ECDSA 算法介绍报告

202100460164 刘莹

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。

ECDSA 处理过程：

1.参与数字签名的所有通信方都使用相同的全局参数，用于定义椭圆曲线以及曲线上的基点

2.签名者首先生成一对公、私钥。对于私钥，选择一个随机数或者伪随机数作为私钥，利用随机数和基点算出另一点，作为公钥。

3.对消息计算 Hash 值，用私钥、全局参数和 Hash 值生成签名

4.验证者用签名者的公钥、全局参数等验证。

全局参数：

q　一个素数

a，b　Z_q 上的整数，通过等式 $y^2 = x^3 + ax + b$ 定义椭圆曲线

G　满足椭圆曲线等式的基点，表示为 $G = (x_g, y_g)$

n　点 G 的阶，即 n 是满足 $nG = O$ 的最小正整数

密钥生成：

每个签名者都要生成一对公、私钥，假设是 Bob。

(1)选择随机整数 d，$d \in [1, n-1]$。

(2)计算 $Q = dG$。得到一个曲线 $E_q(a, b)$ 上的解点。

(3)Bob 的公钥是 Q，私钥是 d。

这里是定义在Z_q

上的椭圆曲线，椭圆曲线上的乘法运算就是多个点的累加运算，最后的结果还是

椭圆曲线上的点，有了公钥之后，Bob 对消息 m 生成 320 字节的数字签名：

(1)选择随机或伪随机整数 k，$k \in [1, n-1]$。

(2)计算曲线的解点 $P = (x, y) = kG$，以及 $r = x \bmod n$。如果 $r = 0$ 则跳至步骤(1)。

(3)计算 $t = k^{-1} \bmod n$。

(4)计算 $e = H(m)$，这里 H 是 Hash 函数 SHA-1，产生 160 位的 Hash 值。

(5)计算 $s = k^{-1}(e + dr) \bmod n$。如果 $s = O$，则跳至步骤(1)。

(6)消息 m 的签名是 (r, s) 对。

第 2 步确保最后算出的公钥（椭圆曲线上的点）是落在曲线上。第 5 步，如果为

O 点也是不符合的，所以也要重新生成。

Alice 在获得 Bob 的公钥和全局参数后，即可校验签名。

(1)检验 r 和 s 是否为 1 到 $n-1$ 之内的整数。

(2)使用 SHA-1，计算 160 位的 Hash 值 $e = H(m)$。

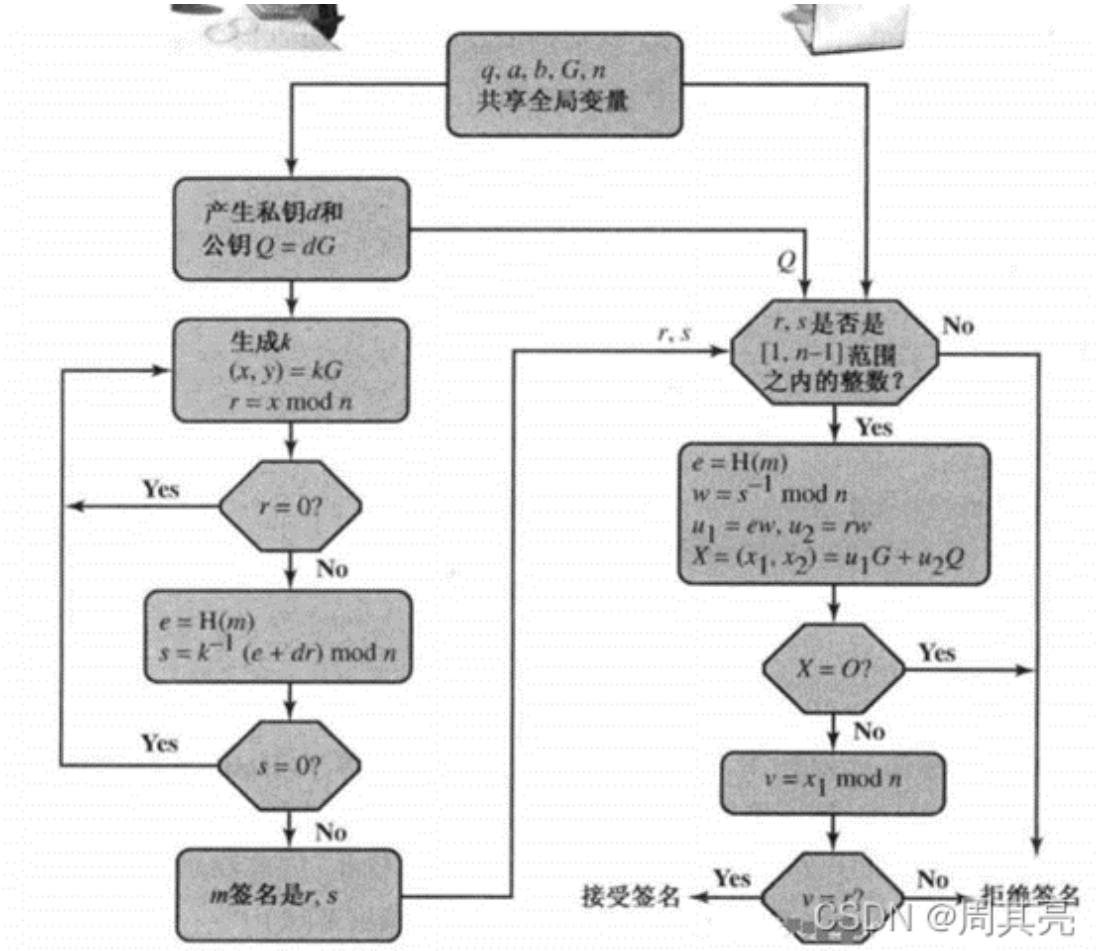
(3)计算 $w = s^{-1} \bmod n$。

(4)计算 $u_1 = ew$ 和 $u_2 = rw$。

(5)计算解点 $X = (x_1, y_1) = u_1 G + u_2 Q$。

(6)如果 $X = O$，拒绝该签名；否则计算 $v = x_1 \bmod n$。

(7)当且仅当 $v = r$ 时，接受 Bob 的签名。



该过程有效性证明如下，如果 Alice 收到的消息确实是 Bob 签署的，那么

s = (e+dr)k^-1 mod n

于是 k= (e+dr)s^-1 mod n

= (e s^-1 + dr s^-1) mod n

= (we +wdr) mod n

=(u1 + u2d) mod n

现在考虑 u1G + u2Q = u1G + u2dG = (u1 + u2d)G = KG

在验证过程的步骤 6 中，有 v = x1 mod n，这里解点 X =(x1,y1) = u1G + u2Q。

因为

R = x mod n 且 x 是解点 kG 的 x 坐标，又因为 我们已知 u1G + u2Q = KG，

所以可得到 v = r。

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。

3.1.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；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。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。I 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。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