今天给各位分享区块链加密算法的知识，其中也会对区块链加密算法有哪些进行解释，如果能碰巧解决你现在面临的问题，别忘了关注本站，如果有不同的见解与看法，请积极在评论区留言，现在开始进入正题！

区块链作为新兴技术受到越来越广泛的关注，是一种传统技术在互联网时代下的新的应用，这其中包括分布式数据存储技术、共识机制和密码学等。随着各种区块链研究联盟的创建，相关研究得到了越来越多的资金和人员支持。区块链使用的Hash算法、零知识证明、环签名等密码算法:

Hash算法

哈希算法作为区块链基础技术，Hash函数的本质是将任意长度（有限）的一组数据映射到一组已定义长度的数据流中。若此函数同时满足：

（1）对任意输入的一组数据Hash值的计算都特别简单；

（2）想要找到2个不同的拥有相同Hash值的数据是计算困难的。

满足上述两条性质的Hash函数也被称为加密Hash函数，不引起矛盾的情况下，Hash函数通常指的是加密Hash函数。对于Hash函数，找到使得被称为一次碰撞。当前流行的Hash函数有MD5,SHA1,SHA2,SHA3。

比特币使用的是SHA256，大多区块链系统使用的都是SHA256算法。所以这里先介绍一下SHA256。

1、 SHA256算法步骤

STEP1：附加填充比特。对报文进行填充使报文长度与448模512同余（长度=448 mod512），填充的比特数范围是1到512，填充比特串的最高位为1，其余位为0。

STEP2：附加长度值。将用64-bit表示的初始报文（填充前）的位长度附加在步骤1的结果后（低位字节优先）。

STEP3：初始化缓存。使用一个256-bit的缓存来存放该散列函数的中间及最终结果。

STEP4：处理512-bit（16个字）报文分组序列。该算法使用了六种基本逻辑函数，由64 步迭代运算组成。每步都以256-bit缓存值为输入，然后更新缓存内容。每

步使用一个32-bit 常数值Kt和一个32-bit
Wt。其中Wt是分组之后的报文，t=1,2,...,16 。

STEP5：所有的512-bit分组处理完毕后，对于SHA256算法最后一个分组产生的输出便是256-bit的报文。

作为加密及签名体系的核心算法，哈希函数的安全性事关整个区块链体系的底层安全性。所以关注哈希函数的研究现状是很有必要的。

2、 Hash函的研究现状

2004年我国密码学家王小云在国际密码讨论年会（CRYPTO）上展示了MD5算法的碰撞并给出了第一个实例（Collisions for hash functions MD4, MD5, HAVAL-128 and RIPEMD，rump session of CRYPTO 2004，How to Break MD5 and Other Hash Functions，EuroCrypt 2005）。该攻击复杂度很低，在普通计算机上只需要几秒钟的时间。2005年王小云教授与其同事又提出了对SHA-1算法的碰撞算法，不过计算复杂度为2的63次方，在实际情况下难以实现。

2017年2月23日谷歌安全博客上发布了世界上第一例公开的SHA-1哈希碰撞实例，在经过两年的联合研究和花费了巨大的计算机时间之后，研究人员在他们的研究网站SHAttered上给出了两个内容不同，但是具有相同SHA-1消息摘要的PDF文件，这就意味着在理论研究长期以来警示SHA-1算法存在风险之后，SHA-1算法的实际攻击案例也浮出水面，同时也标志着SHA-1算法终于走向了生命的末期。

NIST于2007年正式宣布在全球范围内征集新的下一代密码Hash算法，举行SHA-3竞赛。新的Hash算法将被称为SHA-3，并且作为新的安全Hash标准，增强现有的FIPS 180-2标准。算法提交已于2008年10月结束，NIST 分别于2009年和2010年举行2轮会议，通过2轮的筛选选出进入最终轮的算法，最后将在2012年公布获胜算法。公开竞赛的整个进程仿照高级加密标准AES
的征集过程。2012年10月2日，Keccak被选为NIST竞赛的胜利者，成为SHA-3。

