软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software [doi: 10.13328/j.cnki.jos.006912]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

支持批量证明的 SM2 适配器签名及其分布式扩展*

涂彬彬[1,2,3], 陈 宇[1,2,3,4,5]

[1](山东大学 网络空间安全学院, 山东 青岛 266237)

[2](密码科学技术全国重点实验室, 北京 100878)

[3](密码技术与信息安全教育部重点实验室 (山东大学), 山东 青岛 266237)

[4](山东区块链研究院, 山东 济南 250001)

[5](泉城实验室, 山东 济南 250014)

通信作者: 陈宇, E-mail: yuchen@sdu.edu.cn

摘 要: 适配器签名, 又称无脚本脚本, 是解决区块链应用 (如密码货币) 中扩展性差、吞吐量低等问题的重要密码技术. 适配器签名可看作数字签名关于困难关系的扩展, 同时具有签名授权和证据提取两种功能, 在区块链应用中具有以下优点: (1) 降低链上成本; (2) 提高交易的可替代性; (3) 突破区块链脚本语言限制. SM2 签名是我国自主设计的国家标准签名算法, 在各种重要信息系统中有着广泛应用. 基于 SM2 签名构造出高效的适配器签名方案, 并在随机谕言机模型下给出安全性证明. 所提方案结合 SM2 签名结构, 可避免在预签名阶段生成额外的零知识证明, 与现有 ECDSA/SM2 适配器签名相比更加高效, 其中预签名生成效率提升 4 倍, 预签名验证效率提升 3 倍. 随后, 基于 SM2 协同签名, 构造分布式 SM2 适配器签名, 可避免单点故障问题, 提升签名私钥安全. 最后, 在实际应用方面, 基于 SM2 适配器签名构造适用于一对多场景下安全高效的批量原子交换协议.

关键词: SM2 算法; 适配器签名; 分布式适配器签名; 区块链; 原子交换

中图法分类号: TP309

中文引用格式: 涂彬彬, 陈宇. 支持批量证明的SM2适配器签名及其分布式扩展. 软件学报. http://www.jos.org.cn/1000-9825/6912.htm

英文引用格式: Tu BB, Chen Y. SM2-based Adaptor Signature with Batch Proofs and Its Distributed Extension. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/6912.htm

SM2-based Adaptor Signature with Batch Proofs and Its Distributed Extension

TU Bin-Bin[1,2,3], CHEN Yu[1,2,3,4,5]

[1](School of Cyber Science and Technology, Shandong University, Qingdao 266237, China)

[2](State Key Laboratory of Cryptology, Beijing 100878, China)

[3](Key Laboratory of Cryptologic Technology and Information Security of Ministry of Education (Shandong University), Qingdao 266237, China)

[4](Shandong Institute of Blockchain, Jinan 250001, China)

[5](Quan Cheng Laboratory, Jinan 250014, China)

Abstract: Adaptor signature, also known as scriptless script, is an important cryptographic technique that can be used to solve the problems of poor scalability and low transaction throughput in blockchain applications such as cryptocurrency. An adaptor signature can be seen as an extension of a digital signature on hard relations, and it ties together the authorization with witness extraction and has many advantages in blockchain applications, such as (1) low on-chain cost; (2) improved fungibility of transactions; (3) advanced functionality beyond the limitation of the blockchain's scripting language. SM2 signature is the Chinese national standard signature algorithm and has

* 基金项目: 国家重点研发计划 (2021YFA1000600); 国家自然科学基金 (62272269); 泰山学者青年专家项目
收稿时间: 2022-07-20; 修改时间: 2022-09-26, 2023-01-23; 采用时间: 2023-02-03; jos 在线出版时间: 2023-08-09

been widely used in various important information systems. This work designs an efficient SM2-based adaptor signature with batch proofs and gives security proofs under the random oracle model. The scheme avoids to generate zero-knowledge proofs used in the pre-signing phase based on the structure of SM2 signature and is more efficient than existing ECDSA/SM2-based adaptor signature. Specifically, the efficiency of pre-signature generation is increased by 4 times, and the efficiency of pre-signature verification is increased by 3 times. Then, based on distributed SM2 signature, this work develops distributed SM2-based adaptor signature which can avoid the single point of failure and improve the security of signing key. Finally, in real-world applications, this work gives a secure and efficient batch atomic swap protocol for one-to-many scenarios based on SM2-based adaptor signature.

Key words: SM2 algorithm; adaptor signature; distributed adaptor signature; blockchain; atomic swap

1 引 言

自 2009 年比特币[1]出现, 其底层的区块链结构引起了广泛关注. 区块链巧妙地融合了密码、共识机制、P2P 网络等技术, 具有去中心化、不可篡改、匿名性、可追溯性、开放透明等特点, 应用场景非常广泛. 然而, 现有的区块链应用 (如密码货币) 存在着可扩展性差、吞度量低等问题[2−6], 比如: 比特币的交易吞吐量约为每秒十几笔, 与信用卡每秒数万笔的交易量相比, 低了 3 个数量级. 区块链上每笔交易可看作某种脚本构成的应用程序, 如比特币等支持签名授权的货币转账功能; 以太坊等提供图灵完备的脚本语言, 可编码更复杂的交易逻辑实现. 但是, 丰富的功能需要复杂的脚本支持, 导致链上的存储和矿工计算成本增加. 针对上述问题, 支付通道网络[5,7]通过在链上建立通道, 实现用户之间任意次数的链下交易, 减小链上交易规模, 提高交易吞吐量, 降低链上成本, 是目前较为高效且部署广泛的解决方案, 如比特币的闪电网络[8]和以太坊的雷电网络[9]等. 适配器签名[5,10]作为构建支付通道网络的关键技术[11−14], 是解决区块链可扩展性差、吞吐量低等问题的重要工具.

Poelstra[10]首次引入无脚本脚本 (scriptless script) 的概念. 随后由 Aumayr 等人[5]形式化定义为适配器签名 (adaptor signature, AS). 适配器签名可看作是数字签名对困难关系 (hard relation) 的扩展, 包含原签名方案的密钥生成算法、签名算法和签名验证算法外, 还包含了预签名算法 pSign、预签名验证算法 pVrfy、适配算法 Adapt, 以及证据提取算法 Ext. 适配器签名通过嵌入困难关系 (如离散对数), 同时具有签名授权和证据提取两种功能, 在应用中, 签名者可以根据困难关系的实例使用签名私钥对消息进行预签名, 生成预签名值. 该预签名值可由困难关系的证据适配为完整签名值, 同时困难关系的证据可通过预签名值和完整签名值进行提取.

在区块链应用中, 适配器签名可提供原子交换功能, 实现两方 (U_0 和 U_1) 的跨链公平交换货币 (c_0 和 c_1), 具体如图 1 所示. 首先, 交换双方需要使用时间锁 (time-lock) 限制待交换货币, 其中, 时间锁主要给 U_1 足够的时间完成交换, 防止 U_0 在提取 U_1 的货币后, 再提取自己的货币. 交换方 U_0 生成困难关系 $GenR(1^\lambda)\rightarrow(Y,y)$, 根据实例 Y 对交换交易 tx_0 (即, U_0 向 U_1 转 c_0) 生成预签名 $\hat{\sigma}_0$ 发送给交换方 U_1, U_1 验证 $\hat{\sigma}_0$ 的正确性后, 根据实例 Y 对交换交易 tx_1 (即, U_1 向 U_0 转 c_1) 进行预签名, 并将预签名值 $\hat{\sigma}_1$ 返回; U_0 验证 $\hat{\sigma}_1$ 的正确性, 并根据困难关系的证据 y 将预签名值 $\hat{\sigma}_1$ 适配成完整签名值 σ_1, 在链上公布 σ_1 可获得交换的货币 c_1; U_1 根据 $\hat{\sigma}_1$ 和 σ_1 可提取证据 y, 并将 $\hat{\sigma}_0$ 适配成完整签名值 σ_0, 在链上公布 σ_0 可获得交换的货币 c_0, 完成两方的公平交换. 相较于基于哈希时间锁合约 (hash time-lock contracts) 技术[15]构造的原子交换协议, 适配器签名的预签名的生成、传输和验证都在链下完成, 避免了链上支持相同哈希原像条件脚本 (preimage conditioned scripts) 的限制, 降低了链上的存储和验证成本[16−18]. 得益于适配器签名的应用优势, Malavolta 等人[13]基于适配器签名构造匿名多跳锁 (anonymous multi-hop lock) 协议, 并以此为基础构建了安全的支付通道网络. Thyagarajan 等人[19]基于 ECDSA/Schnorr 适配器签名给出两方通用原子交换协议的高效实例化; Aumayr 等人[5]基于适配器签名在脚本受限的 (非图灵完全) 区块链上, 构建了广义通道 (generalized channels) 结构, 可将链上交易安全转移到链下执行, 提升区块链的扩展性.

SM2 签名算法[20,21]是我国国家密码管理局发布的拥有完全自主知识产权的国家标准签名算法. 该算法基于椭圆曲线密码体制, 具有安全性高、签名速度快、占用空间小等优势, 在我国商密领域各类信息系统中有着重要应用. 目前, 适配器签名主要基于 Schnorr 签名[5,10,22,23]、ECDSA 签名[5,24−26], 以及格签名[27,28]等方案构造, 基于 SM2 构造的适配器签名的研究[29]较少, 限制了 SM2 算法在区块链场景下的推广应用. 此外, 现有基于 ECDSA 或 SM2

的适配器签名[5,29]在预签名阶段, 签名者需要以随机值或签名私钥作为证据来计算预签名公共参数和相应的零知识证明, 一方面难以进行分布式扩展; 另一方面, 在多个参与方的情况下, 如 (批量) 原子交换[16]或多跳支付场景[13], 零知识证明的数量与参与者的数量线性相关, 在具体应用中效率不高. 因此, 秉承着核心技术自主创新、信息安全自主可控的理念, 本文主要探索如何基于 SM2 签名设计安全高效的 SM2 适配器签名方案, 为 SM2 算法在区块链场景中提供更加高效契合的应用参考, 为设计安全可控的区块链应用提供保障.

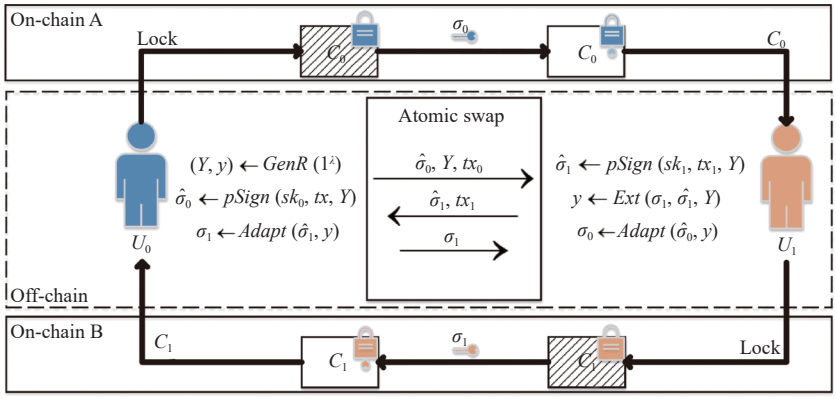


图 1　基于适配器签名的原子交换协议

1.1　本文贡献

在本文中, 我们基于 SM2 签名算法构造了适配器签名方案 (SM2-AS), 并在随机谕言机模型 (random oracle model) 下基于 SM2 签名的安全性给出安全性证明. 该方案以困难关系的证据作为连系 SM2 签名验证公钥和预签名公开参数的证据, 可由困难关系生成方离线批量生成所有预签名公开参数和对应的零知识证明, 特别是考虑具体预签名验证场景, 我们的 SM2-AS 可进一步省略该部分的零知识证明, 相较于现有的 ECDSA 适配器签名 (ECDSA-AS$_k$)[5]和 SM2 适配器签名 (SM-AS$_x$)[29], 预签名的生成和验证效率更高, 预签名值的尺寸更小. 得益于预签名阶段避免了零知识证明, SM2-AS 的预签名操作类似 SM2 签名算法, 我们可基于现有 SM2 协同签名对 SM2-AS 进行分布式扩展, 构造分布式的 SM2 适配器签名. 该分布式方案能有效避免应用系统中的单点故障, 提高签名私钥的安全性. 最后, 针对现有的原子交换协议难以高效适用于一对多应用场景的问题, 我们提出批量原子交换协议, 并基于 SM2-AS 给出了安全高效的构造. 本文主要技术介绍如下.

