中国科学: 信息科学 2015 年 第 45 卷 第 11 期: 1370–1388
www.scichina.com info.scichina.com
《中国科学》杂志社
SCIENCE CHINA PRESS

评 述

可验证计算研究进展

薛锐[①*], 吴迎[①], 刘牧华[①], 张良峰[②], 章睿[①]

① 中国科学院信息工程研究所信息安全国家重点实验室, 北京 100093
② 上海科技大学信息科学与技术学院, 上海 200031
* 通信作者. E-mail: xuerui@iie.ac.cn

收稿日期: 2014–10–31; 接受日期: 2015–04–16; 网络出版日期: 2015–07–06
国家自然科学基金 (批准号: 61402471, 61472414, 61170280) 和中国科学院战略性先导科技专项 (批准号: XDA06010701) 资助

摘要 可验证计算是分布式计算和云计算环境下, 解决任务分包以及任务委托计算中产生的计算结果可靠性 (可信性) 问题的重要手段. 本文总结可验证计算重要研究进展, 主要涉及计算机理论领域和密码学领域的最新研究进展. 在计算机理论领域的可验证计算研究方面, 讨论了交互式证明、可随机检查证明定理 (PCP theorem) 和可靠性证明 (CS proof) 之间的联系、发展及其在可验证计算中的应用. 在密码学领域的可验证计算方面, 主要对应用密码学工具构建的可验证计算方案进行了分析总结. 此外, 简要介绍代理存储背景下的可验证计算. 最后对可验证计算未来的发展方向进行展望.

关键词 可验证计算 交互式证明系统 全同态加密 同态 MAC 同态签名

1 引言

随着分布式计算和云计算的快速发展, 出现了新型的服务计算模式: 分包计算以及外包计算.

在分布式计算中常常需要若干计算机互相配合以完成同一项目. 目前常见的分布式计算项目, 常常利用世界各地志愿计算机的闲置计算能力, 通过互联网通信完成相关任务. 例如, 分析计算蛋白质的内部结构和相关药物的 Folding@home 项目. 该项目结构庞大, 需要惊人的计算量, 单台电脑不可能完成计算. 由于能力超强的超级计算机往往造价昂贵, 而借助分布式计算, 可以相对廉价地完成计算任务. 这种方式相当于一个分包商把大型的计算任务分解, 再分包给不同的计算机进行计算.

随之而来的问题是, 单个用户产生的每个计算结果是否均是按照要求计算得到的? 有没有人为或非人为的因素影响结果的可靠性? 由于在分布式计算中, 计算软件对单个用户方是透明的, 用户方可以方便地修改发包方的软件. 不排除存在着各种动机使得用户方不诚实计算的情况存在. 比如, 某些用户仅仅是想提高在网络上的排名, 而速度很快地返回一个 (并非可靠计算得到的) 结果, 等等. 很多分布式计算项目解决这类问题的办法是, 把同样的任务单元分配给不同的计算方, 通过对比所返回结果的差异来确定正确的计算结果. 这造成了资源的极大浪费.

云计算是继分布式计算后的一种新型计算模式, 它以资源租用、应用托管、服务外包为核心, 迅速成为计算机技术发展的热点. 它旨在通过整合分布式资源, 构建可以应对多种服务要求的计算环境, 满足用户定制并通过网络访问相应服务资源的要求. 外包计算是云计算模式的优点之一, 它使得云用

引用格式: 薛锐, 吴迎, 刘牧华, 等. 可验证计算研究进展. 中国科学: 信息科学, 2015, 45: 1370–1388, doi: 10.1360/N112014-00336

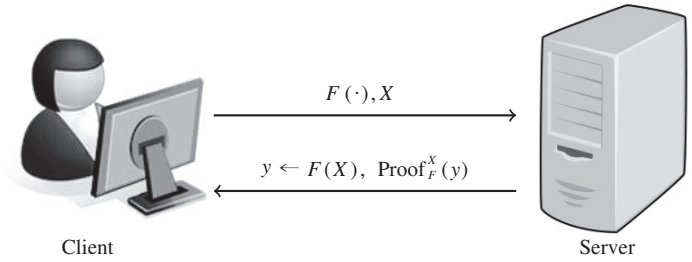

图 1 可验证计算模式

Figure 1 A mode in verifiable computation

户的计算能力不再受限于个体计算资源的约束, 通过外包任务给云端, 用户可以方便地利用云提供的庞大资源来完成高代价的计算.

云计算模式的动态性、随机性、复杂性和开放性等特点也给云计算服务的安全性带来了前所未有的挑战 [1]. 其中之一就是我们关心的计算可靠性问题. 例如, 对消耗大量资源的计算, 云可能会为了节约资源而 "偷懒" 返回一个不经计算的答案. 此外, 某些软件的 bug 和恶意的外部攻击, 也会影响最终计算结果的正确性. 所以, 计算可靠性问题是云计算研究的关键问题之一 [2].