Keccak算法是SHA-3的候选人在2008年10月提交。Keccak采用了创新的的“海绵引擎”散列消息文本。它设计简单，方便硬件实现。Keccak已可以抵御最小的复杂度为2n的攻击，其中N为散列的大小。它具有广泛的安全边际。目前为止，第三方密码分析已经显示出Keccak没有严重的弱点。

KangarooTwelve算法是最近提出的Keccak变种，其计算轮次已经减少到了12，但与原算法比起来，其功能没有调整。

零知识证明

在密码学中零知识证明（zero-knowledge proof, ZKP）是一种一方用于向另一方证明自己知晓某个消息x，而不透露其他任何和x上述文章内容就是的内容的策略，其中前者称为证明者（Prover），后者称为验证者（Verifier）。设想一种场景，在一个系统中， 所有用户都拥有各自全部文件的备份，
并利用各自的私钥进行加密后在系统内公开。 假设在某个时刻，用户Alice希望提供给用户Bob她的一部分文件，这时候出现的问题是Alice如何让Bob相信她确实发送了正确的文件。一个简单地处理办法是Alice将自己的私钥发给Bob，而这正是Alice不希望选择的策略，因为这样 Bob可以轻易地获取到Alice的全部文件内容。零知识证明便是可以用于解决上述问题的一种方案。零知识证明主要基于复杂度理论，并且在密码学中有广泛的理论延伸。在复杂度理论中，我们主要讨论哪些语言可以进行零知识证明应用，而在密码学中，我们主要讨论如何构造各种类型的零知识证明方案，并使得其足够优秀和高效。

环签名群签名

1、群签名

在一个群签名方案中，一个群体中的任意一个成员可以以匿名的方式代表整个群体对消息进行签名。与其他数字签名一样，群签名是可以公开验证的，且可以只用单个群公钥来验证。群签名一般流程：

（1）初始化，群管理者建立群资源，生成对应的群公钥（Group Public Key）和群私钥（Group Private
Key）群公钥对整个系统中的所有用户公开，比如群成员、验证者等。

（2）成员加入，在用户加入群的时候，群管理者颁发群证书（Group Certificate）给群成员。

（3）签名，群成员利用获得的群证书签署文件，生成群签名。

（4）验证，同时验证者利用群公钥仅可以验证所得群签名的正确性，但不能确定群中的正式签署者。

（5）公开，群管理者利用群私钥可以对群用户生成的群签名进行追踪，并暴露签署者身份。

2、环签名

2001年，Rivest, shamir和Tauman三位密码学家首次提出了环签名。是一种简化的群签名，只有环成员没有管理者，不需要环成员间的合作。环签名方案中签名者首先选定一个临时的签名者集合,集合中包括签名者。然后签名者利用自己的私钥和签名集合中其他人的公钥就可以独立的产生签名,而无需他人的帮助。签名者集合中的成员可能并不知道自己被包含在其中。

环签名方案由以下几部分构成：

（1）密钥生成。为环中每个成员产生一个密钥对(公钥PKi,私钥SKi)。

（2）签名。签名者用自己的私钥和任意n个环成员(包括自己)的公钥为消息m生成签名a。

（3）签名验证。验证者根据环签名和消息m,验证签名是否为环中成员所签,如果有效就接收,否则丢弃。

环签名满足的性质：

（1）无条件匿名性:攻击者无法确定签名是由环中哪个成员生成,即使在获得环成员私钥的情况下,概率也不超过1/n。

（2）正确性:签名必需能被所有其他人验证。

（3）不可伪造性:环中其他成员不能伪造真实签名者签名,外部攻击者即使在获得某个有效环签名的基础上,也不能为消息m伪造一个签名。

3、环签名和群签名的比较

（1）匿名性。都是一种个体代表群体签名的体制，验证者能验证签名为群体中某个成员所签，但并不能知道为哪个成员，以达到签名者匿名的作用。

（2）可追踪性。群签名中，群管理员的存在保证了签名的可追踪性。群管理员可以撤销签名，揭露真正的签名者。环签名本身无法揭示签名者,除非签名者本身想暴露或者在签名中添加额外的信息。提出了一个可验证的环签名方案,方案中真实签名者希望验证者知道自己的身份,此时真实签名者可以通过透露自己掌握的秘密信息来证实自己的身份。