● SM2 适配器签名. SM2 适配器签名包含原 SM2 密钥生成算法、签名算法和签名验证算法, 还包含了预签名算法、预签名验证算法、适配算法, 以及证据提取算法. 具体 SM2-AS 方案介绍如下: 令 $(X = xG, x)$ 为 SM2 签名算法的公私钥对, 适配器签名的困难关系分别包含: 离散对数的困难关系 (discrete logarithms, DL) $R_Y^{\mathrm{DL}} = \{((Y,\pi_Y),y)|\exists y \in \mathbb{Z}_n, \text{s.t. } Y = yG \wedge \pi_Y \leftarrow P_Y^{DL}(Y,y)\}$ (其中, 零知识证明系统需要满足直线证据提取性[30], 在安全性证明中, 模拟器可根据困难关系的自证明结构, 通过直线证据提取器 (straight-line extractor), 也叫在线提取器 (online extractor) 提取证据 y) 和离散对数相等困难关系 (equality of discrete logarithms, EDL) $R_Z^{\mathrm{EDL}} = \{((Y,Z,\pi_Z),y)|\exists y \in \mathbb{Z}_n, \text{s.t. } Z = y(X+G) \wedge Y = yG \wedge \pi_Z \leftarrow P_Z^{\mathrm{EDL}}((Y,Z),y)\}$ (该零知识证明系统不需要满足较强的直线证据提取性，可使用高效的 Fiat-Shamir 启发式[31]转换 Chaum-Pedersen 的 Σ-协议[32]实现); 预签名算法 pSign: 选择随机数 $k \in \mathbb{Z}_n$, 并根据点 Z 计算随机点 $\hat{K} = kG + Z = (k + y(x+1))G = (r_x, r_y)$, $r = r_x + H(m) \bmod n$, $\hat{s} = (1+x)^{-1}(k - r \cdot x) \bmod n$, 即预签名值 $\hat{\sigma} = (r, \hat{s})$; 预签名验证算法 pVrfy: 计算 $(r'_x, r'_y) = (\hat{s} + r)X + \hat{s}G + Z$, $r' = r'_x + H(m) \bmod n$, 验证 $r = r'$ 是否成立? 适配算法 *Adapt*: 基于证据 y 将 $\hat{\sigma}$ 转化为完整 SM2 签名值 $\sigma = (r, s)$, 其中 $s = \hat{s} + y = (x+1)^{-1}(k + y(x+1) - r \cdot x)$; 证据提取算法 *Ext*: 根据预签名值 $\hat{\sigma}$ 和签名值 σ 提取证据 $y = s - \hat{s} \bmod n$.

● 分布式 SM2 适配器签名. SM2-AS 的预签名算法类似 SM2 签名算法, 除了计算的随机点不同. 具体而言, SM2 签名算法计算随机点 $K = kG$, SM2-AS 预签名算法计算随机点 $\hat{K} = kG + Z$. 因为困难实例 Z 是公开可验证的,

所以分布式 SM2 适配器签名可沿用现有分布式 SM2 签名的构造方法, 对 SM2-AS 进行分布式扩展, 即在预签名阶段类似分布式 SM2 签名, 各参与方协同计算 $K=kG$ 后, 加上困难实例 Z 计算随机点 $\hat{K}=kG+Z$. 相较于现有的 ECDSA/SM2 适配器签名[5,29], ECDSA-AS$_k$[5]和 SM-AS$_x$[29]需要在预签名阶段分别以随机数 k 和签名私钥 x 作为证据计算预签名公开参数和额外的零知识证明. 因为在分布式构造中各参与方只能拥有随机数 k 和签名私钥 x 的秘密分享, 所以在计算预签名时, 预签名阶段的零知识证明需要多方协同生成, 导致了预签名的分布式构造困难. 因此, SM2-AS 避免了在预签名阶段生成预签名公开参数和对应的零知识证明, 更易于分布式扩展, 对底层的分布式 SM2 签名结构并无要求, 可基于现有的任意 SM2 协同签名直接构造.

● 安全性证明概述. 适配器签名需要满足不可伪造性和证据可提取性, 与签名方案的安全性区别在于敌手不仅能访问签名谕言机, 还能额外访问预签名谕言机. 为了保证可证明安全, 我们在构造 SM2 适配器签名时, 沿用自证明结构的困难关系[5]. 具体而言, 在离散对数困难关系 (Y,y) 的基础上增加额外的零知识证明, 将其扩展为具有自证明结构的离散对数困难关系 $(I_Y^{\mathrm{DL}}=(Y,\pi_Y),y)\in R_Y^{\mathrm{DL}}$, 其中, $\pi_Y\leftarrow P_Y^{\mathrm{DL}}(Y,y)$ 证明存在证据 y 使得 $Y=yG$. 在安全性证明中, 模拟器可根据困难关系的自证明结构, 在随机谕言机模型下通过直线证据提取器功能提取证据 y, 然后结合 SM2 签名谕言机输出的签名值计算预签名值, 模拟 SM2 适配器签名的预签名谕言机, 同时模拟器可根据自己的随机谕言机和 SM2 签名谕言机模拟敌手要访问的这两个谕言机. 因此, 模拟器可模拟敌手的视角, 将 SM2 适配器签名的安全性归约到 SM2 签名的安全性上.

● 批量证明技术. SM2-AS 和 ECDSA-AS$_k$[5], 以及 SM-AS$_x$[29]都基于困难关系的自证明结构保证可证明安全. 不同的是, 我们通过结合 SM2 签名结构嵌入合适的困难关系, 可避免预签名阶段额外计算预签名公开参数和对应的零知识证明. 具体而言, 在预签名过程中, ECDSA-AS$_k$[5]需要以预签名的随机数 k 作为证据, 计算公开参数 $K=kY$, 并证明离散对数相等的困难关系 $R_K^{\mathrm{EDL}}=\{((G,\hat{K},Y,K),k)|\exists k\in\mathbb{Z}_n, \text{s.t. } \hat{K}=kG\wedge K=kY\}$. 该证明只能由预签名方生成 (随机数由预签名方自己选择). SM-AS$_x$[29]需要以签名私钥 x 作为证据, 计算预签名公开参数 $Z=xY+Y$, 并证明困难关系 $R_X^{\mathrm{EDL}}=\{((G,X,Y,Z),x)|\exists x\in\mathbb{Z}_n, \text{s.t. } X=xG\wedge Z-Y=xY\}$. 该证明只能由预签名方生成 (签名私钥由预签名方自己拥有).

我们结合 SM2 签名的结构, 在困难关系 $(I_Y^{\mathrm{DL}}=(Y,\pi_Y),y)$ 之外, 附加困难关系 $((Y,Z,\pi_Z),y)\in R_Z^{\mathrm{EDL}}$, 其中 $Y=yG$, $Z-Y=yX$. 该部分可由困难关系生成方 (拥有 y, 获得签名公钥 X) 批量生成. 因此, 在具体应用中, 困难关系生成方可以在协议执行前, 对需要进行预签名的所有参与方 U_i, 离线批量计算预签名公开参数 Z_i 和对应的零知识证明 π_{Z_i}. 该批量操作可避免参与方在预签名阶段生成零知识证明, 极大地提升预签名的生成和验证效率. 特别是, 考虑适配器签名中预签名的特殊验证场景, 根据预签名公开参数的结构 $Z_i=y(X_i+G)=(x_i+1)Y$, 各参与方 U_i 拥有签名私钥 x_i 可验证 Z_i 的正确性, 不需要额外的零知识证明 π_{Z_i}, 进一步提升 SM2-AS 的效率.

● 性能评估. 我们从理论和实验两个方面分别对比了 SM2-AS、最优 ECDSA-AS$_k$[5]和 SM-AS$_x$[29]. 因为 ECDSA-AS$_k$[5] 和 SM-AS$_x$[29]都需要在预签名阶段计算额外的零知识证明, 并在预签名验证阶段验证零知识证明的正确性, 所以我们的 SM2-AS 的预签名生成和验证更加高效. 具体而言: ECDSA-AS$_k$[5]和 SM-AS$_x$[29]在预签名生成阶段需要 4 次点乘操作, 在预签名验证阶段需要 6 次点乘操作; 我们的 SM2-AS 在预签名生成阶段需要 1 次点乘操作, 在预签名验证阶段需要 2 次点乘操作. 随后, 我们基于 OpenSSL 实现了 SM2-AS、ECDSA-AS$_k$[5]和 SM-AS$_x$[29], 并对比了 3 种方案的预签名算法和预签名验证算法的计算效率. 具体实现发布于: https://github.com/tbb-tobebetter/SM2-adaptor-signature. 实验结果显示 SM2-AS 的预签名计算耗时约为 ECDSA-AS$_k$[5] 和 SM-AS$_x$[29]的 1/4, 预签名验证耗时约为 ECDSA-AS$_k$[5]和 SM-AS$_x$[29]的 1/3. 实验结果符合理论分析.

● 实际应用. 目前的原子交换协议只考虑了两方公平交换的功能实现, 如图 1 所示, 但是在面向交易用户较多的场景, 比如交易所等需要批量向多数用户同时交换, 或者单一用户同时向多个账户 (地址) 交换的场景, 独立并行地执行多次一对一的原子交换协议效率较低. 因此, 在两方原子交换协议的基础上, 我们考虑发起方 U_0 向 n 个交换方 U_i, $i\in[1,n]$ 进行一对多的跨链货币交换的场景, 基于 SM2 适配签名给出批量原子交换协议的安全高效的构造. 对比执行 n 次一对一的原子交换协议需要 n 个困难关系, 我们通过对困难关系进行拆分, 考虑原子交换协议

中适配器签名的特殊验证场景, 可实现一个困难关系用于一次批量交换 (即 U_0 向 n 个参与方 U_i, $i\in[1,n]$ 同时交换). 相较于基于 ECDSA-AS$_k$[5]和 SM-AS$_x$[29]构造批量原子交换协议, 我们的 SM2-AS 可减少预签名阶段的零知识证明和证明验证过程, 极大地提升了协议效率. 具体而言, 基于 ECDSA-AS$_k$[5]构造批量原子交换协议需要 $2n$ 个预签名公开参数和 $2n$ 个对应的零知识证明; 基于 SM-AS$_x$[29]构造批量原子交换协议需要 $n+1$ 个预签名公开参数和 $n+1$ 个对应的零知识证明; 基于 SM2-AS 构造批量原子交换协议只需要 $n+1$ 个预签名公开参数和 1 个对应的零知识证明.

2 基础知识

2.1 符 号

本文中, 用 $\mathbb{N}$ 表示自然数集, 用 $\lambda\in\mathbb{N}$ 表示安全参数. 对任意 $n\in\mathbb{N}$, 符号 $[1,n]$ 表示集合 $\{1,\dots,n\}$. 对任意有限集 X, $x\leftarrow X$ 表示从 X 中均匀随机地选取出一个元素 x. 关于安全参数 λ 的函数 $negl$, 如果 $negl(\lambda)\geqslant 0$, 而且对任意正多项式 $poly$, 存在正整数 $c\in\mathbb{N}$, 使得对于所有的 $n\geqslant c$ 满足 $negl(n)\leqslant 1/(poly(n))$, 我们称函数 $negl$ 是可忽略的. 对任意概率算法 $\mathcal{A}$, 记 $\mathcal{A}$ 所用的随机数集合为 R. $y\leftarrow\mathcal{A}(x_1,\dots,x_t;r)$ 表示 $\mathcal{A}$ 以 $(x_1,\dots,x_t)$ 和随机数 $r\leftarrow R$ 为输入, 输出 y; 通常我们忽略 r, 记为 $y\leftarrow\mathcal{A}(x_1,\dots,x_t)$. 如果 $\mathcal{A}$ 的运行时间是关于安全参数 λ 的多项式, 则称 $\mathcal{A}$ 为概率多项式时间 (probabilistic polynomial time, PPT) 算法.

2.2 签 名

签名方案包含 3 个多项式时间算法 $\sum=(Gen,Sign,Vrfy)$:

- $Gen(1^\lambda)$: 密钥生成算法输入安全参数 λ, 输出签名密钥对 (vk,sk).
- $Sign(sk,m)$: 签名算法输入签名私钥 sk 和消息 $m\in\{0,1\}^*$, 输出签名值 σ.
- $Vrfy(vk,m,\sigma)$: 验证算法输入验证公钥 vk, 消息 $m\in\{0,1\}^*$ 和签名值 σ, 输出 1 代表签名正确, 否则输出 0.

正确性. 对任意 $\lambda\in\mathbb{N}$, $(vk,sk)\leftarrow Gen(1^\lambda)$, $m\in\{0,1\}^*$, 则 $\Pr[Vrfy(vk,m,Sign(sk,m))\rightarrow 1]=1$.

SUF-CMA 安全. 如果对于任意 PPT 敌手 $\mathcal{A}$, 存在可忽略函数 $negl$, 使得在实验 sSigForge$_{\sum,\mathcal{A}}$ 中, 敌手优势 $\mathrm{Adv}_{\mathcal{A},\mathrm{sSigForge}}=\Pr[\mathrm{sSigForge}_{\mathcal{A},\sum}(\lambda)=1]\leqslant negl(\lambda)$, 则称签名方案 $\sum$ 满足选择消息攻击下强不可伪造 (strong existential unforgeablity under chosen message attack, SUF-CMA), 其中实验 sSigForge$_{\sum,\mathcal{A}}$ 定义如图 2.

$\mathrm{sSigForge}_{\mathcal{A},\sum}(\lambda)$	$\mathcal{O}_S(m)$
$\mathcal{Q}=\varnothing$	$\sigma\leftarrow Sign(sk,m)$
$(vk,sk)\leftarrow Gen(1^\lambda)$	$\mathcal{Q}=\mathcal{Q}\cup\{m,\sigma\}$
$(m,\sigma)\leftarrow\mathcal{A}^{\mathcal{O}_S(\cdot)}(vk)$	**return** σ
return $((m,\sigma)\notin\mathcal{Q}\wedge Vrfy(vk,m,\sigma))$	

图 2 选择消息攻击下强不可伪造实验

2.3 SM2 签名

令 $\mathbb{G}$ 为椭圆曲线上的群, G 为群 $\mathbb{G}$ 的基点, 阶为 n. 随机选择签名私钥 $x\in[1,n-2]$, 计算签名验证公钥 $X=xG$. 为了简洁, 本文省略 SM2 的编码部分, 如: H 表示哈希函数, 具体可见 SM2 签名标准[20]; 变换函数 f 定义为点的横坐标, 即 kG[5,33].

关于消息 $m\in\{0,1\}^*$ 的 SM2 签名步骤如下.

1. 计算 $e=H(m)$.
2. 选择随机数 $k\in[1,n-1]$, 计算 $r_x=f(kG)$.
3. 计算 $r=r_x+e \bmod n$, 如果 $r=0$ 或者 $r+k=n$ 则返回步骤 2.
4. 计算 $s=(1+x)^{-1}(k-rx) \bmod n$, 如果 $s=0$ 则返回步骤 2.
5. 输出签名值 $\sigma=(r,s)$.

