”，One Person One Vote（个人决定控制权，代表个人利益）

所替代。

真正实现区块链的控制者、受益者、使用者为同一个群体，真正去中心的全新区块链网络。

自媒链旨在构建区块链上的数据身份证，汇集其贯穿个人一生的所有数据，形成具有连贯的、完整的、生动的、难以舍弃的个人数据集合，也就意味着，每个区块链地址，将直接代表一个真是存在的人，而这个“人”（个人数据集合）将可以参与到各种各样的“事”（评分智能合约）中去，实现“数据”与“价值”的首次结合。

以下会概述几个围绕“数据、人、事、价值”的具体行业诉求。

1.1、以销售线索为目的的媒体营销

对于企业而言，销售线索的获取尤为重要，企业诉求展示与点击效果的同时，更希望获得意向客户的信息，来帮助他们更好的营销和推广产品。因而构建一个消费者托管个人身份标识及隐私数据、自媒体提供推广渠道、企业直接从链上获取消费者授权的身份信息且通过推广渠道获得意向的分布式系统，变得十分有必要。

消费者通过向特定的机构开放信息获得更好，更精准的服务的同时，也能获取对应的收入，将本来应用在彼此不透明的信息猜忌模式里的费用，转移给真正提供信息和获得服务的消费者，最大化每个消费者的利益。

企业和机构也不再需要通过层层漏斗获得自己希望的客户，只需要在链上支付费用并设定筛选条件，就能得到想要的流量及用户。

第二章、技术细节

2.1、数字货币总量

WMC的数字货币为WeMediaCash（WMCH）

WMCH总量为15亿，不会进行增发，

在主网上线前WMCH为以太坊上的ERC20代币，在这个阶段释放的WMCH将在主网上线时在创始块中做映射。

2.1.1、区块奖励策略

主网上线前，

各个行业的DApp将会通过各自累积的数据（人）数量获得不超过1.5亿（10%）WMCH的激励。

主网上线后，

价值交易网络将会以“Proof of Work”的形式，

向矿工提供总量不超过3亿（20%）WMCH的区块奖励，

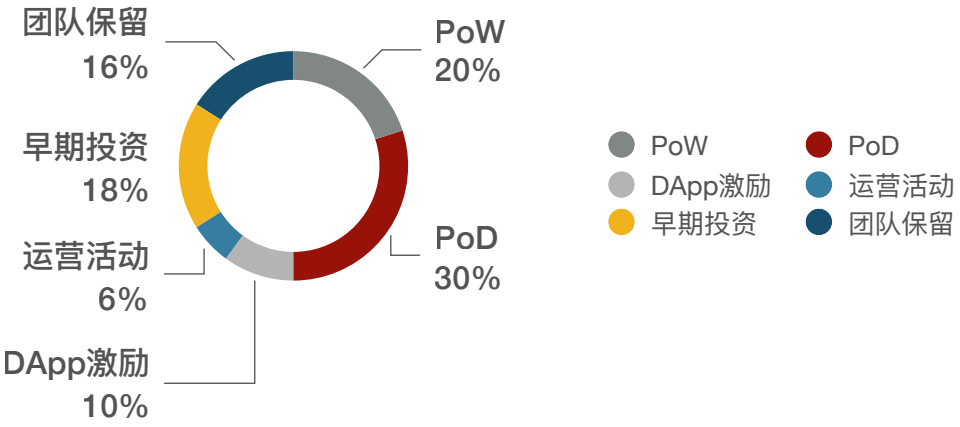
~PoW即提供越多算力的矿工，有更大的机会获得奖励。

数据仓库节点将会以“Proof of Data”的形式，

向数据代理人提供总量不超过4.5亿（30%）WMCH的数据仓库节点奖励，

~PoD即托管越多人数据的节点，有更大的机会获得存储及处理数据的机会，进而获得节点奖励。

2.1.2、比例



16%用于团队激励，总计2.4亿，主网上线前不进入流通，

06%用于运营活动，总计0.9亿，

10%用于应用激励，总计1.5亿，

18%用于早期投资，总计2.7亿，

早期投资比例：不高于 4000 ETH : 2.7亿 WMCH

最低投资额度：不少于 100 ETH

早期投资锁仓：6~12月

50%通过挖矿获得，总计7.5亿，其中PoW 20% 3亿，PoD 30% 4.5亿。

2.1.3、主网上线前的挖矿

主网上线前，为扩大用户规模，向授权我们使用数据的用户提供代币收入，借此吸引更多的DApp和消费线索购买机构加入。

1.5亿代币将在第一年释放50%，第二年释放25%，第三年12.5%，以此类推。

第一年每天共投放20.5479万，将分为两部分投放

1）数据交易，机构及开发者支付购买消费线索时支付的WMCH，将占用当日产量。

2）挖矿奖励，除去数据交易部分的WMCH，剩余的当日产量，将按照个人当日收入

$$I = (205479 * (\frac{1}{2})^{c-1} - Y) * \frac{Xm}{\sum_{i=1}^{n} Xi}$$

计算收益，其中

Y代表机构及开发者数据交易使用的WMC的总量

c代表距创始日年份（1…n）

Xm代表用户个人的收益系数，n代表用户总量

Xi代表每个用户的收益系数。

如果总难度系数为1,000,000，个人难度系数为6，在空投开始第一年，当日机构购买消耗了1,000,000WMC，则该用户当日收入为

$$(205479 * (\frac{1}{2})^{1-1} - 1000000) * \frac{6}{1000000} = 0.232877$$

2.1.4、机构及开发者使用WMCH

初期机构在媒体及应用上发生数据交易时

消费者可根据自己信息丰富程度来获得WMCH，（动态价格）

由消费者设定维度基础价格，机构在设计评分算法时，会增加复杂程度，得到难度系数，其乘积将直接从机构扣除转移给消费者。

2.2、算力证明方式（POW）

2.2.1、PoW

工作量证明，Proof of Work，通过计算来猜测一个数值（nonce），得以解决规定的 hash 问题（来源于 hashcash ）。保证在一段时间内，系统中只能出现少数合法提案。

