GG’20

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/61ce0b63-b2c0-466f-8042-7f20c4e96881/2020-GG20-One20Round20Threshold20ECDSA20with20Identifiable20Abort-1.pdf

一. 基础知识

理解 GG’20 方案之前，建议对以下概念/算法先建立基本的认识。在方案中若遇到相关算法，再对具体的算法进行深入研究即可。

1.1 DSA /ECDSA/DSS

DSA 全称是数字签名算法，于1991年由 Kravitz 提出，其最初是定义在有限域上的数字签名算法。ECDSA 则是将 DSA 扩展到椭圆曲线上的版本。DSA 后经 NIST 指定为 DSS，即数字签名标准。

Reference:

1. 关于 1991 年提出的 DSA 方案，请参考：

[5] Kravitz, D.W.: Digital signature algorithm (Jul 27 1993), uS Patent 5,231,668

2. 关于 DSS 数字签名标准的细节，请参考：

[39] Boneh, D.: Digital signature standard. In: Encyclopedia of cryptography and security, pp.347{347. Springer (2011)

3. 关于 G-DSA 请参考：

[29] Gennaro, R., Goldfeder, S., Narayanan, A.: Threshold-optimal dsa/ecdsa signatures and an application to bitcoin wallet security. In: International Conference on Applied Cryptography and Network Security. pp. 156{174. Springer (2016)

1.2 Paillier encryption: An additively homomorphic encryption

Reference:

1. 关于 Paillier 同态加密算法，请参考：

[44] Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp.223{238. Springer (1999).

1.3 DDH assumption, RSA assumption, Strong RSA assumption

DDH assumption 指的是：对于阶为素数q、生成元为 g 的循环群 G，G 中元素构成的三元组 $DH = \{(g^a, g^b, g^{ab}), a, b \in_R Z_q\}$ 和 $R = \{(g^a, g^b, g^c), a, b, c \in_R Z_q\}$在计算上是不可区分的。

RSA assumption & Strong RSA assumption：

选择两个安全素数 $p = 2p' + 1, q = 2q' + 1$，计算 $N = pq$。$\phi(N)$表示 N的欧拉函数。

RSA assumption 指的是，给定 e 和 $s \in_R Z_N^*$，找到 x 满足 $x^e \equiv s \mod N$ 在计算上是不可行的。

Strong RSA assumption 指的是，给定 $s \in_R Z_N^*$，找到 $x, e \neq 1$，满足 $x^e \equiv s \mod N$ 是不可行的。

Reference:

1. 关于 RSA assumption 的详细内容，请参考：

[45] Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and publickey
cryptosystems. Communications of the ACM 21(2), 120{126 (1978)

2. 关于 Strong RSA assumption 的详细内容，请参考：

[4] Baric, N., Ptzmann, B.: Collision-free accumulators and fail-stop signature schemes without trees. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 480{494. Springer (1997)

1.4 Non-malleable equivocable commitments

Reference:

[16] Damgard, I., Groth, J.: Non-interactive and reusable non-malleable commitment schemes. In: Proceedings of the thirty-fth annual ACM symposium on Theory of computing. pp. 426{437. ACM (2003)

[26] Gennaro, R.: Multi-trapdoor commitments and their applications to proofs of knowledge secure under concurrent man-in-the-middle attacks. In: Annual International Cryptology Conference. pp. 220{236. Springer (2004)

[43] MacKenzie, P., Yang, K.: On simulation-sound trapdoor commitments. In: International Conference on the Theory and Applications of Cryptographic Techniques. pp. 382{400. Springer(2004)

1.4 Secret sharing

为了分享一个秘密 s, (t,n) Shamir 门限秘密共享首先选择一个 $t-1$ 阶的多项式 $p=s+a_1x+a_2x^2+\cdots+a_{t-1}x^{t-1}$，$p$ 的常数项为秘密 s。 n 个参与者分别拥有多项式 p 上的 n 个不同的点作为秘密份额，如 $(1,\ p(1))$ ，$(2,p(2),\cdots(n,p(n)))$，则其中的任意 t 个参与者出示自己拥有的秘密份额，即可恢复出多项式 p，从而得到秘密 s（或者根据拉格朗日插值法直接恢复出 $s=p(0)$。

Feldman VSS 是一种可验证的秘密共享，是 Shamir 门限秘密共享的一个扩展版本，即增加了对参与者拥有秘密份额 $(i, p(i))$ 的正确性的验证过程。

在 **Feldman VSS** 方案中，一个可信第三方不仅会为每个参与者分配共享秘密 s 的一个秘密份额，同时还会发布系数 $s, a_1, a_2, \cdots, a_n$ 的验证值 $g^s, g^{a_1}, \cdots, g^{a_n}$。