SM2 签名 $\sigma=(r,s)$ 的验证步骤如下.

1. 如果 $r\notin[1,n-1]$ 或者 $s\notin[1,n-1]$, 输出 0 并退出.
2. 计算 $e=H(m)$.
3. 计算 $t=(r+s) \bmod n$, 如果 $t=0$, 输出 0 并退出.
4. 计算 $r'_x=f(sG+tX)$.
5. 计算 $r'=r'_x+e \bmod n$, 如果 $r'=r$ 则输出 1 否则输出 0.

SM2 签名算法是我国自主研发的签名算法, 具有安全性高, 运算速度快, 签名尺寸小等优势. SM2 签名算法的安全性分析已有较多工作, 具体可参考文献 [20,21,33].

2.4 困难关系和零知识证明

定义语言 $L_R=\{Y|\exists y, \text{ s.t. } (Y,y)\in R\}$. R 是困难关系[5], 如果以下条件成立:

- $GenR(1^\lambda)\to(Y,y)$: 存在 PPT 取样算法 $GenR$ 输入参数 1^λ, 输出关系 R 上的实例/证据对 (Y,y).
- 该关系是多项式时间可判定的.
- 对任意 PPT 敌手 $\mathcal{A}$, 输入实例 Y 输出证据 y 的概率是可忽略的.

对于困难关系 R, 一组算法 (P,V) 是在随机谕言机 $\mathcal{H}$ 下具有直线证据提取性的非交互零知识的知识证明 (non-interactive zero-knowledge proof of knowledge, NIZKPoK)[5,30], 需要满足以下条件.

- 完备性: 对于任意困难关系 $(Y,y)\in R$, $\pi_Y\leftarrow P^{\mathcal{H}}(Y,y)$, 存在可忽略函数 $negl$, 使得:

$$\Pr[V^{\mathcal{H}}(Y,y)=1]\geqslant 1-negl(\lambda).$$

- 零知识性: 对于任意困难关系 $(Y,y)\in R$, 存在 PPT 的模拟器 $\mathcal{S}$ 输入实例 Y 可模拟证明 π_Y.
- 直线证据提取性: 存在 PPT 提取算法 K, 输入实例 Y 和证明 π_Y, 可访问随机谕言机的询问序列以及对应回复, 能提取证据 y, 满足 $(Y,y)\in R$.

2.5 适配器签名

关于困难关系 $R=(Y,y)$ 和签名 $\sum=(Gen,Sign,Vrfy)$ 的适配器签名 $\sum_{\mathrm{AS}}=(pSign,pVrfy,Adapt,Ext)$ [5]定义如下.

- $pSign(vk,sk,m,Y)\to\hat{\sigma}$. 预签名算法输入签名验证公钥 vk 、签名私钥 sk 、消息 $m\in\{0,1\}^*$, 以及实例 Y, 输出预签名值 $\hat{\sigma}$.
- $pVrfy(vk,m,Y,\hat{\sigma})\to 0/1$. 预签名验证算法输入签名验证公钥 vk 、消息 $m\in\{0,1\}^*$、实例 Y, 以及预签名值 $\hat{\sigma}$, 如果预签名值验证成功输出 1, 否则输出 0.
- $Adapt(\hat{\sigma},y)\to\sigma$. 适配算法输入预签名值 $\hat{\sigma}$ 和证据 y, 输出签名值 σ.
- $Ext(\sigma,\hat{\sigma},Y)\to y$. 提取算法输入签名值 σ, 预签名值 $\hat{\sigma}$ 和实例 Y, 输出证据 y.

适配器签名需要满足原签名的正确性, 还需要满足预签名的正确性, 以及预签名的可适配性. 简单来说, 预签名的正确性可保证签名方诚实地生成关于实例 $Y\in L_R$ 的预签名值. 该预签名值能通过验证, 并且能被适配成一个有效的签名值. 基于该预签名值和适配后的完整签名值能提取出实例 Y 的证据. 该性质保证实例 $Y\in L_R$ 的预签名能被适配成一个有效的签名值, 当且仅当签名方知道实例 Y 的证据 y. 预签名的可适配性保证任意有效的预签名值和实例 Y 的证据 y 可以适配成一个有效的签名值, 即使该实例 Y 、预签名值由预签名方恶意地生成.

- 预签名正确性. 如果对任意 $\lambda\in\mathbb{N}$, $m\in\{0,1\}^*$, $(Y,y)\in R$ 以下式子成立, 则适配器签名满足预签名正确性.

$$\Pr\left[\begin{array}{c|c} pVrfy(vk,m,Y,\hat{\sigma})=1\wedge \\ Vrfy(vk,m,\sigma)=1\wedge \\ (Y,y')\in R \end{array}\right.\left.\begin{array}{c} Gen(1^\lambda)\to(vk,sk) \\ pSign((vk,sk),m,Y)\to\hat{\sigma} \\ Adapt(\hat{\sigma},y)\to\sigma \\ Ext(\sigma,\hat{\sigma},Y)\to y' \end{array}\right]=1.$$

- 预签名可适配性. 如果对任意 $\lambda\in\mathbb{N}$, $m\in\{0,1\}^*$, $(Y,y)\in R$, $Gen(1^\lambda)\to(vk,sk)$, 以及任意可验证成功的预签名 $\hat{\sigma}$,

$pVrfy(vk,m,Y,\hat{\sigma})=1$, 有 $\Pr[Vrfy(vk,m,Adapt(\hat{\sigma},y))\rightarrow 1]=1$, 则适配器签名满足预签名可适配性.

适配器签名需要满足原签名方案的选择消息攻击下的存在不可伪造性 (existential unforgeablity under chosen message attack, EUF-CMA), 还需要满足选择消息攻击下适配器签名的存在不可伪造性 (EUF-CMA for adaptor signature, aEUF-CMA), 以及适配器签名的证据可提取性 (witness extractability for adaptor signature, aWitExt). 具体安全性定义描述如下.

● 选择消息攻击下适配器签名的存在不可伪造性. 对于任意的 PPT 敌手 $\mathcal{A}$, 如果存在可忽略函数 $negl$, 使得敌手优势 $\mathrm{Adv}_{\mathcal{A},\mathrm{aSigForge}}=\Pr[\mathrm{aSigForge}_{\mathcal{A},\Pi_{R,\Sigma}}(\lambda)=1]\leqslant negl(\lambda)$ 成立, 则称适配器签名满足 aEUF-CMA, 其中 $\mathrm{aSigForge}_{\mathcal{A},\Pi_{R,\Sigma}}$ 定义如图 3.

$\mathrm{aSigForge}_{\mathcal{A},\Pi_{R,\Sigma}}(\lambda)$

$\mathcal{Q}=\emptyset$
$(vk,sk)\leftarrow Gen(1^{\lambda})$
$(Y,y)\leftarrow GenR(1^{\lambda})$
$m\leftarrow\mathcal{A}^{\mathcal{O}_{\mathrm{S}}(\cdot),\mathcal{O}_{\mathrm{pS}}(\cdot)}(vk,Y)$
$\hat{\sigma}\leftarrow pSign((vk,sk),Y,m)$
$\sigma\leftarrow\mathcal{A}^{\mathcal{O}_{\mathrm{S}}(\cdot),\mathcal{O}_{\mathrm{pS}}(\cdot)}(\hat{\sigma},Y)$
return $(m\notin\mathcal{Q}\wedge Vrfy(vk,m,\sigma))$

$\mathcal{O}_{\mathrm{S}}(m)$

$\sigma\leftarrow Sign(sk,m)$
$\mathcal{Q}=\mathcal{Q}\cup\{m\}$
return σ

$\mathcal{O}_{\mathrm{pS}}(m,Y)$

$\hat{\sigma}\leftarrow pSign((vk,sk),m,Y)$
$\mathcal{Q}=\mathcal{Q}\cup\{m\}$
return $\hat{\sigma}$

图 3 选择消息攻击下适配器签名的存在不可伪造性实验

除了提供额外的预签名谕言机 $\mathcal{O}_{\mathrm{pS}}(\cdot)$, 适配器签名的 aEUF-CMA 定义与签名的不可伪造性安全定义类似. 预签名谕言机输入消息值 m 和实例 Y 输出一个预签名值, 其中 (m,Y) 可由敌手自适应选择. 预签名谕言机是非常重要的, 因为在具体应用中适配器签名要求敌手在不知道实例 Y 的证据 y 的情况下, 即使知道预签名值 $\hat{\sigma}$ 也难以伪造消息的真实签名值 σ. aEUF-CMA 安全和预签名的可适配性一起保证了关于实例 Y 的预签名值能被适配成一个有效的签名值, 当且仅当知道实例 Y 的证据 y.

● 证据可提取性. 对于任意的 PPT 敌手 $\mathcal{A}$, 如果存在可忽略函数 $negl$, 使得在实验 $\mathrm{aWitExt}_{\mathcal{A},\Pi_{R,\Sigma}}$ 中敌手优势 $\mathrm{Adv}_{\mathcal{A},\mathrm{aWitExt}}=\Pr[\mathrm{aWitExt}_{\mathcal{A},\Pi_{R,\Sigma}}(\lambda)=1]\leqslant negl(\lambda)$, 则称适配器签名满足证据可提取性. $\mathrm{aWitExt}_{\mathcal{A},\Pi_{R,\Sigma}}$ 定义如图 4.

$\mathrm{aWitExt}_{\mathcal{A},\Pi_{R,\Sigma}}(\lambda)$

$\mathcal{Q}=\emptyset$
$(vk,sk)\leftarrow Gen(1^{\lambda})$
$(m,Y)\leftarrow\mathcal{A}^{\mathcal{O}_{\mathrm{S}}(\cdot),\mathcal{O}_{\mathrm{pS}}(\cdot)}(vk)$
$\hat{\sigma}\leftarrow pSign((vk,sk),Y,m)$
$\sigma\leftarrow\mathcal{A}^{\mathcal{O}_{\mathrm{S}}(\cdot),\mathcal{O}_{\mathrm{pS}}(\cdot)}(\hat{\sigma})$
$y'\leftarrow Ext(\sigma,\hat{\sigma},Y)$
return $(m\notin\mathcal{Q}\wedge(Y,y')\notin R$
$\wedge Vrfy(vk,m,\sigma)$

$\mathcal{O}_{\mathrm{S}}(m)$

$\sigma\leftarrow Sign(sk,m)$
$\mathcal{Q}=\mathcal{Q}\cup\{m\}$
return σ

$\mathcal{O}_{\mathrm{pS}}(m,Y)$

$\hat{\sigma}\leftarrow pSign((vk,sk),m,Y)$
$\mathcal{Q}=\mathcal{Q}\cup\{m\}$
return $\hat{\sigma}$

图 4 适配器签名的证据可提取性实验

适配器签名的证据可提取性可保证关于消息和实例 (m,Y) 的一对有效的签名值和预签名值 $(\sigma,\hat{\sigma})$ 能被用于提取出 Y 的证据 y. 实验 aWitExt 和实验 aSigForge 的重要区别: 在实验 aWitExt 中, 敌手选择挑战实例 Y, 敌手知道实例 Y 的证据 y, 可以生成关于挑战消息 m 的有效签名值. 但是这个优势不足以让敌手赢得实验 aWitExt, 因为敌手赢得实验 aWitExt 的条件是他输出的有效签名值不能泄漏实例 Y 的证据 y.

定义 1. 如果适配器签名 $\sum_{\mathrm{AS}}$ 满足预签名可适配性, aEUF-CMA 安全和证据可提取性, 则称该适配器签名 $\sum_{\mathrm{AS}}$ 是安全的.

3 SM2 适配器签名

3.1 方案构造

本节中, 我们构造 SM2 适配器签名方案, 并在随机谕言机模型下给出安全性证明. 令 (X,x) 是 SM2 签名公私

钥对, G 是群 $\mathbb{G}$ 的基点, 阶为 n. 签名方运行困难关系生成算法分别生成困难关系 R_Y^{DL} 和 R_Z^{EDL}, 并秘密保存证据 y, 令困难关系 $R_Y^{\mathrm{DL}}=\{(I_Y^{\mathrm{DL}}=(Y,\pi_Y),y)|Y=yG\wedge V_Y^{\mathrm{DL}}(I_Y^{\mathrm{DL}})\to 1\}$, 其中 $\pi_Y\leftarrow \mathrm{P}_Y^{\mathrm{DL}}(Y,y)$, $\mathrm{P}_Y^{\mathrm{DL}}$ 和 V_Y^{DL} 分别表示具有直线证据提取性的非交互零知识的知识证明 (NIZKPoK)[30]中的证明和验证算法. 令困难关系 $R_Z^{\mathrm{EDL}}=\{(I_Z^{\mathrm{EDL}}=(Y,Z,\pi_Z),y)|Y=yG\wedge Z-Y=yX\wedge V_Z^{\mathrm{EDL}}(I_Z^{\mathrm{EDL}})\to 1\}$, 其中 $\pi_Z\leftarrow \mathrm{P}_Z^{\mathrm{EDL}}(I_Z^{\mathrm{EDL}},y)$, $\mathrm{P}_Z^{\mathrm{EDL}}$ 和 V_Z^{EDL} 分别表示非交互零知识证明 (NIZK)[32]中的证明和验证算法, Z 是预签名算法的公开参数. SM2 适配器签名 SM2-AS 构造如下, 具体见图 5.