同时，这些少量的合法提案会在网络中进行广播，收到的用户进行验证后会基于它认为的最长链上继续难题的计算。因此，系统中可能出现链的分叉（Fork），但最终会有一条链成为最长的链。

hash 问题具有不可逆的特点，因此，目前除了暴力计算外，还没有有效的算法进行解决。反之，如果获得符合要求的 nonce，则说明在概率上是付出了对应的算力。谁的算力多，谁最先解决问题的概率就越大。当掌握超过全网一半算力时，从概率上就能控制网络中链的走向。这也是所谓 `51% 攻击` 的由来。

参与 PoW 计算比赛的人，将付出不小的经济成本（硬件、电力、维护等）。当没有成为首个算出的“幸运儿”时，这些成本都将被沉没掉。这也保障了，如果有人恶意破坏，需要付出大量的经济成本。也有设计试图将后算出结果者的算力按照一定比例折合进下一轮比赛考虑。

有一个很直观的例子可以说明为何这种经济博弈模式会确保系统中最长链的唯一。

超市付款需要排成一队，可能有人不守规矩要插队。超市管理员会检查队伍，认为最长的一条队伍是合法的，并让不合法的分叉队伍重新排队。只要大部分人不傻，就会自觉在最长的队伍上排队。

2.2.2、Hashing

Hash，一般翻译做“散列”，也有直接音译为“哈希”的，就是把任意长度的输入（又叫做预映射， pre-image），通过散列算法，变换成固定长度的输出，该输出就是散列值。这种转换是一种压缩映射，也就是，散列值的空间通常远小于输入的空间，不同的输入可能会散列成相同的输出，而不可能从散列值来唯一的确定输入值。简单的说就是一种将任意长度的消息压缩到某一固定长度的消息摘要的函数。

HASH主要用于信息安全领域中加密算法，它把一些不同长度的信息转化成杂乱的128位的编码,这些编码值叫做HASH值. 也可以说，hash就是找到一种数据内容和数据存放地址之间的映射关系。

2.2.3、Hashcash

“Hashcash是一种用于防止垃圾电子邮件和拒绝服务攻击的工作量证明系统，最近以其在比特币（以及其他加密货币）挖矿算法中的应用而闻名，由Adam Back于1997年3月提出。”（维基百科）你可以点击 这里 阅读Adam Back的论文。

一条消息（例如一封电子邮件）通过包含一些字符串的散列值，证明计算机花费了一些时间或能量在特定的算法上，以“证明”它是合法的消息，具体方法是计算一个 SHA-1 散列使得散列值的前20位为0。因为需要一定的计算时间来通过暴力计算找到这样一个合格的散列值，所以发送者需要花费一些成本来计算散列值，这对于发送大量电子邮件的垃圾邮件发送者来说是不现实的。Hashcash可以被视为“帮助Hashcash用户避免因基于内容和基于黑名单的反垃圾邮件装置导致电子邮件丢失的白名单。”（hashcash.org）

这种“工作量证明”的概念现在主要用于比特币挖矿功能，“充当区块链更新的投票机制，并验证区块链交易日志。”或者换句话说：“比特币采用Hashcash，通过收取一笔用于补偿矿工所希望得到的合作激励作为更新费用，来实现防止区块链被恶意篡改的安全性……在比特币中，Hashcash问题的困难性随着时间的推移而变化，取决于最近解决时间的记录，目标为平均10分钟完成一次。“（The Book of Bitcoin）

1. 获取某种公开数据data（在反垃圾邮件场景下，使用收件人地址；在比特币中，使用block头信息）
2. 使用一个计数器counter，初始化为0
3. 计算data+counter的hash值
4. 检查hash值是否满足某种要求
    1. 满足，结束
    2. 不满足，counter加1，然后重复3-4步骤

这是个暴力型算法：改变counter值，计算得到新hash值，判断该值是否满足要求，不满足，再改变counter值，再计算，再检查…如此反复尝试直到得到满足要求为止，要求很高的算力。

来我们看看hash要满足什么要求，hashcash最初实现中，hash值要满足“头20个bit全部为0”的要求。比特币设计中，hash值要求是动态变化的，随着时间和矿工的增多，算力要求也越来越多。