对于参与者 p_i 所出示的秘密份额 $(i, p(i))$，其他参与方可以该秘密份额值代入等式：

$$g^s \cdot g^{a_1 \cdot i} \cdot g^{a_2 \cdot i^2} + \cdots + g^{a_{t-1} \cdot i^{t-1}} = g^{p(i)}$$

来验证，若等式成立，则说明参与者 p_i 出示了正确的秘密份额，是诚实的。否则，说明该参与者恶意地出示了错误的秘密份额。

Reference:

1. 关于 shamir 秘密共享的内容，请参考：

[47] Shamir, A.: How to share a secret. Communications of the ACM 22(11), 612{613 (1979)

2. 关于 Feldman VSS 的详细内容，请参考：

[*] P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," *28th Annual Symposium on Foundations of Computer Science (sfcs 1987)*, 1987, pp. 427-438, doi: 10.1109/SFCS.1987.4. **(Feldman’s VSS)**

1.5 ZK proofs

Reference:

[46] Schnorr, C.: Ecient signature generation by smart cards. J. Cryptology 4(3), 161{174 (1991)

[28] Gennaro, R., Goldfeder, S.: Fast multiparty threshold ecdsa with fast trustless setup. Cryptology ePrint Archive, Report 2019/114 (2019), https://ia.cr/2019/114.

[42] MacKenzie, P., Reiter, M.K.: Two-party generation of dsa signatures. In: Annual International Cryptology Conference. pp. 137{154. Springer (2001)

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/030672c4-4410-445a-b18e-a7cb1d2b6631/Two-Party_Generation_of_DSA_Signatures.pdf

1.6 Multiplicative-to-additive share conversion protocol (MtA/MtAwc)

MtA 协议 可以实现从 两方乘法秘密共享 到 两方加法秘密共享的转换。在两方乘法秘密共享中，参与者 p_1和 p_2 分别拥有乘法秘密份额 $a \in Z_q$ 和 $b \in Z_q$，即 共享秘密为 $x = ab \mod q$。经过 MtA 协议，p_1, p_2 分别拥有加法秘密份额 $\alpha \in Z_q, \beta \in Z_q$，$\alpha + \beta = x = ab \mod q$。

在 **MtA 协议** 中，参与者的输入没有验证过程，因此，某个参与者很有可能出示错误的 a 或者 b，从而生成错误的加法秘密份额。因此，有必要加入验证过程来提升安全性，我们将加入验证过程的 **MtA 协议** 称为 **MtAwc协议，即 MtA with check**。

Reference:

1. MtA协议 和 MtAwc 协议的详细描述，请参考:

[28] Gennaro, R., Goldfeder, S.: Fast multiparty threshold ecdsa with fast trustless setup. Cryptology ePrint Archive, Report 2019/114 (2019), https://ia.cr/2019/114.

二. One-Round Threshold ECDSA

为了便于对方案本身的理解，下文重点给出一个不带有 ZK proof 和 Commitment 的 ”纯净版“ 协议，主要目的时帮助读者建立对方案的一个整体的理解。然后，在此基础上进一步阐述 ZK proof 和 Commitment 技术给本协议带来的安全层面的保证。

首先假设所有参与者拥有 ECDSA 协议所基于的元素$\{G, g\}$，其中 G 为循环群，g 为生成元。并且每个参与者都拥有一个加法同态加密方案 Paillier 的公钥 E_i。

2.1 Key generation protocol

密钥生成协议具体执行过程如下：

Phase 1：在 密钥生成阶段，参与者 p_i选择随机值 $u_i \in Z_q$，同时广播自己的 Paillier 公钥 E_i。

Phase 2：参与者 p_i 针对 u_i 执行 (t, n) Feldman VSS 秘密共享，同时计算 $y_i = g^{u_i}$。那么，当 n个 Feldman VSS 秘密共享协议完成以后，每个参与者 p_i 获得签名私钥 x 的 (t, n) 秘密份额 x_i，满足$x = \sum u_i$，同时，公开 $X_i = g^{x_i}$。

Phase 3：生成与 Paillier 公钥 E_i 相关联的 RSA 模数 $N_i = p_i q_i$。

2.2 Signing protocol

执行签名协议时，首先输入待签名消息 M 的哈希值 m，以及 ”密钥生成协议“ 的输出。实际上，”密钥生成协议“ 是一个 t-out-of-n 的协议，即 签名私钥 x 通过 (t, n) Shamir 秘密共享协议进行共享。