$pSign(x,m,I_Y^{\mathrm{DL}},I_Z^{\mathrm{EDL}})\to\hat{\sigma}$	$pVrfy(X,m,I_Y^{\mathrm{DL}},I_Z^{\mathrm{EDL}},\hat{\sigma})\to 0/1$	$Adapt(\hat{\sigma},y)\to\sigma$
$e=H(m)$	$e=H(m)$	$s=\hat{s}+y \bmod n$
$k\leftarrow\mathbb{Z}_n$	$\hat{K}=(\hat{s}+r)X+\hat{s}G+Z$	**return** $\sigma=(r,s)$
$r_x=f(kG+Z)$	$r'=f(\hat{K})+e \bmod n$	
$r=e+r_x \bmod n$	If $r'=r$, **return** 1,	$Ext(\sigma,\hat{\sigma},I)\to y$
$\hat{s}=(1+x)^{-1}(k-rx) \bmod n$	else, **return** 0.	$y=s-\hat{s} \bmod n$
return $\hat{\sigma}=(r,\hat{s})$		If $(I_Y^{\mathrm{DL}},y)\in R_Y^{\mathrm{DL}}$ $\wedge(I_Z^{\mathrm{EDL}},y)\in R_Z^{\mathrm{EDL}}$, **return** y, else **return** $\perp$

图 5 SM2 适配器签名

- $pSign(x,m,I_Y^{\mathrm{DL}},I_Z^{\mathrm{EDL}})\to\hat{\sigma}$. 预签名算法输入签名私钥 x, 消息值 m 和实例 I_Y^{DL}, I_Z^{EDL}, 计算 $e=H(m)$, 选择随机数 $k\leftarrow\mathbb{Z}_n$, 计算 $\hat{K}=kG+Z$, $r_x=f(\hat{K})$, $r=e+r_x \bmod n$, $\hat{s}=(1+x)^{-1}(k-rx) \bmod n$, 最后输出预签名值 $\hat{\sigma}=(r,\hat{s})$.
- $pVrfy(X,m,I_Y^{\mathrm{DL}},I_Z^{\mathrm{EDL}},\hat{\sigma})\to 0/1$. 预签名验证算法输入验证公钥 X, 消息 m, 实例 I_Y^{DL}, I_Z^{EDL}, 以及预签名值 $\hat{\sigma}=(r,\hat{s})$, 计算 $e=H(m)$, $\hat{K}=(\hat{s}+r)X+\hat{s}G+Z$, $r'=f(\hat{K})+e \bmod n$. 如果 $r'=r$, 该算法输出 1, 否则输出 0.
- $Adapt(\hat{\sigma},y)\to\sigma$. 适配算法输入预签名值 $\hat{\sigma}=(r,\hat{s})$ 和证据 y, 计算 $s=\hat{s}+y \bmod n$, 并输出签名值 $\sigma=(r,s)$.
- $Ext(\sigma,\hat{\sigma},I_Y^{\mathrm{DL}},I_Z^{\mathrm{EDL}})\to y$. 证据提取算法输入签名值 σ, 预签名值 $\hat{\sigma}$ 和实例 I_Y^{DL}, I_Z^{EDL}, 计算 $y=s-\hat{s} \bmod n$. 如果 $(I_Y^{\mathrm{DL}},y)\in R_Y^{\mathrm{DL}}$ 且 $(I_Z^{\mathrm{EDL}},y)\in R_Z^{\mathrm{EDL}}$, 输出 y, 否则输出 $\perp$.

对比 ECDSA-AS$_{\mathbf{k}}$[5]和 SM-AS$_{\mathbf{x}}$[29]. 在具体构造中, ECDSA-AS$_{\mathbf{k}}$[5] 和 SM-AS$_{\mathbf{x}}$[29]使用相同的离散对数困难关系 $R_Y^{\mathrm{DL}}=\{(I_Y^{\mathrm{DL}}=(Y,\pi_Y),y)|Y=yG\wedge V_Y^{\mathrm{DL}}(I_Y^{\mathrm{DL}})\to 1\}$, 而我们的 SM2-AS 结合了 SM2 签名结构, 在困难关系中附加了 $R_Z^{\mathrm{EDL}}=\{(I_Z^{\mathrm{EDL}}=(Y,Z,\pi_Z),y)|Y=yG\wedge Z-Y=yX\wedge V_Z^{\mathrm{EDL}}(I_Z^{\mathrm{EDL}})\to 1\}$.该部分可以离线批量计算, 并且可保证各参与方的预签名生成和验证操作非常高效. 具体区别如下: ECDSA-AS$_{\mathbf{k}}$[5]在预签名阶段, 需要以预签名的随机数 k 作为证据, 计算预签名公开参数 $K=kY$, 并使用零知识证明技术证明离散对数相等的困难关系 $R_K^{\mathrm{EDL}}=\{((\hat{K},K),k)|\exists k\in\mathbb{Z}_n,$ s.t. $\hat{K}=kG\wedge K=kY\}$ (为了简洁, 省略实例中的公开参数, 如: G, Y). SM-AS$_{\mathbf{x}}$[29]在预签名阶段, 需要以签名私钥 x 作为证据, 计算预签名公开参数 $Z=xY+Y$, 并使用零知识证明技术证明离散对数相等的困难关系 $R_X^{\mathrm{EDL}}=\{((X,Z),x)|\exists x\in\mathbb{Z}_n,$ s.t. $X=xG\wedge Z-Y=xY\}$. ECDSA-AS$_{\mathbf{k}}$[5]和 SM-AS$_{\mathbf{x}}$[29]分别使用预签名随机数和签名私钥作为证据. 因此, 该部分预签名公开参数和对应零知识证明的计算需要各参与方独自生成.

我们的 SM2-AS 结合了 SM2 签名的结构, 以困难关系的证据 y 作为证据, 将 Z 和 π_Z 的计算转移到困难关系生成阶段. 因为困难关系生成方拥有 y, 可以获得各参与方的签名验证公钥 X_i, 所以该部分可以离线批量生成, 极大地提升了预签名生成和验证效率, 减少协议中各参与方交互过程中的等待时间. 在具体应用中, 我们的 SM2-AS 需要其他参与方的公钥信息 X_i, 困难关系生成方才能为其他参与方批量生成预签名公开参数 $Z_i=y(X_i+G)$ 和对应的零知识证明 π_{Z_i}. 因为各参与方的公钥是公开信息, 所以困难关系生成方可以提前获得公钥信息, 并不影响适配器签名的应用.

预签名特殊验证场景. 根据适配器签名在区块链上的应用, 比如图 1 所示的原子交换协议, 适配器签名可分为链上和链下两个阶段. 其中, 在链下阶段, 各参与方生成和验证预签名值, 并适配出完整签名值; 在链上公布完整的签名值. 因此, 预签名值的验证并不需要所有人 (如: 链上的矿工) 验证, 只需要协议参与方验证即可. 针对这种特殊验证场景, 我们可对 SM2-AS 算法进一步优化, 减少额外的零知识证明. 根据预签名公开参数的结构 $Z_i=y(X_i+G)=(x_i+1)Y$, 各参与方 U_i 拥有签名私钥 x_i 可直接验证 Z_i 的正确性, 并不需要额外的零知识证明 π_{Z_i}. 因

此, SM2-AS 可去除附加的困难关系 R_Z^{EDL}, 预签名公开参数可以由各方本地计算, 即困难关系生成方计算预签名公开参数 $Z_i = y(X_i + G)$, 其他参与方计算预签名公开参数 $Z_i = (x_i + 1)Y$.

3.2 安全性证明

定理 1. 如果 SM2 签名 $\sum_{\mathrm{SM2}}$ 满足 SUF-CMA, R_Y^{DL} 和 R_Z^{EDL} 是困难关系, NIZKPoK 是安全的具有直线证据提取性的非交互零知识的知识证明, NIZK 是安全的非交互零知识证明, 则上述 SM2 适配器签名方案 $\Pi_{R,\sum_{\mathrm{SM2}}}$ 在随机谕言机模型下是安全的.

引理 1 (预签名可适配性). SM2 适配器签名 $\Pi_{R,\sum_{\mathrm{SM2}}}$ 满足预签名的可适配性.

证明: 对于选定的消息 $m \in \{0,1\}^*$, $e = H(m)$, 困难关系 $(I_Y^{\mathrm{DL}}, y) \in R_Y^{\mathrm{DL}}$, $(I_Z^{\mathrm{EDL}}, y) \in R_Z^{\mathrm{EDL}}$, SM2 签名验证公钥 $X \in \mathbb{G}$ 和有效的预签名值 $\hat{\sigma} = (r, \hat{s})$. 预签名值能通过预签名值验证算法 $pVrfy(X, m, I, \hat{\sigma}) \to 1$. 即:

$$\hat{K} = (\hat{s} + r)X + \hat{s}G + Z,\ r' = r'_x + e = f(\hat{K}) + e = r \bmod n.$$

根据适配算法定义, $Adapt(\hat{\sigma}, y) \to \sigma$, 其中 $\sigma = (r, s), s = \hat{s} + y \bmod n$. 因此, $K' = (s + r)X + sG$. 即 $r' = r'_x + e = f(K') + e = r \bmod n$. 即预签名值能被适配成 SM2 签名值, 并能通过 SM2 的签名验证算法.

引理 2 (预签名的正确性). SM2 适配器签名 $\Pi_{R,\sum_{\mathrm{SM2}}}$ 满足预签名的正确性.

证明: 对于任意的签名私钥和证据 $x, y \in \mathbb{Z}_n$, 消息 $m \in \{0,1\}^*$, 定义签名验证公钥和实例 $X = xG$, $Y = yG$, $Z = y(X + G)$, $\pi_Y \leftarrow \mathrm{P}_Y^{\mathrm{DL}}(Y, y)$, $I_Y^{\mathrm{DL}} = (Y, \pi_Y)$, $\pi_Z \leftarrow \mathrm{P}_Z^{\mathrm{EDL}}(I_Z^{\mathrm{EDL}}, y)$, $I_Z^{\mathrm{EDL}} = (Y, Z, \pi_Z)$. 根据预签名算法的定义 $pSign(x, m, I_Y^{\mathrm{DL}}, I_Z^{\mathrm{EDL}}) \to (r, \hat{s})$, 则有 $K = (\hat{s} + r)X + \hat{s}G$.

根据 NIZK 的完备性, 因为 $V_Z^{\mathrm{EDL}}(I_Z^{\mathrm{EDL}}) \to 1$, 所以 $Z = y(X + G)$. 因此可计算 $\hat{K} = K + Z$, $r'_x = f(\hat{K})$, $r' = r'_x + e = r \bmod n$, 即可通过预签名验证算法 $pVrfy(X, m, I_Y^{\mathrm{DL}}, I_Z^{\mathrm{EDL}}, \hat{\sigma}) \to 1$. 由于 $Adapt(\hat{\sigma}, y) \to \sigma$, 根据适配算法的定义 $s = \hat{s} + y \bmod n$, 可通过 SM2 签名的签名验证算法 $Vrfy(X, m, \sigma) \to 1$. 同时可提取正确的证据 $Ext(\sigma, \hat{\sigma}, I_Y^{\mathrm{DL}}, I_Z^{\mathrm{EDL}}) = \hat{s} + y - \hat{s} = y$.

引理 3 (aEUF-CMA 安全). 如果 SM2 签名算法 $\sum_{\mathrm{SM2}}$ 满足 SUF-CMA, R_Y^{DL} 和 R_Z^{EDL} 是困难关系, NIZKPoK 是安全的具有直线证据提取性的非交互零知识的知识证明, NIZK 是安全的非交互零知识证明, 则 SM2 适配器签名 $\Pi_{R,\sum_{\mathrm{SM2}}}$ 满足 aEUF-CMA 安全.

证明: 假设存在 PPT 敌手 $\mathcal{A}$, 在 aSigForge 实验中, 打破 SM2 适配器签名的安全性, 则可构造模拟器 $\mathcal{S}$ 打破 SM2 签名的强不可伪造性.

模拟器 $\mathcal{S}$ 可以访问 SM2 签名谕言机 $O_{\mathrm{SM2\text{-}Sign}}$ 和随机谕言机 $\mathcal{H}_{\mathrm{SM2}}$, 并且模拟 $\mathcal{A}$ 的随机谕言机 $\mathcal{H}$, 签名谕言机 O_{S} 和预签名谕言机 O_{pS}. $\mathcal{S}$ 可以通过 $O_{\mathrm{SM2\text{-}Sign}}$ 和 $\mathcal{H}_{\mathrm{SM2}}$ 直接模拟 $\mathcal{A}$ 的 O_{S} 和 $\mathcal{H}$ 的询问. $\mathcal{S}$ 模拟 $\mathcal{A}$ 的 O_{pS} 如下: 当 $\mathcal{A}$ 询问 O_{pS} 关于消息 m 和实例 $I_Y^{\mathrm{DL}}, I_Z^{\mathrm{EDL}}$ 的预签名值时, $\mathcal{S}$ 可通过 NIZKPoK 的直线证据提取器提取实例 $I_Y^{\mathrm{DL}} = (Y, \pi_Y)$ 的证据 y, 并询问 $O_{\mathrm{SM2\text{-}Sign}}$ 获得真实签名 (r, s), 然后根据适配器算法的定义, 基于证据 y 和真实签名值 (r, s) 计算预签名值 $(r, \hat{s} = s - y)$.

通过混合论证技术, 我们描述游戏 $\mathrm{Game}_0, \ldots, \mathrm{Game}_3$, 其中 Game_0 是 aSigForge 实验, Game_i 和 Game_{i+1}, $i = 0, \ldots, 2$ 是不可区分的, 且模拟器可完美地模拟 Game_3, 最后证明敌手赢得 Game_3 的概率是可忽略的.

Game_0: 该游戏与适配器签名 aSigForge 实验相同.