为了展示算法，使用上面示例中的数据（“I like donuts”），不停的计算该数据+counter的hash值，直到找到一个满足要求（“前3个字节为0”）的hash值为止，
“I like donutsca07ca”中的 **ca07ca** 是十六机制格式的值，即counter从0递增到13240266时计算得到满足要求的hash值。
代码实现

```
package main
import (
	"bytes"
	"crypto/sha256"
	"encoding/binary"
	"fmt"
	"log"
	"math"
	"math/big"
	"strconv"
	"time"
)
// 定义block结构体
type Block struct {
	Timestamp     int64
	Data          []byte
	PrevBlockHash []byte
	Hash          []byte
	Nonce         int
}
// 创建一个block对象
func NewBlock(data string, prevBlockHash []byte) *Block {
	block := &Block{time.Now().Unix(), []byte(data), prevBlockHash, []byte{},
0}
	pow := NewProofOfWork(block)
	nonce, hash := pow.Run()
	block.Hash = hash[:]
	block.Nonce = nonce
	return block
}
// 创建一个创世块
func NewGenesisBlock() *Block {
	return NewBlock("Genesis Block", []byte{})
}
// 定义区块链结构体 是一block slice .
type Blockchain struct {
	blocks []*Block
}
//  增加一个区块
func (bc *Blockchain) AddBlock(data string) {
	prevBlock := bc.blocks[len(bc.blocks)-1]
	newBlock := NewBlock(data, prevBlock.Hash)
	bc.blocks = append(bc.blocks, newBlock)
```

```
}
// 创建一个带有创世块的区块链
func NewBlockchain() *Blockchain {
	return &Blockchain{[]*Block{NewGenesisBlock()}}
}
var (
	maxNonce = math.MaxInt64
)
const targetBits = 24
// ProofOfWork 结构体
type ProofOfWork struct {
	block  *Block
	target *big.Int
}
// NewProofOfWork 创建一个 ProofOfWork
func NewProofOfWork(b *Block) *ProofOfWork {
	target := big.NewInt(1)
	target.Lsh(target, uint(256-targetBits))
	pow := &ProofOfWork{b, target}
	return pow
}
func (pow *ProofOfWork) prepareData(nonce int) []byte {
	data := bytes.Join(
		[][]byte{
			pow.block.PrevBlockHash,
			pow.block.Data,
			IntToHex(pow.block.Timestamp),
			IntToHex(int64(targetBits)),
			IntToHex(int64(nonce)),
		},
		[]byte{},
	)
	return data
}
// POW 机制验证的实现
func (pow *ProofOfWork) Run() (int, []byte) {
	var hashInt big.Int
	var hash [32]byte
	nonce := 0
	fmt.Printf("Mining the block containing \"%s\"\n", pow.block.Data)
	for nonce < maxNonce {
		data := pow.prepareData(nonce)
		hash = sha256.Sum256(data)
```

```
            fmt.Printf("\r%x", hash)
            hashInt.SetBytes(hash[:])
            if hashInt.Cmp(pow.target) == -1 {
                  break
            } else {
                  nonce++
            }
      }
      fmt.Print("\n\n")
      return nonce, hash[:]
}
// 认证
func (pow *ProofOfWork) Validate() bool {
      var hashInt big.Int
      data := pow.prepareData(pow.block.Nonce)
      hash := sha256.Sum256(data)
      hashInt.SetBytes(hash[:])
      isValid := hashInt.Cmp(pow.target) == -1
      return isValid
}
// 将int64 转换 []byte
func IntToHex(num int64) []byte {
      buff := new(bytes.Buffer)
      err := binary.Write(buff, binary.BigEndian, num)
      if err != nil {
            log.Panic(err)
      }
      return buff.Bytes()
}
func main() {
      bc := NewBlockchain()
      bc.AddBlock("Send 1 BTC to 土拨鼠")
      bc.AddBlock("Send 2 more BTC to 土拨鼠")
      for _, block := range bc.blocks {
            fmt.Printf("Prev. hash: %x\n", block.PrevBlockHash)
            fmt.Printf("Data: %s\n", block.Data)
            fmt.Printf("Hash: %x\n", block.Hash)
            pow := NewProofOfWork(block)
            fmt.Printf("PoW: %s\n", strconv.FormatBool(pow.Validate()))
            fmt.Println()
      }
}
const targetBits = 24
```

在比特币中，当一个块被挖出来以后，“target bits” 代表了区块头里存储的难度，也就是开头有多少个 0。这里的 24 指的是算出来的哈希前 24 位必须是 0，如果用 16 进制表示，就是前 6 位必须是 0，这一点从最后的输出可以看出来。目前我们并不会实现一个动态调整目标的算法，所以将难度定义为一个全局的常量即可。

24 其实是一个可以任意取的数字，其目的只是为了有一个目标（target）而已，这个目标占据不到 256 位的内存空间。同时，我们想要有足够的差异性，但是又不至于大的过分，因为差异性越大，就越难找到一个合适的哈希。

```
type ProofOfWork struct {
      block  *Block
      target *big.Int
}
func NewProofOfWork(b *Block) *ProofOfWork {
      target := big.NewInt(1)
      target.Lsh(target, uint(256-targetBits))
      pow := &ProofOfWork{b, target}
      return pow
}
```

ProofOfWork 结构，里面存储了指向一个块(`block`)和一个目标(`target`)的指针。这里的 “目标”，也就是前一节中所描述的必要条件。这里使用了一个 大整数，我们会将哈希与目标进行比较：先把哈希转换成一个大整数，然后检测它是否小于目标。

2.2.4、Proof Of Dataset

为了激励拥有数据的节点更长时间在线，我们必须设计一种能够奖励活跃数据的全新共识机制，来保证当有真正的数据检索及BI运算需求的时候，有足够的在线数据集数量支持网络中交易的正常运转。