假设在总共 n 个参与者中，有 $t+1$ 个参与者参与生成签名的过程，其构成集合 S，$|S| = t+1$，则根据拉格朗日插值法有：

$$x = \lambda_{1,S} \cdot x_1 + \lambda_{2,S} \cdot x_2 + \cdots \lambda_{t,S} \cdot x_t$$

令 $w_i = \lambda_{i,S} \cdot x_i$，则 $x = \sum_{i \in S} w_i$。

Phase 1：参与者 p_i 随机选择 $k_i, \gamma_i \in Z_q$，定义 $k = \sum_{i \in S} k_i, \gamma = \sum_{i \in S} \gamma_i$，同时广播 $\Gamma_i = g^{\gamma_i}$。则有：

$$k\gamma = \sum_{i,j \in S} k_i \gamma_j \mod q$$
$$kx = \sum_{i,j \in S} k_i w_j \mod q$$

Phase 2：n 个参与者当中的任意两个参与者 p_i, p_j 一起执行 **MtA 协议** 将乘法秘密共享 $k_i \cdot \gamma_j$ 转化为加法秘密共享 $\alpha ij + \beta_{ij}$，满足 $\alpha_{ij} + \beta_{ij} = k_i \cdot \gamma_j$。

那么，根据

$$k\gamma = \sum_{i,j \in S} k_i \gamma_j \mod q$$

可得：

$$k\gamma = \sum_{i \in S} k_i \gamma_i + \sum_{i \neq j} k_i \gamma_j = \sum_{i \in S} \delta_i$$

其中，$\delta_i = k_i \gamma_i + \sum_{i \neq j} \alpha_{ij} + \sum_{i \neq j} \beta_{ji}$ 为 $k\gamma$ 的 $(t, t+1)$加法秘密份额。

同理，p_i, p_j 运行 **MtAwc 协议** 将乘法秘密共享 $k_i \cdot w_j$ 转化为 加法秘密共享 $\mu_{ij} + \nu_{ji}$，满足 $\mu_{ij} + \nu_{ji} = k_i \cdot w_j$。

可得：

$$kw = \sum_{i,j \in S} k_i w_j = \sum_{i \in S} k_i w_i + \sum_{i \neq j} k_i w_j = \sum_{i \in S} \sigma_i$$

其中，$\sigma_i = k_i w_i + \sum_{i \neq j} \mu_{ij} + \sum_{i \neq j} \nu_{ji}$。

Phase 3：所有参与者广播 δ_i ，并联合计算 $\delta = \sum_{i \in S} \delta_i = k\gamma$，并进一步计算 $\delta^{-1} = k^{-1}\gamma^{-1} \mod q$。则可进一步计算

$$g^{k^{-1}} = (\prod_{i \in S} \Gamma_i)^{\delta^{-1}} = (g^{\sum_{i \in S} \gamma_i})^{k^{-1}\gamma^{-1}} = (g^{\gamma})^{k^{-1}\gamma^{-1}}$$

因此，$r = H\text{'}(R) = H'(g^{k^{-1}})$。

注意，签名 (r,s) 中的 r 已经计算出来了。

Phase 4：参与者 p_i 广播 $s_i = k_i m + r\sigma_i$，并联合生成签名的另一部分 $s = \sum_{i\in S} s_i$。

至此，各参与方联合生成了消息 m 的签名 (r,s)。

三. One-Round Threshold ECDSA with identifiable abort

为了加深读者对方案本身的理解，前文给出一个不带有 ZK proof 和 Commitment 的 ”纯净版“ 协议，主要目的是帮助读者建立对方案整体的理解。然而，本章将在前文的基础上加上这些安全性的保障（这也是 GG’20 相对于 GG’18 的安全性提升），并进一步阐述 ZK proof 和 Commitment 技术给本协议带来的安全层面的保证。

首先假设所有参与者拥有 ECDSA 协议所基于的元素$\{G,g\}$，其中 G 为循环群，g 为生成元。并且每个参与者都拥有一个加法同态加密方案 Paillier 的公钥 E_i。

3.1 Key generation protocol

密钥生成协议具体执行过程如下：

Phase 1：在 密钥生成阶段，参与者 p_i选择随机值 $u_i \in Z_q$，同时广播自己的 Paillier 公钥 E_i。

Phase 2：参与者 p_i 针对 u_i 执行 (t,n) Feldman VSS 秘密共享，同时计算 $y_i = g^{u_i}$，则公钥为 $y = \prod y_i$。那么，当 n个 Feldman VSS 秘密共享协议完成以后，每个参与者 p_i 获得签名私钥 $x = \sum u_i$ 的 (t,n) 秘密份额 x_i。同时，公开 $X_i = g^{x_i}$ 。

