签名算法中存在的隐患与攻击

一、 ECDSA

一、ECDSA

1. 密钥生成：选择一条椭圆曲线 $E_p(a,b)$ 与基点 G
n 为 G 的阶，选择私有密钥 $d<n$，利用基点 G 计算公开密钥 $P=dG$
即 $sk=d$，$pk=P$

2. 签名算法：$sign_{sk}(m)$
$k\leftarrow Z_n^*$，$R=kG$，$r=R_x \bmod n$（其中 $r\neq 0$，否则重新选择）
$e=hash(m)$，$s=r$ $s=k^{-1}(e+dr)\bmod n$ 输出签名 (r,s)

3. 验证算法：$vrfy_{pk}(m,r,s)$
$e=hash(m)$ $w=s^{-1}\bmod n$ $(r',s')=e\cdot wG+r\cdot wP$
当且仅当 $r'==r$ 时，验证通过，输出1；否则输出0

4. 正确性证明：$es^{-1}G+rs^{-1}P=e\cdot wG+r\cdot wP=es^{-1}G+rs^{-1}P$
$=s^{-1}(eG+rP)=k(e+dr)^{-1}(eG+rP)=k(e+dr)^{-1}(eG+drG)$
$=k(e+dr)^{-1}(e+dr)G=kG=R=(r',s')$
$\because r=R_x \bmod n$ $\therefore$ 当且仅当 $r=r'$ 时，验证通过，正确性即证

二、 SM2

二、SM2

1. 预计算：$Z_A=H_{256}(ENTL_A\|ID_A\|a\|b\|x_G\|y_G\|x_A\|y_A)$
2. 密钥生成：$P_A=d_A\cdot G$，阶为 n，$sk=d_A$，$pk=P_A$
3. 签名算法：$sign(Z_A,M)$：$sign_{d_A}(M,Z_A)\rightarrow(r,s)$
令 $\overline{M}=Z_A\|M$，$e=H_v(\overline{M})$，$k\leftarrow Z_n^*$，$kG=(x_1,y_1)$
$r=(e+x_1)\bmod n$，$s=((1+d_A)^{-1}\cdot(k-r\cdot d_A))\bmod n$，输出签名 (r,s)
4. 验证算法：$vrfy_{P_A}(r,s,M)$，
$Z_A=H_{256}(ENTL_A\|ID_A\|a\|b\|x_G\|y_G\|x_A\|y_A)$
令 $\overline{M}=Z_A\|M$，$e=H_v(\overline{M})$，$t=(r+s)\bmod n$，$(x_1,y_1)=sG+tP_A$
$R=(e+x_1)\bmod n$，当且仅当 $R=r$ 时，输出1；否则输出0

三、 k 的泄露会导致 d 的泄露

三、k 的泄露会导致 d 的泄露

① SM2：使用 $\sigma=(s,r)$ 与 k，可计算 d_A
$s=((1+d_A)^{-1}\cdot(k-r\cdot d_A))\bmod n$，$s(1+d_A)=(k-r\times d_A)\bmod n$
$\therefore d_A=(s+r)^{-1}\cdot(k-s)\bmod n$

② ECDSA：使用 $\sigma=(s,r)$ 与 k，可计算 d
$s=k^{-1}(e+dr)\bmod n$，$sk=(e+dr)\bmod n$ 且 $e=H(m)$，
$\therefore d=(sk-e)\cdot r^{-1}\bmod n$，其中 $e=H(m)$ 可计算。

四、 k的重用会导致d的泄露（同一用户）

四、k的重用会导致d的泄露（同一用户）

① SM2：㈠使用 d_A 对 M_1 签名。

随机选 $k \in [1, n-1]$，计算 $kG=(x,y)$

$r_1 = (Hash(Z_A \| M_1) + x) \bmod n$

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

㈡使用 d_A 对 M_2 签名。

随机选 k，但因为随机算法不好，k 发生了重用，$kG=(x,y)$

$r_2 = (Hash(Z_A \| M_2) + x) \bmod n$

$s_2 = ((1+d_A)^{-1} \cdot (k - r_2 \cdot d_A)) \bmod n$

㈢ 由以上2个签名，推出 d_A。

$s_1(1+d_A) = (k - r_1 \cdot d_A) \bmod n$

$s_2(1+d_A) = (k - r_2 \cdot d_A) \bmod n$

$$d_A = \frac{s_2 - s_1}{s_1 - s_2 + r_1 - r_2} \bmod n$$

② ECDSA：同理，用 d 分别对 M_1，M_2 签名，且重用了 k，得到了 (r_1, s_1)，(r_2, s_2)

$$\begin{cases} s_1 k = e_1 + d r_1 \\ s_2 k = e_2 + d r_2 \end{cases}$$ 其中 $\begin{cases} e_1 = H(M_1) \\ e_2 = H(M_2) \end{cases}$ 由 $k = s_1^{-1}(e_1 + d r_1)$