由此我们会产生数据隐私敏感性与数据执行便利性之间的冲突，我们将会通过关键数据委托的形式，来平衡隐私敏感与执行便利之间的关系，

我们的数仓节点数量约为Dataset数量2^16倍，也就是说，当我们拥有2^32（约为42.94亿）数据集时，我们将拥有2^16（约为65536）个数仓节点，每个节点将平均代理65536份数据集，或者分别代理N*2^16份数据集1/N的数据，则每个Dataset能够向N个节点同时委托自己部分或者全部的数据。

```
type DatasetNode struct {
      Delegates          map[*Address]*Sign       // 所有委托人的投票签名信息
      LastHeight         *big.Int                 // 上次派息的区块高度
}

func (p *Person) Delegate(node *NodeClient) {
      signStr := node.Address().ToHex() + ":" + p.Address().ToHex()
      sign := p.Sign(signStr).ToHex()
      node.Vote(sign)
}
```

个人在向数仓节点委托自己的数据时，会将节点的地址及自己的地址加签，来保证数仓网络总的所有节点都可以效验委托投票的真实性。

收益分配方案（一）

数仓网络将在每个区块高度协商出一个一致的收益分配方案，提供给算力网络，算力网络中每个区块产生时，会参考数仓网络提供的分配方案，将各个数仓节点地址对应的收益写入区块的第二笔交易中。数仓节点根据自己的委托情况，扣除手续费后向用户分配这些收益。

```
type DatasetShareApplication struct {
      Vote         map[*Address]*big.Int          // 所有节点的投票信息
      Height       *big.Int                       // 将要派息的区块高度
      Share        map[*Address]*big.Ing          // 所有节点的收益信息
      Signs        map[*Address]string            // 所有节点对这一提案的签名
}
```

如果在某个高度时，数仓节点未广播自己的委托信息，则本轮收益分配时，该数仓节点无法得到本区块高度的奖励，基于这个机制，更多的用户将会把数据委托给在数仓网络中更加稳定的节点，进而完成数仓节点的自我筛选。

收益分配方案（二）

数仓网络将同时维护一条侧链，算力网络中的数仓奖励将会由这条侧链进行分配，最终将结果不定期合并到主链上。

```
type DatasetShareApplication struct {
      Vote         map[*Address]*big.Int          // 所有节点的投票信息
      Height       *big.Int                       // 本次派息的区块高度
      Share        map[*Address]*big.Ing          // 所有节点的收益信息
      Signs        map[*Address]string            // 所有节点对这一提案的签名
}
```

侧链将允许数仓网络不需要强制与算力网络保持协同，数仓网络将有更多的时间去效验各个节点的投票真实性。

收益分配方案（三）

算力网络会根据上次收益的分配方案，选举下次进行收益分配提案的节点。

```
func (n *WorkNetwork) Select(last *DatasetShareApplication) {
      // 以上次的提案节点信息为种子
      rand := NewRander(last.LastMaster+last.Height)
      // 取的本次提案节点的随机
      v := rand.Next()
      // 按照投票数量高获选概率高的规则选举本次提案的节点
      master := SelectNode(last.Vote, v)
```

```
}

type DatasetShareApplication struct {
      Vote         map[*Address]*big.Int          // 所有节点的投票信息
      Height       *big.Int                       // 本次派息的区块高度
      Share        map[*Address]*big.Ing          // 所有节点的收益信息
      Master       *Address                       // 本次进行提案的节点
}
```

新的提案节点将根据上一次的提案节点及投票比例，随机、与投票正相关、可回溯的产生，由他进行投票数效验，收益计算及发起新的提案。

2.3、角色

2.3.1、PD（PersonalDataset）、用户数据集

用户（数据集）为现实世界中个人在区块链世界中的映射，这个映射过程是通过海姆达尔协议完成的，在保证个人数据集的隐私安全基础上，提供一个高效、透明、可信的数据集运算环境及交易平台。
个人（数据集）有了价值，便可以以真实世界个人的映射方式，参与区块链上的应用及活动中。
用户可以将数据集（脱敏后部分）委托给数仓节点，来离线参与链上的活动。

2.3.2、WN（PoW Network）算力网络

算力网络为区块链世界中的交易及智能合约的执行提供去中心的运行环境。

2.3.3、M（Miner）矿工

矿工通过算力竞争打包算力网络中的交易，保证算力网络的去中心特性，同时算力越高，获得更多奖励的概率越大，进而能够不断地吸引新的算力来保证网络的稳定运行。

2.3.4、DN（PoD Network）数仓网络（数据仓库网络）

数仓网络通过分块存储加密的用户数据，接受用户的数据委托，来提升数据运算及交易的效率。

2.3.5、N（Dataset Node）数仓节点

用户委托数将会选举出一定数量的数仓节点，来托管及代理运行用户的数据。

2.3.6、O（Organization）机构

机构发布BI合约来精准获得用户数据的运算结果。

2.3.7、D（Developer）开发者

开发者为机构和用户之间的数据交易建立桥梁，针对不同的领域，开发不同的DApp，帮助用户更安全更便捷的获取服务，同时也让机构能够更直接的与用户建立起联系。

2.3.8、D（Developer）开发者

2.4、技术规范

WeMediaChain使用智能合约保证透明，去中心化的，不可篡改的，高可靠性的基础。具有去中介化的信任，稳定性可靠性和持续性，强安全共识机制，交易的公开透明不可篡改基本特征。借助智能合约，媒体相关的所有数据都可以通过合约代码来描述，并且保留了区块链技术的优点。