（3）管理系统。群签名由群管理员管理，环签名不需要管理，签名者只有选择一个可能的签名者集合，获得其公钥，然后公布这个集合即可，所有成员平等。

链乔教育在线旗下学硕创新区块链技术工作站是中国教育部学校规划建设发展中心开展的“智慧学习工场2020-学硕创新工作站”唯一获准的“区块链技术专业”试点工作站。专业站立足为学生提供多样化成长路径，推进专业学位研究生产学研结合培养模式改革，构建应用型、复合型人才培养体系。

????先放一张以太坊的架构图：

？??在学习的过程中主要是采用单个模块了学习了解的，包括P2P,密码学，网络，协议等。直接开始总结：

？????????????秘钥分配问题也就是秘钥的传输问题，如果对称秘钥，那么只能在线下进行秘钥的交换。如果在线上传输秘钥，那就有可能被拦截。所以采用非对称加密，两把钥匙，一把私钥自留，一把公钥公开。公钥可以在网上传输。不用线下交易。保证数据的安全性。

????????如上图，A节点发送数据到B节点，此时采用公钥加密。A节点从自己的公钥中获取到B节点的公钥对明文数据加密，得到密文发送给B节点。而B节点采用自己的私钥解密。

？????????2、无法解决消息篡改。

????如上图，A节点采用B的公钥进行加密，然后将密文传输给B节点。B节点拿A节点的公钥将密文解密。

？????????1、由于A的公钥是公开的，一旦网上黑客拦截消息，密文形同虚设。说白了，这种加密方式，只要拦截消息，就都能解开。

？????????2、同样存在无法确定消息来源的问题，和消息篡改的问题。

????????如上图，A节点在发送数据前，先用B的公钥加密，得到密文1，再用A的私钥对密文1加密得到密文2。而B节点得到密文后，先用A的公钥解密，得到密文1，之后用B的私钥解密得到明文。

？????????1、当网络上拦截到数据密文2时， 由于A的公钥是公开的，故可以用A的公钥对密文2解密，就得到了密文1。所以这样看起来是双重加密，其实最后一层的私钥签名是无效的。一般来讲，我们都希望签名是签在最原始的数据上。如果签名放在后面，由于公钥是公开的，签名就缺乏安全性。

？????????2、存在性能问题，非对称加密本身效率就很低下，还进行了两次加密过

程。

????????如上图，A节点先用A的私钥加密，之后用B的公钥加密。B节点收到消息后，先采用B的私钥解密，然后再利用A的公钥解密。

????????1、当密文数据2被黑客拦截后，由于密文2只能采用B的私钥解密，而B的私钥只有B节点有，其他人无法机密。故安全性最高。

????????2、当B节点解密得到密文1后， 只能采用A的公钥来解密。而只有经过A的私钥加密的数据才能用A的公钥解密成功，A的私钥只有A节点有，所以可以确定数据是由A节点传输过来的。

????????经两次非对称加密，性能问题比较严重。

????????基于以上篡改数据的问题，我们引入了消息认证。经过消息认证后的加密流程如下：

????????当A节点发送消息前，先对明文数据做一次散列计算。得到一个摘要, 之后将照耀与原始数据同时发送给B节点。当B节点接收到消息后，对消息解密。解析出其中的散列摘要和原始数据，然后再对原始数据进行一次同样的散列计算得到摘要1, 比较摘要与摘要1。如果相同则未被篡改，如果不同则表示已经被篡改。

????????在传输过程中，密文2只要被篡改，最后导致的hash与hash1就会产生不同。

????????无法解决签名问题，也就是双方相互攻击。A对于自己发送的消息始终不承认。比如A对B发送了一条错误消息，导致B有损失。但A抵赖不是自己发送的。

????????在(三)的过程中，没有办法解决交互双方相互攻击。什么意思呢？
有可能是因为A发送的消息，对A节点不利，后来A就抵赖这消息不是它发送的。

????????为了解决这个问题，故引入了签名。这里我们将(二)-4中的加密方式，与消息签名合并设计在一起。

????在上图中，我们利用A节点的私钥对其发送的摘要信息进行签名，然后将签名+原文，再利用B的公钥进行加密。而B得到密文后，先用B的私钥解密，然后 对摘要再用A的公钥解密，只有比较两次摘要的内容是否相同。这既避免了防篡改问题，有规避了双方攻击问题。因为A对信息进行了签名，故是无法抵赖的。