有 $s_2 s_1^{-1}(e_1 + d r_1) = e_2 + d r_2$，$s_2 s_1^{-1} e_1 - e_2 = d r_2 - s_2 s_1^{-1} d r_1 = d(r_2 - s_2 s_1^{-1} r_1)$

$$\therefore d = \frac{s_2 s_1^{-1} e_1 - e_2}{r_2 - s_2 s_1^{-1} r_1} \bmod n$$，其中 $\begin{cases} e_1 = H(M_1) \\ e_2 = H(M_2) \end{cases}$ 可计算。

五、 不同用户重用了k，可推导出得到对方的私钥

五、不同用户重用了k，可推出对方的私钥

① SM2：㈠A使用 d_A 对 M_1 进行了签名，$\sigma_A = (r_1, s_1)$

随机选择 $k \in [1, n-1]$，计算 $kG=(x,y)$

$r_1 = (Hash(Z_A \| M_1) + x) \bmod n$

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

㈡B使用 d_B 对 M_2 进行了签名，$\sigma_B = (r_2, s_2)$

重用了相同的 k，计算 $kG=(x,y)$

$r_2 = (Hash(Z_B \| M_2) + x) \bmod n$

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

㈢ A可以得到B的私钥：$$d_B = \frac{k - s_2}{s_2 + r_2} \bmod n$$

B可以得到A的私钥：$$d_A = \frac{k - s_1}{s_1 + r_1} \bmod n$$

② ECDSA：同理，A使用 d_A 对 M_1 签名得到 $\sigma_A = (r_1, s_1)$

B使用 d_B 对 M_2 签名得到 $\sigma_B = (r_2, s_2)$，且A与B使用了相同的 k。

~~$\begin{cases} s_1 k \neq e_1 + d_A r_1 \\ s_2 k \neq e_2 + d_B r_2 \end{cases}$~~ $\begin{cases} s_1 k = e_1 + d_A r_1 \\ s_2 k = e_2 + d_B r_2 \end{cases}$ 其中 $\begin{cases} e_1 = H(M_1) \\ e_2 = H(M_2) \end{cases}$ 可计算

A可以得到B的私钥：$d_B = (s_2 k - e_2) \cdot r_2^{-1} \bmod n$，其中 $e_2 = H(M_2)$

B可以得到A的私钥：$d_A = (s_1 k - e_1) \cdot r_1^{-1} \bmod n$，其中 $e_1 = H(M_1)$

六、 在不同的椭圆曲线算法中使用了相同的d和k，会导致d的泄露

六、 在不同的椭圆曲线算法中使用相同的 d 与 k，会导致 d 的泄露.

常见的椭圆曲线算法有 ECDSA, SM2, Schnorr 等，以下以 ECDSA 和 SM2 举例.

(1) 在 ECDSA 中用 d 对 m 签名，得到 (r_1, s_1)

随机选择 k，$R = kG = (x, y)$，$e_1 = hash(m)$，$r_1 = x \bmod n$

$s_1 = (e_1 + r_1 d) k^{-1} \bmod n$，输出签名 (r_1, s_1)

(2) 在 SM2 中用 d 对 m 签名，得到 (r_2, s_2)

重用相同的 k，$(x, y) = kG$，$e_2 = h(Z_A \| m)$，$r_2 = (e_2 + x) \bmod n$

$s_2 = (1+d)^{-1} \cdot (k - r_2 d) \bmod n$，输出签名 (r_2, s_2)

(3) 由 $m, (r_1, s_1), (r_2, s_2)$ 可推出 d

$$\begin{cases} d \cdot r_1 = k s_1 - e_1 \bmod n \\ d \cdot (s_2 + r_2) = k - s_2 \bmod n \end{cases}$$

$$d = \frac{s_1 s_2 - e_1}{r_1 - s_1 s_2 - s_1 r_2} \bmod n$$

七、 若验证时不需要验证 m，则签名可以被伪造

二、ECDSA 签名伪造.

PS: 以下构造基于不要求具体的消息，而仅要求提供消息的 Hash 即可的情景.

已知的信息：公钥：P，公开参数：G，n

首先选择 $u, v \in F_n^*$

计算 $R' = (x', y') = uG + vP$，为了便于签名运算，需令 $r' = x' \bmod n$

因为 $s'^{-1}(e'G + r'P) = uG + vP$

所以有 $\begin{cases} s'^{-1} r' = v \bmod n \\ s'^{-1} e' = u \bmod n \end{cases}$ 因此可先求出 $s' = r' v^{-1} \bmod n$

然后求出 $e' = r' u v^{-1} \bmod n = Hash(m')$

若 Hash 足够安全，则不从可忽略的概率有求出相应的 m'

但因此情景中，不要求求出 m'，求出 Hash(m') 即可

因此可以成功伪造出公钥为 P 的用户对于 $Hash(m') = e'$ 的合法签名 (r', s')