为了实现所描绘的应用场景，我们将通过如下的实际技术解决方案来为链上的应用提供信用背书和技术保障。

2.4.1、ERC20代币，主网上线前的权益确认方案

在主网上线前采用符合 ERC20代币标准，可兼容以太坊钱包，方便用户进行存储、转账、交易等操作。通过ERC20代币实现去中心化的权益确认，WeMediaCoin的ERC20代币与主网的WeMediaCoin将具备相同的价值，这些代币将在主网上线时通过映射转移到对应的WeMediaCoin主网钱包中。

WeMediaCoin的ERC20代币在主网上线前会提供两种存储模式：

1、运营者可在合作机构官网查询到自己钱包中的余额，可进行站内转账、充值、提现等操作；提现操作会扣除WeMediaCoin的ERC20代币作为交易手续费，站内转账、充值不扣除手续费；

2、运营者可以下载桌面钱包、Chrome扩展MetaMask，在本地生成冷钱包，通过模式一中的体现动作将WeMediaCoin的ERC20代币提现到自己的钱包中，提现完成后，转账动作将扣除ETH作为矿工费，但是在以后我们将实现WeMediaCoin的ERC20代币支付矿工费的能力。

2.4.2、星际文件系统（INTERPLANETARY FILE SYSTEM，IPFS)，去中心的流量追踪及资源访问方案

IPFS的全称是InterPlanetary File System星际文件系统，是一个点对点的网络超媒体协议。它的目标是成为更快、更安全、更开放的下一代互联网。

IPFS将加密后的数据分块存储在由IPFS矿工组成的去中心存储网络上，可以随时随地高效安全的调取需要的数据，因为数据回源来自于最近的节点，所以能很好解决最后一公里传输性能的问题，而因为数据被分块且加密存储在分布式的节点上，数据被盗、遭受DDoS攻击的风险也大大下降。

WeMediaChain中的用户敏感数据、私钥、指纹等信息将隐匿存储在WeMediaChain中，资源提交较大、安全需求程度低、访问频繁的资源会通过私钥加密后储存在IPFS中，提供更为高效的访问、检索模式。

InterPlanetary文件系统（IPFS）是一个内容可寻址的分布式文件系统，它保证由其加密散列标识的文件内容的固定性。文件通过对等网络延迟解决。然而，内容寻址引用本质上是不可变的，因此在每个应用程序中都不实用。例如，如果HTML网页使用其引用嵌入图像，则每次更新图像时都需要更新引用，否则网页仍将引用旧版本的图像。如果许多网页中包含相同的图像，则所有这些图像都需要更新，因此它们自己的哈希值也会改变。这具有级联效应，并且通过引用而非价值来杀死包括对象的主要目的，以实现关注和重用的分离。

为了解决这个问题，IPFS使用InterPlanetary命名系统（IPNS），该系统提供从人类可读URI到其对应的当前IPFS哈希的映射。域名的所有者可以通过用他/她的私钥对请求进行签名来更新该域下所有URI的映射。IPNS可以以多种方式实现，但其当前的实现使用分布式哈希表（DHT）。因此，只有每个URI与其对应的散列的最近映射才可用于解析，而忽略任何历史映射。从档案的角度来看，这并不好，因为以前的文件版本可能仍然存在于IPFS存储中，但是其对应的URI映射却会丢失。

传统的网络档案可能仍然有一些历史观察，可以使用给定的URI来检索旧版本的文件，但这些记录将在IPFS系统之外，并且历史可能是稀疏的而不是事务性的。

我们可以通过对IPNS记录使用区块链来解决这些问题。通过这样做，IPFS可以像事务性存档引擎一样工作，同时将所有历史“URI - >哈希”映射保留在公共区块链中。使用IPNS Blockchain解析URI应该返回当前映射，而使用Datetime解析URI应该返回当时存在的映射。该备忘录框架可用于基于时间的脉冲中子源的分辨率。

WeMediaChain在为机构或者开发者提供流量追踪功能时，会将投放资源存储在IPFS并提供IPNS下更短更便于传输和记忆的资源域名，并同时提供访问追踪的能力，杜绝传统流量追踪方案中中心作假及数据不透明的问题，将会出现更多第三方清洗流量的应用，最大程度的规避流量作弊的灰色手段。

2.4.3、数据公证方案，数据公证人规则

对于WeMediaChain在执行智能合约时需要的真实数据问题，我们会采取数据公证人制度对数据提供者（见证人）进行群体智慧验证，由于这些真实数据是在现实世界已经发生的，此时投票的人通常已经从现实世界获取了真是的数据，大部分人会给出肯定或者否定的投票。我们使用80%（而非51%，因为这不仅仅是奖励及规则的竞争，而关系到数据的真实性）这个阈值来决定是否让结果生效。而数据提供者会缴纳押金，进而保证诚实节点及投票者能获得更多的利益。

2.4.4、数据脱敏及阻止二次贩卖方案，零知识证明&智能评分合约

零知识证明(Zero—Knowledge Proof)，是由S.Goldwasser、S.Micali及C.Rackoff在20世纪80年代初提出的。它指的是证明者能够在不向验证者提供任何有用的信息的情况下，使验证者相信某个论断是正确的。零知识证明实质上是一种涉及两方或更多方的协议，即两方或更多方完成一项任务所需采取的一系列步骤。证明者向验证者证明并使其相信自己知道或拥有某一消息，但证明过程不能向验证者泄漏任何关于被证明消息的信息。大量事实证明，零知识证明在密码学中非常有用。如果能够将零知识证明用于验证，将可以有效解决许多问题。