Phase 3：生成与 Paillier 公钥 E_i 相关联的 RSA 模数 $N_i = p_i q_i$。

3.2 Signing protocol

执行签名协议时，首先输入待签名消息 M 的哈希值 m，以及 "密钥生成协议" 的输出。实际上，"密钥生成协议" 是一个 t-out-of-n 的协议，即 签名私钥 x 通过 (t, n) Shamir 秘密共享协议进行共享。

假设在总共 n 个参与者中，有 $t+1$ 个参与者参与生成签名的过程，其构成集合 S，$|S| = t+1$，则根据拉格朗日插值法有：

$$x = \lambda_{1,S} \cdot x_1 + \lambda_{2,S} \cdot x_2 + \cdots \lambda_{t+1,S} \cdot x_{t+1}$$

令 $w_i = \lambda_{i,S} \cdot x_i$，则 $x = \sum_{i \in S} w_i$。

根据公开的 $X_i = g^{x_i}$和 $\lambda_{i,S}$，各参与方可以计算 $W_i = g^{w_i} = X_i^{\lambda_{i,S}}$。

Phase 1：参与者 p_i 随机选择 $k_i, \gamma_i \in Z_q$，定义 $k = \sum_{i \in S} k_i, \gamma = \sum_{i \in S} \gamma_i$，同时生成承诺 $[C_i, D_i] = \text{Com}(g^{\gamma_i})$并广播 C_i。

则有：

$$\begin{aligned} k\gamma &= \sum_{i,j \in S} k_i \gamma_j \mod q \\ kx &= \sum_{i,j \in S} k_i w_j \mod q \end{aligned}$$

Phase 2：n 个参与者当中的任意两个参与者 p_i, p_j 一起执行 **MtA 协议** 将乘法秘密共享 $k_i \cdot \gamma_j$ 转化为加法秘密共享 $\alpha ij + \beta_{ij}$，满足 $\alpha_{ij} + \beta_{ij} = k_i \cdot \gamma_j$。

那么，根据

$$k\gamma = \sum_{i,j \in S} k_i \gamma_j \mod q$$

可得：

$$k\gamma=\sum_{i\in S}k_i\gamma_i+\sum_{i\neq j}k_i\gamma_j=\sum_{i\in S}\delta_i$$

其中，$\delta_i=k_i\gamma_i+\sum_{i\neq j}\alpha_{ij}+\sum_{i\neq j}\beta_{ji}$ 为 $k\gamma$ 的 $(t,t+1)$加法秘密份额。

同理，p_i,p_j 运行 **MtAwc 协议** 将乘法秘密共享 $k_i\cdot w_j$ 转化为 加法秘密共享 $\mu_{ij}+\nu_{ji}$，满足 $\mu_{ij}+\nu_{ji}=k_i\cdot w_j$。

可得：

$$kw=\sum_{i,j\in S}k_iw_j=\sum_{i\in S}k_iw_i+\sum_{i\neq j}k_iw_j=\sum_{i\in S}\sigma_i$$

其中，$\sigma_i=k_iw_i+\sum_{i\neq j}\mu_{ij}+\sum_{i\neq j}\nu_{ji}$。

Phase 3：所有参与者 p_i 广播 δ_i ，联合计算 $\delta=\sum_{i\in S}\delta_i=k\gamma$，并进一步计算 $\delta^{-1}=k^{-1}\gamma^{-1}\mod q$。之后，$p_i$ 广播 $T_i=g^{\sigma_i}h^{l_i},l_i\in_R Z_q$通过 ZK 证明自己知道 σ_i 和 。

Phase 4：从参与者 p_i 的承诺解除信息 D_i 恢复出承诺解除值 Γ_i ，所有参与者联合计算

$$R=g^{k^{-1}}=(\prod_{i\in S}\Gamma_i)^{\delta^{-1}}=(g^{\sum_{i\in S}\gamma_i})^{k^{-1}\gamma^{-1}}=(g^{\gamma})^{k^{-1}\gamma^{-1}}$$

因此，$r=H'(R)=H'(g^{k^{-1}})$。

注意，签名 (r,s) 中的 r 已经计算出来了。

Phase 5：每个参与者广播 $\overline{R}_i=R^{k_i}$，

若：

$$g\neq\prod_{i\in S}\overline{R}_i$$

则中止协议。

Phase 6：每个参与者广播 $S_i=R^{\sigma_i}$，

若：

$$y \neq \prod_{i \in S} S_i$$

则中止协议。

Phase 7：参与者 p_i 广播 $s_i = k_i m + r\sigma_i$，并联合生成签名的另一部分 $s = \sum_{i \in S} s_i$。

至此，各参与方联合生成了消息 m 的签名 (r, s)。

通过公钥 y，各参与者验证生成的 m 的签名 (r, s) 是否合法。