Game_1: 该游戏和 Game_0 的区别是当 $\mathcal{A}$ 输出真实签名值伪造 σ^* 时, 如果 $\sigma^* = Adapt(\hat{\sigma}, y)$, 则输出 $\perp$.

Game_2: 该游戏和 Game_1 的区别是在敌手询问 O_{pS} 时, Game_2 使用 NIZKPoK 的直线证据提取器提取证据 y, 并询问 $O_{\mathrm{SM2\text{-}Sign}}$ 获得完整签名值 $\sigma = (r, s)$, 计算 $\hat{s} = s - y$, 输出预签名值 $\hat{\sigma} = (r, \hat{s})$.

Game_3: 该游戏和 Game_2 的区别是在接收 $\mathcal{A}$ 输出的挑战消息 m^* 后, Game_3 生成困难关系 I_Y^{DL}, I_Z^{EDL}, 并通过 $O_{\mathrm{SM2\text{-}Sign}}$ 获得完整签名值 $\sigma^* = (r^*, s^*)$, 根据证据 y^* 计算并输出预签名值 $\hat{\sigma}^* = (r^*, \hat{s}^*)$.

构造模拟器 $\mathcal{S}$ 完美模拟 Game_3, 并利用 $\mathcal{A}$ 的能力赢得 SM2 签名的 sSigForge 实验. 具体构造如下:

- 模拟 $\mathcal{A}$ 的签名谕言机: $\mathcal{S}$ 利用自己的签名谕言机 $O_{\mathrm{SM2\text{-}Sign}}$ 回复 $\mathcal{A}$ 的询问.
- 模拟 $\mathcal{A}$ 的随机谕言机: 对于没有询问过的值, $\mathcal{S}$ 利用 $\mathcal{H}_{\mathrm{SM2}}$ 回复询问, 对于已经询问过的值, 输出与上次相同

的回复.

● 模拟$\mathcal{A}$的预签名谕言机: $\mathcal{S}$接收$\mathcal{A}$的$(m, I_Y^{\mathrm{DL}}, I_Z^{\mathrm{EDL}})$, 基于 NIZKPoK 的直线证据提取器提取证据y, 并询问自己的$O_{\mathrm{SM2\text{-}Sign}}$关于消息$m$的签名获得$(r,s)$. 然后$\mathcal{S}$计算$\hat{s}=s-y$, 并输出$\hat{\sigma}=(r,\hat{s})$.

● 挑战阶段: $\mathcal{S}$接收$\mathcal{A}$的挑战消息m^*, 运行$GenR(1^\lambda)\to(I_Y^{\mathrm{DL}}, I_Z^{\mathrm{EDL}}, y)$, 询问$O_{\mathrm{SM2\text{-}Sign}}$消息$m^*$获得$\sigma=(r,s)$, 然后计算$\hat{s}=s-y$, 输出$\hat{\sigma}^*=(r,\hat{s})$. 最后接收$\mathcal{A}$的伪造$\hat{\sigma}^*$, 并输出伪造$(m^*,\hat{\sigma}^*)$.

Claim 1. 令事件Bad_1指Game_1输出$\perp$, 则$\Pr[\mathrm{Bad}_1]\leqslant negl_1(\lambda)$.

证明: 我们将Bad_1归约到困难关系R的困难性, 即假设存在 PPT 敌手$\mathcal{A}$能使得Game_1以非可忽略的概率输出$\perp$, 则可构造模拟器$\mathcal{S}$打破R的困难性. 具体构造如下: 模拟器$\mathcal{S}$获得挑战I^*, 使用Game_1中的方式模拟$\mathcal{A}$的$\mathcal{H}$, O_{S}和O_{pS}. 当收到$\mathcal{A}$的挑战消息m时, 以$I_Y^{*\mathrm{DL}}$, $I_Z^{*\mathrm{EDL}}$作为实例计算预签名值$\hat{\sigma}$并发送给敌手. 随后, 获得敌手输出的完整签名值σ.

假设事件Bad_1发生, 即$Adapt(\hat{\sigma},y)\to\sigma$, 因此$\mathcal{S}$可以提取证据$y^*\leftarrow Ext(\sigma,\hat{\sigma},I_Y^{*\mathrm{DL}},I_Z^{*\mathrm{EDL}})$, 可获得证据$y^*$使$(I_Y^{*\mathrm{DL}},y^*)\in R_Y^{\mathrm{DL}}$且$(I_Z^{*\mathrm{EDL}},y^*)\in R_Z^{\mathrm{EDL}}$成立. 因此, $\mathcal{S}$打破困难关系的概率和Bad_1发生的概率相同, 即$\Pr[\mathrm{Bad}_1]\leqslant negl_1(\lambda)$. 因此, $|\Pr[\mathrm{Game}_0=1]-\Pr[\mathrm{Game}_1=1]|\leqslant negl_1(\lambda)$.

Claim 2. Game_1, Game_2和Game_3是不可区分的.

证明: 根据 NIZKPoK 的直线证据提取性, 基于实例I_Y^{DL}可提取证据y, 使得$V_Y^{\mathrm{DL}}(I_Y^{\mathrm{DL}})\to 1$成立. 根据$O_{\mathrm{SM2\text{-}Sign}}$可获得完整签名$\sigma=(r,s)$, 并模拟预签名的$\hat{s}=s-y$, 因此, Game_2和Game_1是不可区分的, 即$\Pr[\mathrm{Game}_2=1]=\Pr[\mathrm{Game}_1=1]$. 在$\mathrm{Game}_3$中, 模拟器自己生成困难关系, 拥有证据$y$, 同样可模拟预签名谕言机$O_{\mathrm{pS}}$, 因此, Game_3和Game_2是不可区分的, 即$\Pr[\mathrm{Game}_3=1]=\Pr[\mathrm{Game}_2=1]$.

Claim 3. (m^*,σ^*)是实验 sSigForge 中的有效伪造.

证明: 在挑战阶段之前, 敌手$\mathcal{A}$没有询问过O_{S}或者O_{pS}关于挑战消息m^*的签名值或预签名值. 因此, 关于m^*对于$O_{\mathrm{SM2\text{-}Sign}}$的询问发生在挑战阶段. 根据$\mathrm{Game}_1$可知, 敌手输出的伪造$\sigma^*$和$O_{\mathrm{SM2\text{-}Sign}}$输出的$\sigma$相同的概率是可忽略的. 因此, 关于$m^*$的询问, $O_{\mathrm{SM2\text{-}Sign}}$从没有输出过$\sigma^*$, 即$(m^*,\sigma^*)$是实验 sSigForge 中的有效伪造.

综上所述, $\mathcal{S}$可完美模拟Game_3. $\mathcal{A}$在实验 aSigForge 中的优势为:

$$\mathrm{Adv}_{\mathcal{A},\mathrm{aSigForge}}=\Pr[\mathrm{Game}_0=1]\leqslant\Pr[\mathrm{Game}_3=1]+negl(\lambda)\leqslant\mathrm{Adv}_{\mathcal{S},\mathrm{sSigForge}}+negl(\lambda).$$

因此, 上述 SM2 适配器签名满足 aEUF-CMA 安全.

引理 4 (证据可提取性). 假设 SM2 签名满足 SUF-CMA 安全, R_Y^{DL}和R_Z^{EDL}是困难关系, NIZKPoK 是安全的具有直线证据提取性的非交互零知识的知识证明, NIZK 是安全的非交互零知识证明, 则 SM2 适配器签名$\Pi_{R,\sum_{\mathrm{SM2}}}$满足证据可提取性.

证明: 假设存在 PPT 敌手$\mathcal{A}$能以非可忽略的概率赢得 SM2 适配器签名的 aWitExt 实验, 则可构造 PPT 的模拟器赢得 SM2 签名的 sSigForge 实验. 不同于实验 aSigForge, 在实验 aWitExt 中, 挑战阶段的困难关系实例也是由敌手$\mathcal{A}$生成. 因此, $\mathcal{S}$无法在挑战阶段生成证据y. 但是$\mathcal{S}$可通过 NIZKPoK 的直线证据提取器提取I_Y^{DL}中的证据y, $\mathcal{S}$同样可模拟敌手的预签名值谕言机O_{pS}.

我们定义游戏$\mathrm{Game}_0,\ldots,\mathrm{Game}_2$, 其中$\mathrm{Game}_0$是原 aWitExt 实验, 各游戏之间是不可区分的, 且模拟器可完美地模拟Game_2, 最后我们证明敌手赢得游戏Game_2的概率是可忽略的.

Game_0: 该游戏和真实的 SM2 适配器签名的 aWitExt 实验相同.

Game_1: 该游戏和Game_0的区别是在模拟O_{pS}时, 通过 NIZKPoK 的直线证据提取器提取实例I_Y^{DL}的证据, 并询问签名谕言机获得签名值$\sigma=(r,s)$, 使用证据y计算$\hat{s}=s-y$, 输出预签名值$\hat{\sigma}=(r,\hat{s})$.

Game_2: 该游戏和Game_1的区别是在挑战阶段使用 NIZKPoK 的直线证据提取器提取证据y, 并询问签名谕言机获得签名值$\sigma=(r,s)$, 使用证据y计算$\hat{s}=s-y$, 输出预签名值$\hat{\sigma}=(r,\hat{s})$.

可构造模拟器$\mathcal{S}$完美模拟Game_2, 并利用$\mathcal{A}$的能力赢得 sSigForge 实验. 具体构造如下.

● 模拟$\mathcal{A}$的签名谕言机: $\mathcal{S}$利用自己的签名谕言机$O_{\mathrm{SM2\text{-}Sign}}$回复$\mathcal{A}$的询问.

● 模拟 $\mathcal{A}$ 的随机谕言机: 对于没有询问过的值, $\mathcal{S}$ 利用 $\mathcal{H}_{\text{SM2}}$ 回复, 对于已经询问过的值, 输出与上次相同的回复.

● 模拟 $\mathcal{A}$ 的预签名谕言机: $\mathcal{S}$ 接收到 $\mathcal{A}$ 的 $(m, I_Y^{\text{DL}}, I_Z^{\text{EDL}})$, 基于 NIZKPoK 的直线证据提取器提取证据 y, 并询问谕言机 $O_{\text{SM2-Sign}}$ 关于消息 m 的签名 (r,s). 然后计算 $\hat{s}=s-y$. 最后输出 $\hat{\sigma}=(r,\hat{s})$.

● 挑战阶段: $\mathcal{S}$ 接收到 $\mathcal{A}$ 的挑战 $(m^*, I_Y^{*\text{DL}}, I_Z^{*\text{EDL}})$, 首先基于 NIZKPoK 的直线证据提取器提取证据 y, 并询问谕言机 $O_{\text{SM2-Sign}}$ 关于消息 m^* 的签名 (r,s). 然后 $\mathcal{S}$ 计算 $\hat{s}=s-y$, 输出 $\hat{\sigma}=(r,\hat{s})$. 最后接收 $\mathcal{A}$ 的伪造 σ^*, 并输出伪造 (m^*,σ^*).

Claim 4. 各个游戏 $\text{Game}_0, \text{Game}_1, \text{Game}_2$ 是不可区分的.

证明: 根据 NIZKPoK 的直线证据提取性, 模拟器可根据实例 I_Y^{DL} 在随机谕言机模型下提取证据 y, 使得 $V_Y^{\text{DL}}(I_Y^{\text{DL}}) \to 1$, 并询问签名谕言机获得签名值 $\sigma=(r,s)$, 使用证据 y 计算 $\hat{s}=s-y$, 输出预签名值 $\hat{\sigma}=(r,\hat{s})$. 因此, Game_1 和 Game_0 是不可区分的, 即 $|\Pr[\text{Game}_0=1]-\Pr[\text{Game}_1=1]| \leqslant negl_1(\lambda)$. 同理, 在挑战阶段上述性质依旧满足, Game_2 和 Game_1 是不可区分的, 即 $|\Pr[\text{Game}_1=1]-\Pr[\text{Game}_2=1]| \leqslant negl_2(\lambda)$.

Claim 5. (m^*,σ^*) 在 sSigForge 实验中是一个有效的伪造.

证明: 在挑战阶段之前, 敌手 $\mathcal{A}$ 没有询问过 O_{S} 或者 O_{pS} 关于挑战消息 m^* 的签名值或预签名值. 关于 m^* 对于 $O_{\text{SM2-Sign}}$ 的询问发生在挑战阶段. 如果在挑战阶段敌手输出的 SM2 签名伪造 $*$ 和模拟器询问 $O_{\text{SM2-Sign}}$ 获得的 σ 相同, 则可提取证据 $y=s-\hat{s}$ 使得 $(I_Y^{\text{DL}}, y) \in R_Y^{\text{DL}}$ 且 $(I_Z^{\text{EDL}}, y) \in R_Z^{\text{EDL}}$ 成立. 因此, 关于 m^* 的询问, $O_{\text{SM2-Sign}}$ 没有输出过 σ^*, 即 $(m^*,\hat{\sigma}^*)$ 是实验 sSigForge 中的有效伪造.

综上所述, $\mathcal{S}$ 可完美模拟 Game_2. $\mathcal{A}$ 在实验 aWitExt 中的优势为:

$$\text{Adv}_{\mathcal{A},\text{aWitExt}} = \Pr[\text{Game}_0=1] \leqslant \Pr[\text{Game}_2=1]+negl(\lambda) \leqslant \text{Adv}_{\mathcal{S},\text{sSigForge}}+negl(\lambda).$$

因此, 上述 SM2 适配器签名满足证据可提取性.