在有必要证明一个命题是否正确，又不需要提示与这个命题相关的任何信息时，零知识证明系统(也叫做最小泄露证明系统)是不可或缺的。零知识证明系统包括两部分：宣称某一命题为真的示证者(prover)和确认该命题确实为真的验证者(verifier)。证明是通过这两部分之间的交互来执行的。在零知识协议的结尾，验证者只有当命题为真时才会确认。但是，如果示证者宣称一个错误的命题，那么验证者完全可能发现这个错误。这种思想源自交互式证明系统。交互式系统在计算复杂度理论方面已经获得异常独立的地位。

零知识证明(Zero—Knowledge Proof)起源于最小泄露证明。设P表示掌握某些信息，并希望证实这一事实的实体，设V是证明这一事实的实体。假如某个协议向V证明P的确掌握某些信息，但V无法推断出这些信息是什么，我们称P实现了最小泄露证明。不仅如此，如果V除了知道P能够证明某一事实外，不能够得到其他任何知识，我们称P实现了零知识证明，相应的协议称作零知识协议。

在通常的大数据平台中，数据以结构化的格式存储，每个表有诸多行组成，每行数据有诸多列组成。根据列的数据属性，数据列通常可以分为以下几种类型：

1、可确切定位某个人的列，称为可识别列，如身份证号，地址以及姓名等。
2、单列并不能定位个人，但是多列信息可用来潜在的识别某个人，这些列被称为半识别列，如邮编号，生日及性别等。美国的一份研究论文称，仅使用邮编号，生日和性别信息即可识别87%的美国人。
3、包含用户敏感信息的列，如交易数额，疾病以及收入等。
4、其他不包含用户敏感信息的列。

所谓避免隐私数据泄露，是指避免使用数据的人员（机构、DApp开发者、数据分析师，BI工程师等）将某组数据识别为某个人的信息。数据脱敏技术通过对数据进行脱敏，如移除识别列，转换半识别列等方式，使得

数据使用人员在保证可对#2半识别列（转换后）、#3敏感信息列、#4其他列进行数据分析的基础上，在一定程度上保证其无法根据数据反识别用户，达到保证数据安全与最大化挖掘数据价值的平衡。

通常的数据脱敏算法包括以下几种，

名称	描述	示例
Hiding	将数据替换成一个常量，常用作不需要该敏感字段时。	500 —> 0 635 —> 0
Hashing	将数据映射为一个hash值（不一定是一一映射），常用作将不定长数据映射成定长的hash值。	Jim, Green —> 4563934453 Tom, Cluz —> 4334565433
Permutation	将数据映射为唯一值，允许根据映射值找回原始值，支持正确的聚合或连接操作。	Smith —> Clemetz Jones —> Spefde
Shift	为数量值增加一个固定的偏移量，隐藏数值部分特征。	253 —> 1253 254 —> 1254
Enumeration	将数据映射为新值，同时保持数据顺序。	500 —> 25000 400 —> 20000
Truncation	将数据尾部截断，只保留前半部分。	021-66666666 —> 021 010-88888888 —> 010
Prefix-preserving	保持IP前n位不变，混淆其余部分。	10.199.90.105 —> 10.199.32.12 10.199.90.106 —> 10.199.56.192
Mask	数据长度不变，但只保留部分数据信息。	23454323 —> 234---23 14562334 —> 145---34
Floor	数据或是日期取整	28 —> 20 20130520 12:30:45 —> 20130520 12:00:00

除此之外，根据实际的业务场景我们还可以通过维度转换的方式，保证更大的数据安全级别同时满足企业数据挖掘的最大化收益，

假设一家金融机构评估目标用户的逾期可能性P

$$\mathscr{P} = f_s(f_1(Y_1), f_2(Y_2), f_3(X_1), f_4(X_2), f_5(X_3))$$

其中，

F1-F5为机构评估各个维度的标准化算法

Fs为机构为各个维度加权的算法

Yi为机构已知数据

Xi为用户敏感数据

而如果我们能找到函数Fx使得

$$X = f_x(fa(X_1), fb(X_2), fc(X_3))$$

$$\mathscr{P} = f_{s1}(f_1(Y_1), f_2(Y_2), f_n(X))$$

就可以使得机构在不知道Xi的情况下，得到诉求的结果P。

而因为Fx得到的结果X，与机构的算法Fsi及实际业务有着密切的关系，因此另外一家机构获得X的值对自身业务也是没有任何帮助的。

甚至进一步杜绝有相同需求的机构共享用户信息，我们可以使用用户的公钥加密Fx，用户本地使用私钥解密并运行算法Fx后，使用机构公钥加密X，返回给机构，机构不会向自己的竞争对手提供Fx的详情，因而机构贩卖X到市场中并不会有人为其买单。

2.4.5、阻止骚扰，私密通道

在传统中心化的营销体系中，通过电话及IM工具与消费者直接通信一直以来都存在这严重的问题，不仅机构可以肆意出售消费者的联系方式（此类问题已经通过私密专线等问题得以解决，但提供私密专线的中心仍旧保有用户的联系方式），且用户无法定向阻止机构的进一步骚扰电话。