????????为了解决非对称加密数据时的性能问题，故往往采用混合加密。这里就需要引入对称加密，如下图:

????????在对数据加密时，我们采用了双方共享的对称秘钥来加密。而对称秘钥尽量不要在网络上传输，以免丢失。这里的共享对称秘钥是根据自己的私钥和对方的公钥计算出的，然后适用对称秘钥对数据加密。而对方接收到数据时，也计算出对称秘钥然后对密文解密。

????????以上这种对称秘钥是不安全的，因为A的私钥和B的公钥一般短期内固定，所以共享对称秘钥也是固定不变的。为了增强安全性，最好的方式是每次交互都生成一个临时的共享对称秘钥。那么如何才能在每次交互过程中生成一个随机的对称秘钥，且不需要传输呢？

????????那么如何生成随机的共享秘钥进行加密呢？

????????对于发送方A节点，在每次发送时，都生成一个临时非对称秘钥对，然后根据B节点的公钥 和 临时的非对称私钥 可以计算出一个对称秘钥(KA算法-Key Agreement)。然后利用该对称秘钥对数据进行加密，针对共享秘钥这里的流程如下：

????????对于B节点，当接收到传输过来的数据时，解析出其中A节点的随机公钥，之后利用A节点的随机公钥 与 B节点自身的私钥
计算出对称秘钥(KA算法)。之后利用对称秘钥机密数据。

????????对于以上加密方式，其实仍然存在很多问题，比如如何避免重放攻击(在消息中加入 Nonce)，再比如彩虹表(参考 KDF机制解决
)之类的问题。由于时间及能力有限，故暂时忽略。

????????那么究竟应该采用何种加密呢？

????????主要还是基于要传输的数据的安全等级来考量。不重要的数据其实做好认证和签名就可以，但是很重要的数据就需要采用安全等级比较高的加密方案了。

????????密码套件 是一个网络协议的概念。其中主要包括身份认证、加密、消息认证(MAC)、秘钥交换的算法组成。

????????在整个网络的传输过程中，根据密码套件主要分如下几大类算法：

????????秘钥交换算法：比如ECDHE、RSA。主要用于客户端和服务端握手时如何进行身份验证。

????????消息认证算法：比如SHA1、SHA2、SHA3。主要用于消息摘要。

????????批量加密算法：比如AES, 主要用于加密信息流。

????????伪随机数算法：例如TLS 1.2的伪随机函数使用MAC算法的散列函数来创建一个 主密钥 ——连接双方共享的一个48字节的私钥。主密钥在创建会话密钥（例如创建MAC）时作为一个熵来源。

????????在网络中，一次消息的传输一般需要在如下4个阶段分别进行加密，才能保证消息安全、可靠的传输。

????????握手/网络协商阶段：

????????在双方进行握手阶段，需要进行链接的协商。主要的加密算法包括RSA、DH、ECDH等

? ????? 身份认证阶段：

????????身份认证阶段，需要确定发送的消息的来源来源。主要采用的加密方式包括RSA、DSA、ECDSA(ECC加密，DSA签名)等。

????????消息加密阶段：

????????消息加密指对发送的信息流进行加密。主要采用的加密方式包括DES、RC4、AES等。

????????消息身份认证阶段/防篡改阶段：

????????主要是保证消息在传输过程中确保没有被篡改过。主要的加密方式包括MD5、SHA1、SHA2、SHA3等。

? ?????? ECC ：Elliptic Curves Cryptography，椭圆曲线密码编码学。是一种根据椭圆上点倍积生成公钥、私钥的算法。用于生成公私秘钥。

