ECDSA算法介绍 一、ECDSA概述 椭圆曲线数字签名算法（ECDSA）是使用椭圆曲线密码（ECC）对数字签名算法（DSA）的模拟。与普通的离散对数问题（DLP）和大数分解问题（IFP）不同，椭圆曲线离散对数问题没有亚指数时间的解决方法。因此椭圆曲线密码的单位比特强度要高于其他公钥体制。

数字签名算法（DSA）在联邦信息处理标准FIPS中有详细论述，称为数字签名标准。它的安全性基于素域上的离散对数问题。可以看作是椭圆曲线对先前离散对数问题（DLP）的密码系统的模拟，只是群元素由素域中的元素数换为有限域上的椭圆曲线上的点。椭圆曲线离散对数问题远难于离散对数问题，单位比特强度要远高于传统的离散对数系统。因此在使用较短的密钥的情况下，ECC可以达到于DL系统相同的安全级别。这带来的好处就是计算参数更小，密钥更短，运算速度更快，签名也更加短小。

二、ECDSA原理 ECDSA是ECC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。 签名过程如下： 1、选择一条椭圆曲线Ep(a,b)，和基点G； 2、选择私有密钥k（k<n，n为G的阶），利用基点G计算公开密钥K=kG； 3、产生一个随机整数r（r<n），计算点R=rG； 4、将原数据和点R的坐标值x,y作为参数，计算SHA1做为hash，即Hash=SHA1(原数据,x,y)； 5、计算s≡r - Hash \* k (mod n) 6、r和s做为签名值，如果r和s其中一个为0，重新从第3步开始执行 验证过程如下： 1、接受方在收到消息(m)和签名值(r,s)后，进行以下运算 2、计算：sG+H(m)P=(x1,y1), r1≡ x1 mod p。 3、验证等式：r1 ≡ r mod p。 4、如果等式成立，接受签名，否则签名无效。

三、ECDSA的实践 实施ECDSA时出现的一些问题。在曲线和密钥生成或签名生成和验证过程中可能会出现一些漏洞。我们只调查与椭圆曲线的选择有关的问题。在实施过程中出现的一般问题，例如不检查一个点是否是无穷大的点，在这里不涉及。 第一个漏洞可能是操纵∶建议的安全曲线可能有一个后门不安全因素。比特币和以太坊使用一个固定的曲线--secp256k1--并且只生成私钥和公钥。根据Safecurves,椭圆曲线secp256k1可以被认为有些“僵硬"，这意味着几乎所有的参数对公众是透明的，因此可以假设不是为了弱点而生成的。 1。梯子。椭圆曲线E上的一个点P的标量乘法在ECDSA中经常使用--例如用于公钥的生成。所谓的Mont- gomery梯子是一种快速而简单的算法，可以在恒定时间内完成这一计算。为了实现这个阶梯，椭圆曲线必须是一个特定的形状。secp256k1曲线不允许使用蒙哥马利阶梯。作为一个序列，除了简单和高效之外，secp256k1可能会因为某些计算的时间不恒定而泄露信息，从而导致侧信道攻击。这已经导致了成功的密钥提取，并反映在libsecp256ki实现套件中（见[GPP+16])。笔者不知道这个实现是否快速和简单。Brier-Joye梯子也可以应用，但会使运算速度降低很多。最后，Safecurves推荐蒙哥马利的单坐标梯子，因为它更容易实现对我们接下来讨论的攻击的保护。 3.2.扭曲的安全性。正如[BHH+14]中指出的，无效曲线攻击可能导致secp256k1的严重漏洞。因此，攻击者使用一个类似的椭圆曲线--原始曲线的扭曲--而只是假装使用原始曲线。如果这个扭曲在第2.2节的意义上是不安全的，而且实现者没有检查攻击者建议的点是否位于原始曲线上，那么他就有很大的机会在一些查询之后提取私钥。现在，secp256k1的标准二次扭曲也是一条安全的曲线(群的cardinality有220位素数﹔见[BL13])，但更大的自动变形群又导致了四个扭曲。它们的最大素数除数是133、i88、135和16l，但也有较小的素数因子，使得攻击可能更加可行(见[BHH+14]，第4页)。一条不是扭曲安全的曲线需要在签名和验证过程中检查这些点是否真的在曲线上。这一点是可以做到的，但会使实现的效率和安全性降低。关于这种攻击的更多细节，我们可以参考[FLRVo8]，在那里我们可以找到一个关于secp256k1的计算实例。 3.3.完整性和不可分性。椭圆曲线上的加法和标量乘法的公式对于曲线上的某些特定点可能会略有变化。一个没有照顾到这些例外情况的实现会产生错误。完整性是指曲线没有例外情况。曲线secp256k1上的标量乘法是不完整的。 椭圆曲线上的点的表示通常可以从随机产生的字符串中区分出来。为了掩盖椭圆曲线上的点的外观，有一些可用的策略（参见[BL13])。这些构造不适用于secp256k1。 3.4.多个ECDLP。在比特币中，任何用户的公钥都是由secp256k1，他们的私钥和基点P产生的。情况看起来如下。假设我们有L个用户，他们的公钥Qi = ai PE(Fp), 1 iL。如果有人想找到他们的私钥，她必须解决以下离散对数问题。 Qi = ai P(1 ≤i≤ L)。 椭圆曲线上的点的表示通常可以从随机产生的字符串中区分出来。为了掩盖椭圆曲线上的点的外观，有一些可用的策略（参见[BL13])。这些构造不适用于secp256k1。 3.4.多个ECDLP。在比特币中，任何用户的公钥都是由secp256k1，他们的私钥和基点P产生的。情况看起来如下。假设我们有L个用户，他们的公钥Qi = ai PE(Fp), 1 iL。如果有人想找到他们的私钥，她必须解决以下离散对数问题。l Qi = ai P(1 ≤i≤ L)。 我们必须要解决L倍的离散对数问题吗?或者可以利用ECDLPs发生在同一基点P的椭圆曲线上的事实?到目前为止，还没有已知的算法能够更快地找到一个实例的解决方案(见[GG16]，第4页）。但使用Pollard's rho方法的扩展版本，一旦找到一个，就能逐步加速找到其他的解。例如，如果L<r八，r是我们组的大小，Kuhn和Struik表明，我们平均需要2rL组操作。在[HMVo4]第164页提出的Pollard's rho的扩展算法，在找到第一个实例后，第二个ECDLP的运行时间降低了50%，第三个降低了37%，以此类推。所以我们必须确保找到一个实例是不可行的，而secp256k1做到了。

参考文献： 1.https://weibo.com/ttarticle/p/show?id=2309404720225936605782#\_loginLayer\_1659241375037

2.Hartwig Mayer.ECDSA Security in Bitcoin and Ethereum: a Research Survey