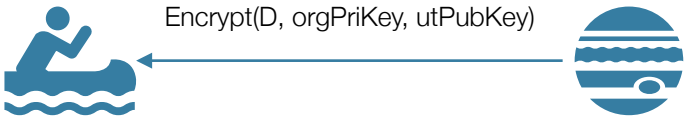
我们将通过基于时间种子的加密算法，为支付费用的机构建立起与消费者的私密通道，通过我们合作的各种DApp提供机构与消费者的通信通道，与此同时，一旦超过了通信的约定时间，机构向消费者的消息传输就会被中断。整个过程中不会有用户的私密联系方式流出，最大程度的保障了消费者的隐私。

我们认为在机构与消费者的通信过程中，消费者是处于相对弱势的地位，消费者使用通信产品的目的是为了与自己的朋友、家人通信，而非接收营销信息，这不仅仅指电话，也指那些中心化的IM软件，诸如QQ、微信、Facebook、Line等等，营销往往是这些中心化的通信产品的核心商业模式，在现代商业社会这无可厚非，但是随着越来越多的安全、私密、去中心的通信产品诞生，消费者将拥有更多在通信产品中的主动权，可以选择接受哪些或者不接受哪些信息；

基于这个即将发生变革，我们相信，搭建一条从机构到消费者，单向的，在一定时间内可以传输有效信息的通信通道，将是机构未来能与消费者取得联系的唯一通道。

而这一特性，将广泛应用到我们合作的去中心通信产品中，成为有效隔离骚扰同时又能为开发者和消费者带来收入的全新营销渠道。

机构消息传送时的加密过程可以简化描述为如下图示：



机构加密消息的过程如下：

$$utMessage = AES(encrypt(D, utPubKey), utPropSigned)$$

机构需要推送的数据D，会通过用户许可的一定时间有效的临时钥匙对utKey的公钥进行加密，得到仅用户能解密的隐匿消息体_utMessage，并使用utPropSigned进行对称加密，得到utMessage。

用户在公布utPubKey（用户许可的一定时间有效的临时钥匙对中的公钥）时，会使用utPriKey加密utProp（用户许可的一定时间有效的临时钥匙对中的时间信息）得到utPropSigned。

$$orgMessage = encrypt(orgMessage + utPropSigned, orgPriKey)$$

机构使用自己的私钥orgPriKey加密utMessage及utProp得到携带机构签名的orgMessage，

并将orgMessage，utPropSigned广播到区块网络中。

节点接收到消息后会经历以下的效验过程

1）节点使用机构广播的公钥解密orgMessage，得到utMessage及utPropSigned

2）节点使用用户广播的utPubKey解密utPropSigned得到utProp，

3）节点检查utProp生命的时间是否仍旧有效，如果已经超出utProp声名的时间，则拒绝该消息的传播

4）节点推送orgMessage到用户终端

用户接收到消息后会通过以下过程获取消息内容

1）通过自己已知的授权机构，确认消息来自自己授权的机构。

2）使用orgPubKey解密得到utMessage及utPropSigned

3）检查utPropSigned是否为自己签发，以及时间是否合法

4）解密得到消息体D

2.5、钱包

2.5.1、ERC20代币钱包

代币钱包提供冷钱包的基本能力，可以通过本地生成新的私钥及地址，来保存从合作机构提供的WMC；
代币钱包生成的钱包地址可接受WMC代币及ETH，由于ETH网络的限制，转移WMC需要支付ETH手续费，因而当WMC提现到冷钱包之后，转账需向冷钱包地址存入相应的ETH方可进行；
在ETH随后的升级过程中，我们会增加合约余额用户支付用户转移WMC的手续费，相应的会扣除一定的WMC；
因而现在本地钱包多用于余额汇集和长期存储的能力，如果需要进行转账可以通过合作机构提供的内部转账能力进行转账；

第三章、应用场景

3.1、追溯自媒体价值

成交价格及媒体数据将会写入区块链，无法篡改，新的交易会根据以往的交易信息进行评估；而媒体数据将会由见证人共同确认，避免作弊行为的发生；

3.2、合约交易

运用智能合约设定流量交易模式并经发布后不可修改

3.2.1、条件支付模式

媒体数据达到不同的级别，支付不同的WMC；

3.2.2、按量支付模式

媒体数据根据公式计算得到应支付的WMC；

3.2.3、收益转让

当媒体所有者发生变化时，能将价值及未来的收益过渡给新的所有者，变更一旦发生并且得到节点确认，就无法被撤回，原所有者也将得到相应的WMC；

3.3、数据交易

运用智能评分合约脱敏用户数据，并向机构提供有价值的维度评分数据。

3.3、消费线索交易

超级节点将提供用于追踪消费者点击行为基于IPFS的星际域名，当用户点击企业制作的广告同时，将会自动完成一次数据交易，将消费者的消费意向及维度评分提供给企业主并获得WMC。

第四章、WeMediaChain团队

4.1、开发机构

决策委员会，

主要负责制定重要决策、召开紧急会议，以及聘请解聘各职能委员会负责人。首届决策委员会成员将由团队成员以及早期投资人组成，任期为 3 年，期满后重新选出。决策委员会由5名成员构成；

