SM2 签名算法误用场景的 PoC 验证与推导文档

一、场景概述

20250713-wen-sm2-public.pdf 文档中明确指出 SM2、ECDSA 等椭圆曲线签名算法在实际应用中存在多种因参数误用导致的安全风险，主要包括随机数泄露、随机数重复使用、跨算法参数复用等场景，这些误用可能导致私钥被推导泄露。针对这些场景分别设计 PoC 验证方案，并进行详细推导。

二、场景 1：泄露随机数 k 导致私钥泄露

1、误用描述

签名过程中使用的随机数 k 被泄露，攻击者可通过签名(r,s)和 k 推导出私钥d_A。

2、PoC 验证步骤

参数初始化

采用 SM2 标准参数：椭圆曲线$y^2 = x^3 + ax + b$，阶 n，基点$G = (x_G, y_G)$。

生成用户私钥$d_A \in [1, n-1]$，公钥$P_A = d_A \cdot G$。

预计算$Z_A = H_{256}(ENTL_A \| ID_A \| a \| b \| x_G \| y_G \| x_A \| y_A)$。

签名生成

消息 M，计算$\overline{M} = Z_A \| M$，$e = H_v(\overline{M})$。

随机选$k \in [1, n-1]$，计算$kG = (x_1, y_1)$，$r = (e + x_1) \bmod n$（确保$r \neq 0$ 且$r + k \neq n$）。

计算$s = ((1 + d_A)^{-1} \cdot (k - r \cdot d_A)) \bmod n$，签名为(r,s)。

模拟 k 泄露

攻击者获取 k、(r,s)、e。

私钥推导

根据公式$d_A = (s + r)^{-1} \cdot (k - s) \bmod n$计算$d_A$。

验证：计算$P_A' = d_A \cdot G$，若$P_A' = P_A$则推导成功。

3、推导过程

由签名公式$s = ((1 + d_A)^{-1} \cdot (k - r \cdot d_A)) \bmod n$，

两边同乘$(1+d_A)$得：$s(1+d_A)=k-r\cdot d_A \bmod n$
整理得：$d_A(s+r)=k-s \bmod n$
因此：$d_A=(k-s)\cdot(s+r)^{-1} \bmod n$。

三、场景 2：同一用户重复使用随机数 k 导致私钥泄露

1、误用描述

同一用户对不同消息重复使用随机数 k，攻击者通过两个签名(r_1,s_1)和(r_2,s_2)可推导私钥d_A。

2、PoC 验证步骤

参数初始化

同场景 1，用户私钥d_A，公钥P_A。

重复 k 签名

消息M_1：计算$e_1=H_v(Z_A\|M_1)$，$kG=(x,y)$，$r_1=(e_1+x) \bmod n$，$s_1=((1+d_A)^{-1}\cdot(k-r_1\cdot d_A)) \bmod n$。

消息M_2：复用 k，计算$e_2=H_v(Z_A\|M_2)$，$r_2=(e_2+x) \bmod n$，$s_2=((1+d_A)^{-1}\cdot(k-r_2\cdot d_A)) \bmod n$。

私钥推导

根据公式$d_A=(s_2-s_1)\cdot(s_1-s_2+r_1-r_2)^{-1} \bmod n$计算$d_A$。

验证：同场景 1。

3、推导过程

对两个签名分别列方程：$s_1(1+d_A)=k-r_1d_A \bmod n$

$$s_2(1+d_A)=k-r_2d_A \bmod n$$

两式相减得：$(s_1-s_2)(1+d_A)=(r_2-r_1)d_A \bmod n$
整理得：$d_A(s_1-s_2+r_1-r_2)=s_2-s_1 \bmod n$
因此：$d_A=(s_2-s_1)\cdot(s_1-s_2+r_1-r_2)^{-1} \bmod n$。

四、场景 3：不同用户重复使用 k 导致相互推导私钥

1、误用描述

Alice 和 Bob 使用相同随机数 k 分别签名，双方可通过对方签名推导其私钥。

2、PoC 验证步骤

参数初始化

Alice：私钥d_A，公钥P_A；Bob：私钥d_B，公钥P_B。

复用k签名

Alice 签名M_1：$r_1 = (H(Z_A \| M_1) + x) \bmod n$，$s_1 = ((1 + d_A)^{-1} \cdot (k - r_1 d_A)) \bmod n$。

Bob 签名M_2：复用 k，$r_2 = (H(Z_B \| M_2) + x) \bmod n$，$s_2 = ((1 + d_B)^{-1} \cdot (k - r_2 d_B)) \bmod n$。

私钥推导

Alice 推导d_B：$d_B = (k - s_2) \cdot (s_2 + r_2)^{-1} \bmod n$。

Bob 推导d_A：$d_A = (k - s_1) \cdot (s_1 + r_1)^{-1} \bmod n$。

验证：分别验证公钥一致性。

3、推导过程

以 Bob 的签名为例，由$s_2 = ((1 + d_B)^{-1} \cdot (k - r_2 d_B)) \bmod n$，

整理得：$s_2(1 + d_B) = k - r_2 d_B \bmod n \; d_B(s_2 + r_2) = k - s_2 \bmod n$

因此：$d_B = (k - s_2) \cdot (s_2 + r_2)^{-1} \bmod n$。

五、场景 4：同一 d 和 k 在 ECDSA 与 SM2 中使用导致私钥泄露

1、误用描述

同一私钥 d 和随机数 k 分别用于 ECDSA 和 SM2 签名，攻击者可通过两个签名推导 d。

2、PoC 验证步骤

参数初始化

私钥 d，公钥$P = d \cdot G$，复用 k。

跨算法签名

ECDSA 签名 M：$e_1 = hash(M)$，$r_1 = x \bmod n$，（$kG = (x, y)$），$s_1 = (e_1 + r_1 d) \cdot k^{-1} \bmod n$。

SM2 签名 M：$e_2 = H(Z_A \| M)$，$r_2 = (e_2 + x) \bmod n$，$s_2 = ((1 + d)^{-1} \cdot (k - r_2 d)) \bmod n$。

私钥推导

根据公式$d = (s_1 s_2 - e_1) \cdot (r_1 - s_1 s_2 - s_1 r_2)^{-1} \bmod n$计算 d。

验证：公钥一致性验证。

3、推导过程

由 ECDSA 得：$dr_1 = ks_1 - e_1 \bmod n$
由 SM2 得：$d(s_2 + r_2) = k - s_2 \bmod n$
联立消去 k 并整理得：$d = (s_1 s_2 - e_1) \cdot (r_1 - s_1 s_2 - s_1 r_2)^{-1} \bmod n$。

六、结论

SM2 签名算法的安全性严重依赖随机数 k 的保密性和唯一性，以及参数在不同算法中的隔离性。上述 PoC 验证表明，任何对随机数的泄露、重复使用或跨算法复用都会导致私钥泄露风险。实际应用中需严格遵循规范，采用 RFC6979 等标准生成随机数，避免参数误用。