4 分布式 SM2 适配器签名

本节中, 我们根据 SM2-AS 的结构, 基于 SM2 协同签名算法构造分布式 SM2 适配器签名. SM2-AS 的预签名操作类似 SM2 签名操作, 只在计算随机点的步骤中有所区别: 在 SM2 签名中随机点 $K=kG$, 而在 SM2 适配器签名中随机点 $\hat{K}=kG+Z$. 因此, 对于 SM2 适配器签名算法的分布式扩展类似分布式 SM2 签名的构造, 只需要在协同计算随机点的过程中, 将原分布式 SM2 签名中协同计算 kG, 变换成计算 $kG+Z$. 因为困难实例 Z 是公开可验证的信息, 所以各参与方可沿用 SM2 协同签名过程计算 kG, 然后加上 Z. 特别的, 该转换对于底层的分布式 SM2 签名算法并无结构要求. 因此, 分布式 SM2 适配器签名可基于任意分布式 SM2 签名 [34−39] 构造.

对于具体协议的构造, 我们基于涂彬彬等人 [34] 给出分布式 SM2 签名, 设计分布式 SM2 适配器签名. 同理, 基于其他 SM2 协同签名构造分布式 SM2 适配器签名方法类似. 首先考虑两方场景, 基于两方 SM2 签名设计两方 SM2 适配器签名, 然后基于 Shamir 的秘密分享技术, 将两方场景扩展至门限场景.

在两方 SM2 适配器签名中, 参与方 U_1 和 U_2 分别选择随机值 x_1 和 x_2 作为 SM2 签名私钥 x 的分享, 选择随机值 k_1 和 k_2 作为随机值 k 的分享, 然后协同生成预签名值 $(r, \hat{s}=(1+x)^{-1}\cdot(k-r\cdot x) \bmod n)$, 其中 $x=x_1+x_2 \bmod n$, $k=k_1+k_2 \bmod n$, $r=H(m)+r_x \bmod n$, $(r_x,r_y)=kG+Z$. 适配器签名的预签名算法构造如下.

---

1. 参与方 U_1 拥有分享 k_1, 参与方 U_2 拥有分享 k_2, 双方分别公开 k_1G 和 k_2G, 可计算 $(r_x,r_y)=k_1G+k_2G+Z=kG+Z$, $r=H(m)+r_x \bmod n$.

2. 双方各自计算 $(1+x)$ 和 $(k-r\cdot x)$ 的分享 $[(1+x)]_1$, $[(k-r\cdot x)]_1$ 和 $[(1+x)]_2$, $[(k-r\cdot x)]_2$, 然后以 $[(1+x)]_1$, $[(1+x)]_2$ 作为输入, 运行协同求逆运算协议, 分别获得 $(1+x)^{-1}$ 的分享 $[(1+x)^{-1}]_1$ 和 $[(1+x)^{-1}]_2$, 即 $(1+x)^{-1}=[(1+x)^{-1}]_1+[(1+x)^{-1}]_2$.

3. U_1 输入 $[(1+x)^{-1}]_1$ 和 $[(k-r\cdot x)]_1$, U_2 输入 $[(1+x)^{-1}]_2$, $[(k-r\cdot x)]_2$, 运行协同乘法运算协议, 分别获得预签名 $\hat{s}=(1+x)^{-1}\cdot(k-r\cdot x)$ 的分享 $[\hat{s}]_1$ 和 $[\hat{s}]_2$. 即双方公布预签名分享, 可协同生成预签名 $\hat{s}=[\hat{s}]_1+[\hat{s}]_2 \bmod n$.

其中, 协同求逆运算和协同乘法运算协议可参考文献 [34]. 根据上述构造可知, 两方 SM2 适配器签名与两方 SM2 签名的构造区别只在预签名算法步骤 1 中计算随机点的不同. 即由协同计算 kG, 变换成计算 $kG+Z$. 因为困难实例 Z 是公开可验证的, 所以该步骤的变换不影响具体协议的安全性和可扩展性. 因此, 两方 SM2 适配器签名可基于两方 SM2 签名算法的优化思想和门限扩展方法, 给出更加高效的实现, 以及门限化的构造.

5 性能分析

5.1 理论分析

本节分别在计算效率和通信成本两个方面对 SM2-AS, ECDSA-AS$_k$[5,26] 和 SM-AS$_x$[29]进行对比分析, 具体结果可见表 1. 在进行计算效率对比时, 我们主要考虑了较为复杂的点乘运算, 忽略了其他较为简单的运算, 如点加运算等. Moreno-Sanchez 等人[26]提出了首个 ECDSA-AS 构造并未给出安全性证明. 随后 Aumayr 等人[5]在文献 [26] 基础上, 提出了自证明结构 $(I_Y^{\mathrm{DL}}, y) \in R_Y^{\mathrm{DL}}$, 并给出可证明安全的 ECDSA-AS$_k$. 彭聪等人[29]使用相同的自证明结构给出 SM2 适配器签名方案. 然而, 上述 ECDSA-AS$_k$[5,26] 和 SM-AS$_x$[29]在预签名过程中, 基于预签名随机数/签名私钥生成预签名公开参数及对应的零知识证明, 需要 4 次点乘操作, 预签名验证也需要对该零知识证明进行验证, 需要 6 次点乘操作. 此外, 预签名值也需要包含预签名公开参数以及对应的零知识证明, 即需要 4 个 $\mathbb{Z}_n$ 群元素和 1 个群 $\mathbb{G}$ 中的点.

表 1 通信成本与计算效率对比

方案	预签名公钥尺寸	预签名私钥尺寸	预签名尺寸	预签名计算	预签名验证	离线批量零知识证明	是否可证明安全
ECDSA-AS$_k$[26]	$\lvert\mathbb{G}\rvert$	$\lvert\mathbb{Z}_n\rvert$	$4\lvert\mathbb{Z}_n\rvert+\lvert\mathbb{G}\rvert$	4Exp	6Exp	×	?
ECDSA-AS$_k$[5]	$\lvert\mathbb{G}\rvert$	$\lvert\mathbb{Z}_n\rvert$	$4\lvert\mathbb{Z}_n\rvert+\lvert\mathbb{G}\rvert$	4Exp	6Exp	×	√
SM2-AS$_x$[29]	$\lvert\mathbb{G}\rvert$	$\lvert\mathbb{Z}_n\rvert$	$4\lvert\mathbb{Z}_n\rvert+\lvert\mathbb{G}\rvert$	4Exp	6Exp	×	√
Our SM2-AS	$\lvert\mathbb{G}\rvert$	$\lvert\mathbb{Z}_n\rvert$	$2\lvert\mathbb{Z}_n\rvert$	1Exp	2Exp	√	√

注: $\lvert\mathbb{G}\rvert$ 表示椭圆曲线上点的尺寸, $\lvert\mathbb{Z}_n\rvert$表示 $\mathbb{Z}_n$ 群元素的尺寸, Exp表示椭圆曲线上点乘运算, ?表示未知

为了保证可证明安全, 我们沿用自证明结构[5] $(I_Y^{\mathrm{DL}}, y) \in R_Y^{\mathrm{DL}}$ 构造 SM2 适配器签名, 同时结合 SM2 签名的结构, 附加困难关系 $(I_Z^{\mathrm{EDL}}, y) \in R_Z^{\mathrm{EDL}}$. 尽管附加困难关系 R_Z^{EDL} 会增加困难关系生成方的计算量, 但是该操作可由困难关系生成方离线批量生成. 得益于附加困难关系 R_Z^{EDL}, 我们的 SM2-AS 可避免各参与方在预签名过程中生成预签名公开参数以及对应的零知识证明, 极大提升了预签名生成、传输和验证时间. 我们的 SM2-AS 的预签名操作类似 SM2 签名运算, 只需要 1 次点乘操作, 预签名验证只需要 2 次点乘操作; 预签名值只包含 2 个 $\mathbb{Z}_n$ 群元素.

5.2 实验分析

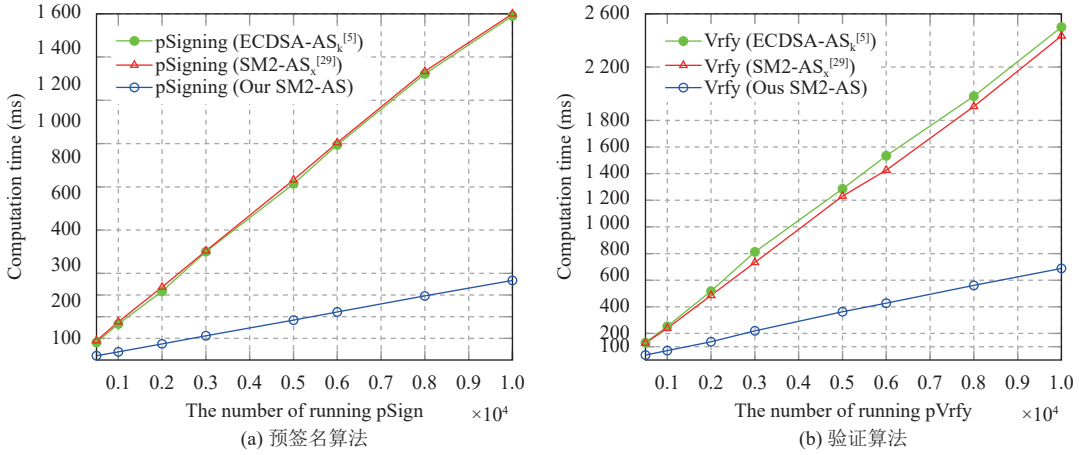
为了评估适配器签名方案的实际性能, 我们基于 OpenSSL 库实现了 SM2-AS、ECDSA-AS$_k$[5] 和 SM-AS$_x$[29], 具体实现在 GitHub 上发布: https://github.com/tbb-tobebetter/SM2-adaptor-signature. 执行环境为: Intel Core i5 CPU 2.3 GHz, 8 GB RAM, macOS High Sierra 10.13.3 system.

我们主要对比预签名算法和预签名验证算法的计算效率, 具体如图 6 所示. 我们分别运行各方案的程序 1 000 次, 发现 ECDSA-AS$_k$[5] , SM-AS$_x$[29]和我们的 SM2-AS 在预签名操作单次平均耗时分别为 166.54 μs, 176.27 μs, 37.43 μs; 预签名验证操作单次平均耗时分别为 252.86 μs, 238.46 μs, 70.76 μs. 我们的 SM2-AS 预签名计算耗时约为 ECDSA-AS$_k$[5] 和 SM-AS$_x$[29]的 1/4, 预签名验证耗时约为 ECDSA-AS$_k$[5] 和 SM-AS$_x$[29]的 1/3, 实验结果符合上述理论分析.

6 应 用

在本节中, 我们针对现有原子交换协议难以高效适用于一对多的交换场景的问题, 基于 SM2 适配器签名给出

批量原子交换协议安全高效的构造. 随后, 根据现有适配器签名构造的支付通道网络[5,13,28], 我们基于 SM2 适配器签名给出具体的实例化, 相较于基于现有 ECDSA-AS$_k$[5] 和 SM-AS$_x$[29]构造的协议更加高效.



(a) 预签名算法 (b) 验证算法

图 6 预签名算法和验证算法效率对比

6.1 批量原子交换协议

原子交换协议 (atomic swap) 能在无可信第三方的情况下公平地实现不同密码货币之间的交换. 即用户 U_0 和 U_1 可公平地跨链交换不同的密码货币 c_0 和 c_1, 其中公平性体现在双方完成交换或者双方都交换失败. 最初, 原子交换协议[15]基于哈希函数和时间锁 (time-lock) 设计, 可实现拥有哈希函数原像才能提取货币的功能, 但该方法对底层区块链脚本结构有限制, 要求密码货币的脚本语言都支持哈希原像条件脚本 (preimage conditioned scripts). 随后, Poelstra [16]使用困难关系 (嵌入签名算法) 代替哈希条件, 基于适配器签名构造原子交换协议, 避免了区块链脚本限制, 减少了链上的存储和验证运算. 具体的协议介绍如图 1 所示. 最近, Esgin 等人[28]给出了基于格的适配器签名, 并以此为基础实例化了抗量子安全的原子交换协议; Thyagarajan 等人[19]考虑通用原子交换协议的理论构造框架, 可兼容任意区块链结构, 实现两个用户间多货币的原子交换, 但是该框架效率较低难以实际应用. 此外, 他们在支持 ECDSA/Schnorr 签名的区块链结构 (非通用) 上, 基于 ECDSA/Schnorr 适配器签名[5]给出原子交换协议的高效实例化.

目前原子交换协议[15,16,19,28]主要关注两方公平交换的功能实现, 但是在面向交易量较大的应用场景, 如交易所场景等, 需要批量向多数用户同时进行交换, 或者单一参与方同时向多个账户 (地址) 进行交换的场景, 两方的原子交换协议效率较低. 因此, 在一对一原子交换协议的基础上, 我们考虑批量原子交换协议. 该协议可实现单方 U_0 批量向多方 $U_i, i\in[1,n]$ 同时进行跨链货币交换, 即发起方 U_0 (困难关系生成方) 向多个参与方 U_i 传输货币 c_{0i}, 而每个参与方 U_i 向 U_0 传输货币 c_{i0}. 首先 U_0 和 U_i 先对待交换的货币使用时间锁绑定, 留足够的时间完成交换 (类似一对一原子交换协议, 所有参与方需要使用时间锁锁定待交换的货币. 因为 U_0 先提取 U_i 的货币 c_{i0}, 所以为了留给 U_i 足够的时间提取 c_{0i} 完成交换, c_{0i} 的时间锁要比 c_{i0} 更久). U_0 生成 SM2 签名公私钥对 (X_0,x_0), 将公钥 X_0 传输给 U_i, 秘密保存签名私钥 x_0; U_i 生成 SM2 签名公私钥对 (X_i,x_i), 将公钥 X_i 传输给 U_0, 秘密保存签名私钥 x_i. 协议构造如下, 具体如图 7 所示.