代码审核委员会，

由开发团队中的核心开发人员组成，负责底层技术开发、开放端口开发和审核、各产品开发和审核等等；

财务及人事管理委员会，

主要负责项目募集资金的运用和审核、开发人员薪酬管理、日常运用费用审核等；

市场及公共关系委员会，

负责WeMediaChain技术推广、WeMediaChain产品推广和宣传、对外公告管理、公关维护等等；

4.2、团队介绍

Edward
WMC首席执行官
萌小助核心创始人，曾任职腾讯SNG。拥有多年大数据挖掘及项目管理经验，熟悉P2P网络及并发场景设计。

Disac
WMC首席信息官
萌小助架构师，擅长区块链与中心业务对接及架构设计，丰富的项目管理及工程化架构设计经验。

Aaron
WMC首席执技术官
原今日头条架构师。擅长区块链架构设计大数据、工作量证明算法优化，密码学专家。

第五章、合作机构

5.1、自媒体平台类型

预打包分配机制

公众号在自媒体平台的公众号图文、效果广告，可自主选择将部分流量以WMC形式结算，而不再继续参与流量变现服务；

参与主网上线前的挖矿

用户可以在合作机构及合作机构自媒体的平台上参与主网上线前的挖矿。

充值、转账、提现

运营者在合作机构获得的WMC将存储在运营者的合作机构账号中，在合作机构内的WMC内部转账可随时进行（无手续费），同时获得一个ETH地址，用户也可以从外部转入该ETH地址WMC进行充值（无手续费），用户希望将WMC提现到自己的冷钱包中时，可以支付一定的手续费（根据ETH矿工费动态调整）提现到自己的ETH地址；

5.2、合作机构规范

5.2.1、WMC兑换规则

合作机构在兑换比例不高于2.1.4规定的标准时，可自由约定兑换比例，且流量兑换WMC的兑换行为只能单向进行；

兑换请求将提交至WMC合作机构委员会进行审核，通过后方可完成兑换工作；

5.2.2、接入并接收WMC投放

合作机构应提供可以完全由WMC进行的CPC或图文交易，按照2.1.1规定的标准进行广告投放；

5.2.3、提供WMC充值、提现、内部转移、合作机构间转移的能力

合作机构应提供WMC向支持ERC20标准的冷钱包提现、从支出ERC20标准的冷钱包充值的功能，提现手续费由WMC支付（不超过转出WMC的5%或者100WMC，两者相较取较大值），可自由裁定，充值不应扣除手续费；

合作机构内部WMC转账应免除手续费；

合作机构机构间转账应按照转出机构的标准收取手续费（不超过转出WMC的5%或者100WMC，两者相较取较大值），接收机构不再收取手续费；

第六章、开发规划

6.1、第一阶段2018年4月 - 2018年6月

6.1.1、能够进行简单的信息打包，并以此为依据产生预分配的代币

6.1.2、完成钱包

6.1.3、支持合作机构进行主网上线前的挖矿

6.2、第二阶段2018年7月 - 2018年12月

6.2.1、完成PoW主网，矿工可参与到挖矿工作中去。

6.2.2、完成PoD主网，数据可以委托并且参与BI运算。

6.2.3、完成智能合约，能进行合约交易

6.2.4、完成侧链的测试网络。

6.3、第三阶段2019年

6.3.1、实现PoW网络的分片方案。

第七章、其他事务及法律风险

7.1 法律事务

自媒体链基金会会聘请专项法律顾问及常年法律顾问，主要目的用于建立法律纠纷预防机制、处理 已存在的相关法律问题，协助和配合相关部门。

7.2 免责条款

自媒体链基金会目标转变为非营利组织，链上用户获取的是WeMediaChain的使用权。购买者需明白在法律范围内，WeMediaChain 不做任何明示或暗示的保证，并且 WMC 是“按现状”购买的。此外，购买者应明白 WMC 不会在任何情况下提供退款。 本白皮书会根据项目进展升级版本，同样具备法律效应。

7.3 争议解决条款

当出现争议争议时，有关方面应依据协议通过协商解决，如协商无果，可通过法律解决。

第八章、风险提示

WeMediaChain 项目的所有权属于全体 WMC 代币持有人，并非权益投资项目。WMC 代币及用于换取 WMC 代币的比特币（BTC）、比特现金（BCC）等数字货币均非法定货币，本文档所提供信息不 构成任何投资建议，同时参与者应注意到（包括但不限于）以下风险：

8.1 运营性风险

指的是 WeMediaChain 在认筹资金以及开展业务的过程中违反了当地法律法规，造成无法继续经营的风险。

8.2 流通性风险

指的是在 WMC 没有被市场接纳或没有足够用户使用，业务开展停滞。

8.3 系统性风险

指的是底层技术出现重大问题，导致关键资料被才或丢失； 項目資金出現重大損失，例如：資金被盗，資金虧損，儲備金、大幅貶值等。

8.4 其他不可抗力风险

一切不可预料及不可抗力风险

免责说明

该文档只用于传达信息之途，并不构成本项目买卖的相关意见。以上信息或分析不构成投资 决策。本文档不构成任何投资建议，投资意向或教唆投资。本文档不组成也不理解为提供任 何买卖证券的行为，也不是任何形式上的合约或者承诺。相关意向用户明确了解本项目的风 险，投资者一旦参与投资即表示了解并接受该项目风险，并愿意个人为此承担一切相应结果 或后。运营团队不承担任何参与本项目项目造成的直接或间接的损失。