? ?????? ECDSA ：用于数字签名,是一种数字签名算法。一种有效的数字签名使接收者有理由相信消息是由已知的发送者创建的，从而发送者不能否认已经发送了消息(身份验证和不可否认)，并且消息在运输过程中没有改变。ECDSA签名算法是EC

C与DSA的结合，整个签名过程与DSA类似，所不一样的是签名中采取的算法为ECC，最后签名出来的值也是分为r,s。 主要用于身份认证阶段 。

？?????? ECDH ：也是基于ECC算法的霍夫曼树秘钥，通过ECDH，双方可以在不共享任何秘密的前提下协商出一个共享秘密，并且是这种共享秘钥是为当前的通信暂时性的随机生成的，通信一旦中断秘钥就消失。 主要用于握手磋商阶段。

？?????? ECIES： 是一种集成加密方案,也可称为一种混合加密方案，它提供了对所选择的明文和选择的密码文本攻击的语义安全性。ECIES可以使用不同类型的函数：秘钥协商函数(KA)，秘钥推导函数(KDF)，对称加密方案(ENC)，哈希函数(HASH), H-MAC函数(MAC)。

？?????? ECC
是椭圆加密算法，主要讲述了按照公私钥怎么在椭圆上产生，并且不可逆。
ECDSA 则主要是采用ECC算法怎么来做签名， ECDH 则是采用ECC算法怎么生成对称秘钥。以上三者都是对ECC加密算法的应用。而现实场景中，我们往往会采用混合加密(对称加密，非对称加密结合使用，签名技术等一起使用)。 ECIES 就是底层利用ECC算法提供的一套集成(混合)加密方案。其中包括了非对称加密，对称加密和签名的功能。

????????ECC 是?Elliptic Curve Cryptography的简称。那么什么是椭圆加密曲线呢？Wolfram MathWorld给出了很标准的定义： 一条椭圆曲线就是一组被? ?定义的且满足? 的点集。 ？

这个先订条件是为了保证曲线不包含奇点。

所以，随着曲线参数a和b的不断变化，曲线也呈现出了不同的形状。比如:

????????所有的非对称加密的基本原理基本都是基于一个公式 K = k*G。其中K代表公钥，k代表私钥，G代表某一个选取的基点。非对称加密的算法就是要保证 该公式? 不可进行逆运算(也就是说G/K是无法计算的)。

????????ECC是如何计算出公私钥呢？这里我按照我自己的理解来描述。

？?????? 我理解，ECC的核心思想就是：选择曲线上的一个基点G，之后随机在ECC曲线上取一个点k(作为私钥)，然后根据k*G计算出我们的公钥K。并且保证公钥K也要在曲线上。

????????那么k*G怎么计算呢？如何计算k*G才能保证最后的结果不可逆呢？这就

是ECC算法要解决的。

????????首先，我们先随便选择一条ECC曲线，a = -3, b = 7 得到如下曲线：

????????在这个曲线上，我随机选取两个点，这两个点的乘法怎么算呢？我们可以简化下问题，乘法是都可以用加法表示的，比如2*2 = 2+2，3*5 = 5+5+5。 那么我们只要能在曲线上计算出加法，理论上就能算乘法。所以，只要能在这个曲线上进行加法计算，理论上就可以来计算乘法，理论上也就可以计算k*G这种表达式的值。