● 协议建立阶段: U_0 运行困难关系生成算法生成两个困难关系 $(I_Y^{\mathrm{DL}},y)\in R_Y^{\mathrm{DL}}$ 和 $(I_{Z_0}^{\mathrm{EDL}},y)\in R_Z^{\mathrm{EDL}}$, 其中 $I_Y^{\mathrm{DL}}=(Y=yG,\pi_Y\leftarrow \mathrm{P}_Y^{\mathrm{DL}}(Y,y))$, $I_{Z_0}^{\mathrm{EDL}}=(Y,Z_0=yX_0+Y,\pi_{Z_0}\leftarrow \mathrm{P}_Z^{\mathrm{EDL}}(Y,Z_0,y))$. 生成关于传输货币 c_{0i} 给 U_i 的交易 tx_{0i} 的预签名 $\hat{\sigma}_{0i}\leftarrow pSign((X_0,x_0),tx_{0i},I_Y^{\mathrm{DL}},I_{Z_0}^{\mathrm{EDL}})$, $i\in[1,n]$; 将困难实例、预签名值和交易信息 $(I_Y^{\mathrm{DL}},I_{Z_0}^{\mathrm{EDL}},\hat{\sigma}_{0i},tx_{0i})$ 发送给 U_i, 秘密保存证据 y; U_i 可验证实例 I_Y^{DL}, $I_{Z_0}^{\mathrm{EDL}}$ 和预签名值 $\hat{\sigma}_{0i}$, 即如果 $V_Y^{\mathrm{DL}}(I_Y^{\mathrm{DL}})\to 0$, $V_Z^{\mathrm{EDL}}(I_{Z_0}^{\mathrm{EDL}})\to 0$ 或者 $pVrfy(X_0,tx_{0i},I_Y^{\mathrm{DL}},I_{Z_0}^{\mathrm{EDL}},\hat{\sigma}_{0i})\to 0$, 则拒绝交易; 否则, 计算预签名公开参数 $Z_i=(x_i+1)Y$, 并使用预签名算法生成传输货币 c_{i0} 给 U_0 的交易 tx_{i0} 的预签名值 $\hat{\sigma}_{i0}\leftarrow pSign((X_i,x_i),tx_{i0},Y,Z_i)$; 将预签名值和交易信息 $(\hat{\sigma}_{i0},tx_{i0})$ 发送给 U_0.

●交换阶段: U_0 可使用困难关系的证据 y 和对应参与方的公钥 X_i 生成预签名公开参数 $Z_i = y(X_i + G)$, $i \in [1,n]$, 并验证预签名值的正确性, 如果 $pVrfy(X_i, tx_{i0}, Y, Z_i, \hat{\sigma}_{i0}) \to 0$, 则拒绝交易 ($U_0$ 必须先验证所有预签名值的正确性, 全部验证通过后才能进行适配操作, 并提交完整签名获得待交换的货币 c_{i0}. 因为同一批次使用相同的困难关系, 所以公布任意一个完整签名相当于公布了证据 y), 否则根据证据 y, 运行适配算法 $\sigma_{i0} \leftarrow Adapt(\hat{\sigma}_{i0}, y)$ 将 U_i 的预签名适配成 SM2 签名, 即获得 U_i 传输货币 c_{i0} 给 U_0 的 SM2 签名值 σ_{i0}, 在链上公布 σ_{i0} 可获得 c_{i0}; U_i 获得签名值 σ_{i0}, 运行提取算法可计算证据 $y \leftarrow Ext(\sigma_{i0}, \hat{\sigma}_{i0}, I_Y^{\mathrm{DL}}, I_{Z_0}^{\mathrm{EDL}})$, 并通过适配算法 $\sigma_{0i} \leftarrow Adapt\ (\hat{\sigma}_{0i}, y)$, 计算 U_0 传输货币 c_{0i} 给 U_i 的 SM2 签名值 σ_{0i}, 在链上公布 σ_{0i} 获得 c_{0i}, 完成公平交换.

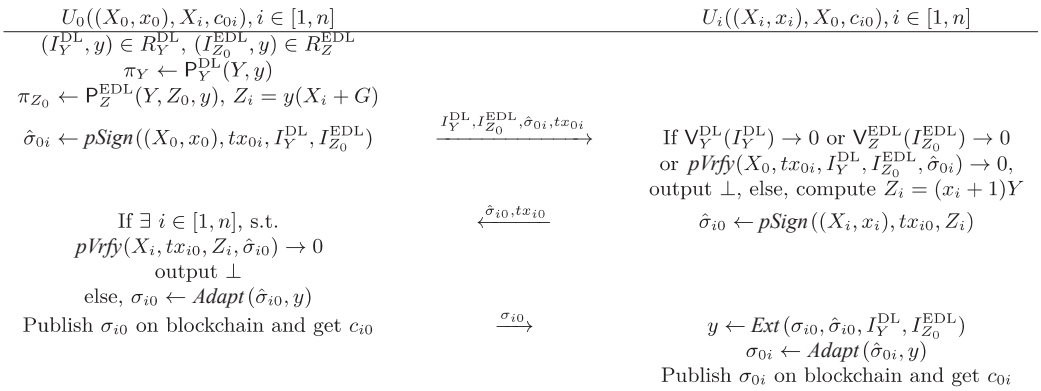


图 7 基于 SM2 适配器签名的批量原子交换协议

相比于 ECDSA-AS$_k$[5]和 SM-AS$_x$[29], 我们的 SM2-AS 更加适用于批量原子交换协议. 考虑适配器签名的特殊验证场景, 即预签名值只需要协议参与方验证. 困难关系生成方 U_0 和其他参与方 U_i, $i \in [1,n]$ 可生成预签名公开参数 $Z_i = y(X_i + G) = (x_i + 1)Y$, 不需要使用零知识证明进行验证. 因此, 各参与方 U_i 的预签名生成操作更加高效, U_0 对于 U_i 的预签名验证更加高效. 此外, U_0 在预签名过程中, 同一批 (相同证据 y) 对应相同的预签名公开参数 $Z_0 = y(X_0 + G)$ 和零知识证明 π_{Z_0}, 因此, Z_0 和 π_{Z_0} 计算一次即可. 但是对于 ECDSA-AS$_k$[5], 同一批每一个预签名都对应不同的预签名公开参数 $K = k_i Y$, 导致 U_0 在给每个 U_i 计算预签名时, 需要重新选择随机数, 重新计算预签名公开参数和零知识证明, 并传输给 U_i. 总体而言, 关于 1 对 n 的批量原子交换协议, 基于 ECDSA-AS$_k$[5]构造需要 $2n$ 个预签名公开参数和 $2n$ 个对应的零知识证明; 基于 SM-AS$_x$[29]构造需要 $n+1$ 个预签名公开参数和 $n+1$ 个对应的零知识证明; 基于我们的 SM2-AS 构造只需要 $n+1$ 个预签名公开参数和 1 个对应的零知识证明.

6.2 支付通道网络

支付通道网络 (payment channel network, PCN) [5,11,13]通过将部分交易转移到链下来提高吞吐量, 是目前解决区块链可扩展性差, 吞吐量低的主流方案. 在 PCN 中, 交易双方可将货币锁定在一个通道中可以进行即时和任意多次的交易. PCN 允许各方进行多跳支付, 即没有直接通道的参与方可以使用中介节点的通道来实现支付. 在多跳支付中, 各方需要同步路线上每个通道信息, 要求所有通道都同步更新, 或者都不更新. 目前, 最受欢迎的 PCN 应用是基于比特币的闪电网络[8]. 闪电网络通过使用哈希时间锁合约 (hash time-lock contract, HTLC) 实现上述要求. 然而, Malavolta 等人[13]提出虫洞攻击打破了 HTLC 机制的安全性, 并基于适配器签名弥补了上述漏洞, 构造匿名多跳锁 (anonymous multi-hop lock, AMHL) 构建了安全的 PCN.

在现有基于 SM2 签名算法的区块链应用中, 我们可使用 SM2 的适配器签名, 构造 AMHL 实现 PCN 的多跳支付. 具体应用中, 发送者 S (或 U_0) 通过中介节点 $U_1, \ldots, U_{k-1}$ 向接收方 R (或 U_k) 进行支付, 为了简洁, 我们省略中介节点的费用, 令 (X_j, x_j) 是 U_j 的 SM2 签名公私钥对, G 是群 $\mathbb{G}$ 的基点, 阶为 n, tx_j 是指 U_j 到 U_{j+1} 的交易. 各参与方 U_j 将公钥 X_j 发送给 U_{j+1}, 秘密保存自己的签名私钥 x_j; 发送方 U_0 (困难关系生成方) 为相邻的两个中介节点批量生成困难关系. 具体协议如图 8 所示.

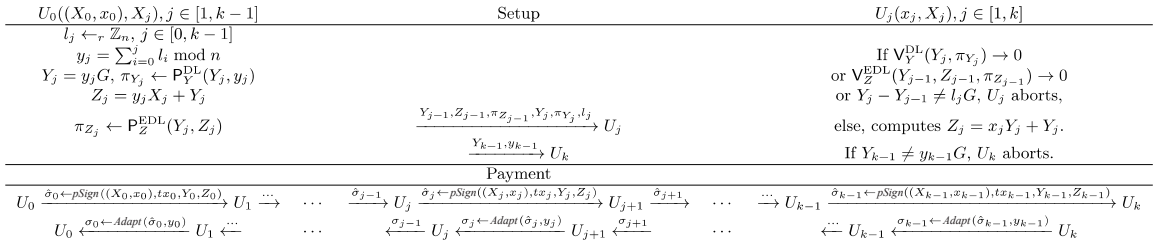


图 8　基于 SM2 适配器签名的多跳支付协议

● 建立阶段: U_0选择随机数 $l_j\leftarrow\mathbb{Z}_n, j=0,\ldots,k-1$, 然后分别计算 $y_j=\sum_{i=0}^{j} l_i \bmod n$, $Y_j=y_jG$, $\pi_{Y_j}\leftarrow \mathrm{P}_Y^{\mathrm{DL}}(Y_j,y_j)$, 计算相邻中介节点的预签名公开参数 $Z_j=y_jX_j+Y_j,\pi_{Z_j}\leftarrow \mathrm{P}_Z^{\mathrm{EDL}}(Y_j,Z_j,y_j)$. U_0 将困难关系的实例和证据关系 $(Y_{j-1},Z_{j-1},\pi_{Z_{j-1}},Y_j,\pi_{Y_j},l_j)$ 发送给 $U_j, j\in[1,k-1]$, 发送 (Y_{k-1},y_{k-1}) 给 U_k . $U_j, j\in[1,k-1]$ 可验证传输值的正确性: 如果 $V_Y^{\mathrm{DL}}(Y_j,\pi_{Y_j})\to 0$ 或者 $V_Z^{\mathrm{EDL}}(Y_{j-1},Z_{j-1},\pi_{Z_{j-1}})\to 0$ 或者 $Y_j-Y_{j-1}\neq l_jG$, 拒绝交易, 否则计算预签名公开参数 $Z_j=x_jY_j+Y_j$. U_k 可验证 $Y_{k-1}=y_{k-1}G$, 否则拒绝交易.

● 支付阶段: U_0 使用自己的签名私钥 x_0 计算预签名 $\hat{\sigma}_0\leftarrow pSign((X_0,x_0),tx_0,Y_0,Z_0)$ 发送给 U_1 , U_1 使用 U_0 的签名验证公钥 X_0 , 困难实例 Y_0 和预签名公开参数 Z_0 , 验证预签名值 $pVrfy(X_0,tx_0,Y_0,Z_0,\hat{\sigma}_0)\to 1$, 否则拒绝. U_j 从 U_{j-1} 收到预签名 $\hat{\sigma}_{j-1}$ 并验证正确后, 计算预签名 $\hat{\sigma}_j\leftarrow pSign((X_j,x_j),tx_j,Y_j,Z_j)$, $j=1,\ldots,k-1$ 发送给 U_{j+1} . 所有预签名传输和验证完成, 即 U_k 获得预签名 $\hat{\sigma}_{k-1}$, U_j 获得预签名 $\hat{\sigma}_{j-1}$. U_k 根据证据 y_{k-1} 可将预签名值 $\hat{\sigma}_{k-1}$ 适配成 SM2 签名值 $\sigma_{k-1}\leftarrow Adapt(\hat{\sigma}_{k-1},y_{k-1})$ 获得 U_{k-1} 的支付. U_k 发送 $\hat{\sigma}_{k-1}$ 给 U_{k-1} ; U_{k-1} 运行提取算法提取证据 $y_{k-1}\leftarrow Ext(\sigma_{k-1},\hat{\sigma}_{k-1},(Y_{k-1},\pi_{Y_{k-1}}))$, 计算 $y_{k-2}=y_{k-1}-l_{k-1}$, 运行适配算法计算 SM2 签名值 $\sigma_{k-2}\leftarrow Adapt(\hat{\sigma}_{k-2},y_{k-2})$ 获得 U_{k-2} 的支付, 并将 $\hat{\sigma}_{k-2}$ 发送给 U_{k-2} . 如此, U_j 获得 SM2 签名 σ_j 可通过提取算法获得 $y_j\leftarrow Ext(\sigma_j,\hat{\sigma}_j,(Y_j,\pi_{Y_j}))$, 并计算出 $y_{j-1}=y_j-l_j$, 然后通过适配算法计算 SM2 签名 $\sigma_{j-1}\leftarrow Adapt(\hat{\sigma}_{j-1},y_{j-1})$ 获得 U_{j-1} 的支付, 并将 σ_{j-1} 发送给 U_{j-1} , $j=k-1,k-2,\ldots,1$. 最后, U_1 可计算 SM2 签名 σ_0 获得 U_0 的支付.