????????曲线上两点的加法又怎么算呢？这里ECC为了保证不可逆性，在曲线上自定义了加法体系。

????????现实中，1+1=2，2+2=4，但在ECC算法里，我们理解的这种加法体系是不可能。故需要自定义一套适用于该曲线的加法体系。

？?????? ECC定义，在图形中随机找一条直线，与ECC曲线相交于三个点(也有可能是两个点)，这三点分别是P、Q、R。

？?????? 那么P+Q+R = 0。其中0
不是坐标轴上的0点，而是ECC中的无穷远点。也就是说定义了无穷远点为0点。

????????同样，我们就能得出 P+Q = -R。 由于R 与-
R是关于X轴对称的，所以我们就能在曲线上找到其坐标。

????????P+R+Q = 0, 故P+R = -Q , 如上图。

以上就描述了ECC曲线的世界里是如何进行加法运算的。

????????从上图可看出，直线与曲线只有两个交点，也就是说
直线是曲线的切线。此时P,R 重合了。

????????也就是P = R, 根据上述ECC的加法体系，P+R+Q = 0, 就可以得出
P+R+Q = 2P+Q = 2R+Q=0

????????于是乎得到 2*P = -Q (是不是与我们非对称算法的公式 K = k*G
越来越近了)。

????????于是我们得出一个结论，可以算乘法，不过只有在切点的时候才能算乘法

，而且只能算2的乘法。

????????假若 2 可以变成任意个数进行想乘，那么就能代表在ECC曲线里可以进行乘法运算，那么ECC算法就能满足非对称加密算法的要求了。

????????那么我们是不是可以随机任何一个数的乘法都可以算呢？
答案是肯定的。 也就是点倍积 计算方式。

????????选一个随机数 k, 那么k * P等于多少呢？

????????我们知道在计算机的世界里，所有的都是二进制的，ECC既然能算2的乘法，那么我们可以将随机数k描 述成二进制然后计算。假若k = 151 = 10010111

????????由于2*P = -Q 所以 这样就计算出了k*P。 这就是点倍积算法 。所以在ECC的曲线体系下是可以来计算乘法，那么以为这非对称加密的方式是可行的。

????????至于为什么这样计算
是不可逆的。这需要大量的推演，我也不了解。但是我觉得可以这样理解：

????????我们的手表上，一般都有时间刻度。现在如果把1990年01月01日0点0分0秒作为起始点，如果告诉你至起始点为止时间流逝了 整1年，那么我们是可以计算出现在的时间的，也就是能在手表上将时分秒指针应该指向00：00：00。但是反过来，我说现在手表上的时分秒指针指向了00：00：00,你能告诉我至起始点算过了有几年了么？

????????ECDSA签名算法和其他DSA、RSA基本相似，都是采用私钥签名，公钥验证。只不过算法体系采用的是ECC的算法。交互的双方要采用同一套参数体系。签名原理如下：

????????在曲线上选取一个无穷远点为基点?G = (x,y)。随机在曲线上取一点k 作为私钥， K = k*G 计算出公钥。

? ?????? 签名过程：

????????生成随机数R, 计算出RG.

????????根据随机数R,消息M的HASH值H,以及私钥k, 计算出签名S = (H+kx)/R.

????????将消息M,RG,S发送给接收方。

? ?????? 签名验证过程：

????????接收到消息M, RG,S

????????根据消息计算出HASH值H

????????根据发送方的公钥K,计算 HG/S + xK/S, 将计算的结果与RG比较。如果相等则验证成功。

? ?????? 公式推论：

????????HG/S + xK/S = HG/S + x(kG)/S = (H+xk)/GS = RG

????????在介绍原理前，说明一下ECC是满足结合律和交换律的，也就是说A+B+C = A+C+B = (A+C)+B。

????????这里举一个WIKI上的例子说明如何生成共享秘钥，也可以参考 ?Alice And Bob ?的例子。

????????Alice 与Bob 要进行通信，双方前提都是基于 同一参数体系的ECC生成的公钥和私钥。所以有ECC有共同的基点G。

? ?????? 生成秘钥阶段：

????????Alice 采用公钥算法 KA = ka * G ，生成了公钥KA和私钥ka, 并公开公钥KA。

????????Bob 采用公钥算法 KB = kb * G ，生成了公钥KB和私钥 kb, 并公开公钥KB。

? ?????? 计算ECDH阶段：

????????Alice 利用计算公式 Q = ka * KB ?计算出一个秘钥Q。

????????Bob 利用计算公式 Q’ = kb * KA 计算出一个秘钥Q’ 。

? ?????? 共享秘钥验证：

????????Q = ka? KB = ka * kb * G = ka * G * kb = KA * kb = kb * KA = Q’