基于 ECDSA-AS$_k$[5]和 SM-AS$_x$[29]构造的多跳支付协议, 中介节点在生成和验证预签名时, 需要额外地生成和验证预签名公开参数和对应的零知识证明. 基于 SM2-AS 构造的多跳支付协议可由发送方 U_0 作为困难关系生成方, 在协议建立 (setup) 阶段离线批量地为各中介节点 U_j 生成预签名公开参数 Z_j 和对应的零知识证明 π_{Z_j}, 避免了各中介节点 U_j 额外计算 (Z_j,π_{Z_j}), 使得在线支付 (Payment) 阶段更加高效. 根据上述多跳支付协议可知, 各参与方串行生成和验证预签名值. 因此, 相较于 ECDSA-AS$_k$[5]和 SM-AS$_x$[29], 我们的 SM2-AS 更适用于上述多跳支付协议, 特别是中介节点较多的情况.

7 总　结

针对现有区块链技术的可扩展性差、交易吞吐量低等问题, 同时秉承积极推广国密算法应用的理念, 我们基于自证明结构设计了可证明安全的 SM2 适配器签名方案. 该方案与现有的 ECDSA 适配器签名[5]和 SM2 适配器签名[29]相比, 通过结合 SM2 签名结构, 预签名公开参数和对应的零知识证明可由困难关系生成方离线批量生成, 不需要预签名方在预签名阶段单独计算, 计算效率更高, 应用优势更强. 此外, 我们基于现有 SM2 协同签名的构造方法, 给出了分布式 SM2 适配器签名方案的构造. 最后, 在现有适配器签名应用的基础上, 我们给出了 SM2 适配器签名在区块链上的具体应用, 比如: (批量) 原子交换协议和支付通道网络, 为后续 SM2 签名算法的应用推广提供参考.

References:

[1] Nakamoto S. 2008. Bitcoin: A peer-to-peer electronic cash system. https://nakamotoinstitute.org/bitcoin/

[2] Bano S, Sonnino A, Al-Bassam M, Azouvi S, McCorry P, Meiklejohn S, Danezis G. SoK: Consensus in the age of blockchains. In: Proc.

of the 1st ACM Conf. on Advances in Financial Technologies. Zurich: ACM, 2019. 183–198. [doi: 10.1145/3318041.3355458]

[3] Gudgeon L, Moreno-Sanchez, P, Roos S, McCorry P, Gervais A. SoK: Layer-two blockchain protocols. In: Proc. of the 24th Int'l Conf. on Financial Cryptography and Data Security. Kota Kinabalu: Springer, 2020. 201–226. [doi: 10.1007/978-3-030-51280-4_12]

[4] Zamyatin A, Al-Bassam M, Zindros D, Kokoris-Kogias E, Moreno-Sanchez P, Kiayias A, Knottenbelt WJ. SoK: Communication across distributed ledgers. In: Proc. of the 25th Int'l Conf. on Financial Cryptography and Data Security. Springer, 2021. 3–36. [doi: 10.1007/978-3-662-64331-0_1]

[5] Aumayr L, Ersoy O, Erwig A, Faust S, Hostáková K, Maffei M, Moreno-Sanchez P, Riahi S. Generalized channels from limited blockchain scripts and adaptor signatures. In: Proc. of the 27th Int'l Conf. on the Theory and Application of Cryptology and Information Security. Singapore: Springer, 2021. 635–664. [doi: 10.1007/978-3-030-92075-3_22]

[6] Shan JY, Gao S. Research progress on theory of blockchains. Journal of Cryptologic Research, 2018, 5(5): 484–500 (in Chinese with English abstract). [doi: 10.13868/j.cnki.jcr.000258]

[7] Bitcoin Wiki: Payment channels. 2018. https://en.bitcoin.it/wiki/Payment-_channels

[8] Poon J, Dryja T. The bitcoin lightning network: Scalable off-chain instant payments. 2016. https://lightning.network/lightning-network-paper.pdf

[9] Raiden Team. Raiden network 3.0.1 documentation. 2022. https://raiden-network.readthedocs.io/en/latest/

[10] Poelstra A. Lightning in scriptless scripts. mimblewimble team mailing list archive. 2017. https://lists.launchpad.net/mimblewimble/msg00086.html

[11] Decker C, Wattenhofer R. A fast and scalable payment network with bitcoin duplex micropayment channels. In: Pelc A, Schwarzmann AA, eds. Proc. of the 17th Int'l Symp. on Stabilization, Safety, and Security of Distributed Systems. Edmonton: Springer, 2015. 3–18. [doi: 10.1007/978-3-319-21741-3_1]

[12] Eckey L, Faust S, Hostáková K, Roos S. Splitting payments locally while routing interdimensionally. Cryptology ePrint Archive, 2020.

[13] Malavolta G, Moreno-Sanchez P, Schneidewind C, Kate A, Maffei M. Anonymous multi-hop locks for blockchain scalability and interoperability. In: Proc. of the 2019 Network and Distributed Systems Security (NDSS) Symp. San Diego: The Internet Society, 2019. [doi: 10.14722/ndss.2019.23330]

[14] Miller A, Bentov I, Bakshi S, Kumaresan R, McCorry P. Sprites and state channels: Payment networks that go faster than lightning. In: Goldberg I, Moore T, eds. Proc. of the 23rd Int'l Conf. on Financial Cryptography and Data Security. Frigate Bay, St. Kitts and Nevis: Springer, 2019. 508–526. [doi: 10.1007/978-3-030-32101-7_30]

[15] Nolan T. Alt chains and atomic transfers. https://bitcointalk.org/index.php?topic=193281.msg2224949#msg2224949

[16] Poelstra A. Adaptor signatures and atomic swaps from scriptless scripts. 2017. https://github.com/ElementsProject/scriptless-scripts/tree/master/slides/2017-05-milan-meetup

[17] Deshpande A, Herlihy M. Privacy-preserving cross-chain atomic swaps. In: Proc. of the 2020 Financial Cryptography and Data Security. Kota Kinabalu: Springer, 2020. 540–549. [doi: 10.1007/978-3-030-54455-3_38]

[18] Gugger J. Bitcoin-monero cross-chain atomic swap. Cryptology ePrint Archive, 2020.

[19] Thyagarajan SAK, Malavolta G, Moreno-Sanchez P. Universal atomic swaps: Secure exchange of coins across all blockchains. In: Proc. of the 2022 IEEE Symp. on Security and Privacy. San Francisco: IEEE, 2022. 1299–1316. [doi: 10.1109/SP46214.2022.9833731]

[20] GB/T 32918.2-2016 Public key cryptographic algorithm SM2 based on elliptic curves—Part 2: Digital signature algorithm. 2016. https://std.samr.gov.cn/gb/search/gbDetailed?id=71F772D811F7D3A7E05397BE0A0AB82A

[21] Wang ZH, Zhang ZF. Overview on public key cryptographic algorithm SM2 based on elliptic curves. Journal of Information Security Research, 2016, 2(11): 972–982 (in Chinese with English abstract).

[22] Schnorr CP. Efficient identification and signatures for smart cards. In: Proc. of the 1989 Advances in Cryptology. New York: Springer, 1990. 239–252. [doi: 10.1007/0-387-34805-0_22]

[23] Erwig A, Faust S, Hostáková K, Maitra M, Riahi S. Two-party adaptor signatures from identification schemes. In: Proc. of the 24th IACR Int'l Conf. on Practice and Theory of Public Key Cryptography. Springer, 2021. 451–480. [doi: 10.1007/978-3-030-75245-3_17]

[24] American National Standards Institute. X9.62: Public key cryptography for the financial services industry: The elliptic curve digital signature algorithm, 2005. https://standards.globalspec.com/std/1955141/ANSI%20X9.62#:~:text=Public%20Key%20Cryptography%20for%20the%20Financial%20Services%20Industry%3A,using%20the%20Elliptic%20Curve%20Digital%20Signature%20Algorithm%20%28ECDSA%29

[25] Hoffman P, Wijngaards WCA. Elliptic curve digital signature algorithm (DSA) for DNSSEC. RFC 6605. 2012. https://www.rfc-editor.org/rfc/pdfrfc/rfc6605.txt.pdf

[26] Moreno-Sanchez P, Kate A. Scriptless scripts with ECDSA. 2018. https://lists.linuxfoundation.org/pipermail/lightning-dev/attachments/

20180426/fe978423/attachment-0001.pdf

[27] Ducas L, Lepoint T, Lyubashevsky V, Schwabe P, Seiler G, Stehlé D. Crystals-Dilithium: Digital signatures from module lattices. Cryptology ePrint Archive, 2017.

[28] Esgin MF, Ersoy O, Erkin Z. Post-quantum adaptor signatures and payment channel networks. In: Proc. of the 25th European Symp. on Research in Computer Security. Guildford: Springer, 2020. 378–397. [doi: 10.1007/978-3-030-59013-0_19]

[29] Peng C, Luo M, He DB, Huang XY. Adaptor signature scheme based on the SM2 digital signature algorithm. Journal of Computer Research and Development, 2021, 58(10): 2278–2286 (in Chinese with English abstract). [doi: 10.7544/issn1000-1239.2021.20210645]

[30] Fischlin M. Communication-efficient non-interactive proofs of knowledge with online extractors. In: Proc. of the 25th Annual Int'l Conf. on Advances in Cryptology. Santa Barbara: Springer, 2005. 152–168. [doi: 10.1007/11535218_10]

[31] Fiat A, Shamir A. How to prove yourself: Practical solutions to identification and signature problems. In: Proc. of the 1986 Advances in Cryptology. Berlin: Springer, 1986. 186–194. [doi: 10.1007/3-540-47721-7_12]

[32] Chaum D, Pedersen TP. Wallet databases with observers. In: Proc. of the 12th Annual Int'l Cryptology Conf. on Advances in Cryptology. Santa Barbara: Springer, 1992. 89–105. [doi: 10.1007/3-540-48071-4_7]

[33] Zhang ZF, Yang K, Zhang J, Chen C. Security of the SM2 signature scheme against generalized key substitution attacks. In: Proc. of the 2nd Int'l Conf. on Security Standardization Research. Tokyo: Springer, 2015. 140–153. [doi: 10.1007/978-3-319-27152-1_7]

[34] Tu BB, Wang XF, Zhang LT. Two distributed applications of SM2 and SM9. Journal of Cryptologic Research, 2020, 7(6): 826–838 (in Chinese with English abstract). [doi: 10.13868/j.cnki.jcr.000409]

[35] Shang M, Ma Y, Lin JQ, Jing JW. A threshold scheme for SM2 elliptic curve cryptographic algorithm. Journal of Cryptologic Research, 2014, 1(2): 155–166 (in Chinese with English abstract). [doi: 10.13868/j.cnki.jcr.000015]

[36] Feng Q, He DB, Luo M, Li L. Efficient two-party SM2 signing protocol for mobile internet. Journal of Computer Research and Development, 2020, 57(10): 2136–2146 (in Chinese with English abstract). [doi: 10.7544/issn1000-1239.2020.20200401]

[37] Zhang YD, He DB, Zhang MW, Choo KKR. A provable-secure and practical two-party distributed signing protocol for SM2 signature algorithm. Frontiers of Computer Science, 2020, 14: 143803. [doi: 10.1007/s11704-018-8106-9]

[38] He DB, Zhang YD, Lin C, Feng Q, Wang J, Zhang JN. Method for multi-party associated generation of SM2 digital signature. China, 109474422A, 2019-03-15 (in Chinese).

[39] Zhang LT, Wang XF, Pan WL. SM2-based both side signature method and system. China, 109450640A, 2019-03-08 (in Chinese).

附中文参考文献:

[6] 单进勇, 高胜. 区块链理论研究进展. 密码学报, 2018, 5(5): 484–500. [doi: 10.13868/j.cnki.jcr.000258]

[20] SM2椭圆曲线公钥密码算法第2部分: 数字签名算法 2016 (in Chinese). https://std.samr.gov.cn/gb/search/gbDetailed?id=71F772D811F7D3A7E05397BE0A0AB82A

[21] 汪朝晖, 张振峰. SM2椭圆曲线公钥密码算法综述. 信息安全研究, 2016, 2(11): 972–982.

[29] 彭聪, 罗敏, 何德彪, 黄欣沂. 基于SM2数字签名算法的适配器签名方案. 计算机研究与发展, 2021, 58(10): 2278–2286. [doi: 10.7544/issn1000-1239.2021.20210645]

[34] 涂彬彬, 王现方, 张立廷. 两种分布式SM2/9算法应用. 密码学报, 2020, 7(6): 826–838. [doi: 10.13868/j.cnki.jcr.000409]

[35] 尚铭, 马原, 林璟锵, 荆继武. SM2椭圆曲线门限密码算法. 密码学报, 2014, 1(2): 155–166. [doi: 10.13868/j.cnki.jcr.000015]

[36] 冯琦, 何德彪, 罗敏, 李莉. 移动互联网环境下轻量级SM2两方协同签名. 计算机研究与发展, 2020, 57(10): 2136–2146. [doi: 10.7544/issn1000-1239.2020.20200401]

[38] 何德彪, 张语荻, 林超, 冯琦, 王婧, 张佳妮. 一种多方协同产生SM2数字签名的方法. 中国, 109474422A. 2019-03-15.

[39] 张立廷, 王现方, 潘文伦. 基于SM2的两方签名方法及系统. 中国, 109450640A, 2019-03-08.

涂彬彬(1993－), 男, 博士生, 主要研究领域为公钥密码学, 安全多方计算.

陈宇(1983－), 男, 博士, 教授, 博士生导师, 主要研究领域为理论密码学及其应用.