????????故 双方分别计算出的共享秘钥不需要进行公开就可采用Q进行加密。我们将Q称为共享秘钥。

????????在以太坊中，采用的ECIEC的加密套件中的其他内容：

????????1、其中HASH算法采用的是最安全的SHA3算法 Keccak 。

????????2、签名算法采用的是 ECDSA

????????3、认证方式采用的是? H-MAC

????????4、ECC的参数体系采用了secp256k1,? 其他参数体系 参考这里

????????H-MAC 全程叫做 Hash-based Message Authentication Code.
其模型如下：

在 以太坊 的
UDP通信时(RPC通信加密方式不同)，则采用了以上的实现方式，并扩展化了。

首先，以太坊的UDP通信的结构如下：

????????其中，sig是 经过
私钥加密的签名信息。mac是可以理解为整个消息的摘要，
ptype是消息的事件类型，data则是经过RLP编码后的传输数据。

????????其UDP的整个的加密，认证，签名模型如下：

区块链本质上是加密算法，基于哈希值256位算法原理，实现信息安全；现代信息的应用将越来越趋于全球化以及全民化，对于信息的安全除了防篡改、抗抵赖、可信等基础需求之外，更需要加强隐私方面的保护，区块链技术是因为现代密码学发展才产生的，现今应用的密码学是20年前的的密码学成果，因此要将区块链技术应用于更多参与场景，特别是应用于互联网经济等方面，现有的加密技术是否满足需求还需要更多的验证，需要更深入的整合密码学前沿技术，不断创新。

隐私保护手段可以分为三类：

一是对交易信息的隐私保护，对交易的发送者、交易接受者以及交易金额的隐私保护，有混币、环签名和机密交易等。

二是对智能合约的隐私保护，针对合约数据的保护方案，包含零知识证明、多方安全计算、同态加密等。

三是对链上数据的隐私保护，主要有账本隔离、私有数据和数据加密授权访问等解决方案。

拓展资料：

一、区块链加密算法隔离身份信息与交易数据

1、区块链上的交易数据，包括交易地址、金额、交易时间等，都公开透明可查询。但是，交易地址对应的所用户身份，是匿名的。通过区块链加密算法，实现用户身份和用户交易数据的分离。在数据保存到区块链上之前，可以将用户的身份信息进行哈希计算，得到的哈希值作为该用户的唯一标识，链上保存用户的哈希值而非真实身份数据信息，用户的交易数据和哈希值进行捆绑，而不是和用户身份信息进行捆绑。

2、由此，用户产生的数据是真实的，而使用这些数据做研究、分析时，由于区块链的不可逆性，所有人不能通过哈希值还原注册用户的姓名、电话、邮箱等隐私数据，起到了保护隐私的作用。

二、区块链“加密存储+分布式存储”

加密存储，意味着访问数据必须提供私钥，相比于普通密码，私钥的安全性更高，几乎无法被暴力破解。分布式存储，去中心化的特性在一定程度上降低了数据全部被泄漏的风险，而中心化的数据库存储，一旦数据库被黑客攻击入侵，数据很容易被全部盗走。通过“加密存储+分布式存储”能够更好地保护用户的数据隐私。

三、区块链共识机制预防个体风险

共识机制是区块链节点就区块信息达成全网一致共识的机制，可以保障最新区块被准确添加至区块链、节点存储的区块链信息一致不分叉，可以抵御恶意攻击。区块链的价值之一在于对数据的共识治理，即所有用户对于上链的数据拥有平等的管理权限，因此首先从操作上杜绝了个体犯错的风险。通过区块链的全网共识解决数据去中心化，并且可以利用零知识证明解决验证的问题，实现在公开的去中心化系统中使用用户隐私数据的场景，在满足互联网平台需求的同时，也使部分数据仍然只掌握在用户手中。

四、区块链零知识证明

零知识证明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，即证明者既能充分证明自己是某种权益的合法拥有者，又不把上述文章内容就是的信息泄漏出去，即给外界的“知识”为“零”。应用零知识证明技术，可以在密文情况下实现数据的关联关系验证，在保障数据隐私的同时实现数据共享。

只要你仔细阅读了上述，那么你就已经了解了区块链加密算法有哪些的相关知识，如果屏幕面前的你还有什么对区块链加密算法好的建议和想法，欢迎各位再下面评论区评论出来，我们将及时回复。