上述分布式计算中的分包问题, 以及云外包计算中的问题是目前的研究热点. 抽象来说, 计算可靠性问题就是一方委托另一方 (或多方) 完成某个计算任务, 所得到的结果要能够被验证其可靠性. 可验证计算可以概括为: 客户端 (client) 由于自身能力或资源的限制, 无法完成复杂函数 $F(x)$ 在点 a 上值的计算, 将计算 $F(a)$ 的任务委托给一个计算能力强的服务器 (server). 由于服务器不是完全可信的, 这就需要服务器返回一个可验证其结果正确性的证明. 这种验算过程必须比实际计算 $F(a)$ 值本身简易的多, 否则就失去委托计算的意义. 可验证计算模式如图 1 所示. 根据实际使用背景和意义, 一个可验证计算方案需要满足以下 3 个基本条件.

(1) 正确性. 服务器按照方案诚实运算得到的结果, 返回给客户端一定能够通过验证.

(2) 安全性或可靠性. 安全或者可靠的可验证计算方案保证客户端不会接受服务器返回的错误结果.

(3) 高效性. 方案中规定的客户端对函数和输入预处理所花费的时间, 连同随后进行的可靠性验证所花费的时间, 要远小于直接计算函数所需要的时间, 否则也就失去了委托计算的意义.

除了以上 3 个基本要求外, 数据的隐私性也是可验证计算中考虑的主要因素之一. 可验证计算所涉及的隐私性主要有两种, 一种是客户端的数据相对于服务器的隐私保护, 另一种是最终计算结果相对于非授权用户的隐私保护. 这两种隐私保护主要采用密码学手段来实现, 如使用同态加密, 既可以保护数据的隐私, 又有利于计算的实施.

尽管数据的隐私保护非常重要, 在可验证计算方案的设计中如何实现高效的验证机制是一个挑战, 也是方案设计主要关键点. 所以本文以可验证计算方案中验证机制的实现手段为主线进行综述. 但是, 隐私保护是外包计算研究的主要问题之一, 读者可以参见综述文献 [3] 中详尽的介绍.

可验证计算已有近 30 年的发展历程. 由于解决该问题所采用的研究方法和工具的差异, 逐步形成 3 个不同的学术研究团体和领域: 应用安全领域、计算机理论领域和密码学领域.

在应用安全领域, 主要是由分布式计算系统中的安全问题, 引发了可验证计算的研究. 该领域采用的方法是从应用角度出发的, 以基于审计和各式各样的安全协处理器工具为主. 基于审计的办法 [4,5] 主要是客户端随机选取服务器的一部分工作重新计算, 以证实其计算的可靠性. 由于选取的随机性,

服务器无法预测哪些工作被重新分配, 这样就可以检验出服务器是否有不诚实的行为. 但若由客户端重新计算, 就要求客户端有足够资源. 若随机选取服务器来重新计算则无法抵御服务器的合谋攻击.

另外一种方法就是使用安全硬件的方法, 如利用协处理器的方法. 一个安全协处理器 [6,7] 是一个硬件模块, 它包含 3 个部分: (1) 一个 CPU; (2) 只读存储器引导程序 (bootstrap ROM); (3) 安全的非遗失性存储器 (secure non-volatile memory). 安全协处理器 [6] 可以提供隔离的执行环境, 入侵者可能破坏处理器并看到它的构造, 但是入侵者并不能知道或改变处理器的内部状态. 它保证了存储的隐私性和完整性, 并且为建立安全的分布式系统提供了基础. 所谓的可信计算就是利用了这个原理. 然而这种防篡改的能力使得它的成本非常昂贵, 因此很难推广使用.

在计算复杂性和密码学两个领域, 近年来涌现出很多新的可验证方案, 并且这些方案对硬件没有要求, 发展要比应用安全领域活跃. 本文仅仅关注基于计算复杂性和利用密码学的手段进行可验证计算方案构建方面的研究, 不涉及上述安全应用领域的方案.

本文第 2 和 3 节分别介绍了典型的基于计算复杂性理论和密码学手段的可验证计算方案. 我们根据可验证计算模式中客户端和服务器的数量进行分类讨论. 对于基于计算复杂性理论的可验证计算方案, 以方案中采用的工具为主线展开介绍: 交互式证明系统、PCP 定理、二次张成程序和 Referred Game 等. 基于密码学手段的可验证计算主要按照方案满足的性质: 私自可验证计算和公开可验证计算为主线作介绍.

2 基于计算复杂性理论的可验证计算

计算复杂性理论是整个计算机科学领域的核心组成部分之一. 近 20 年来, 计算复杂性理论最重要的成果之一, 是发展了交互式证明系统理论. 交互式证明系统与可验证计算有着非常相似的应用场景. 这就使得利用交互证明系统的重要结论来构造可验证计算方案成为一个重要研究方向.

根据可验证计算模式中客户端和服务器的数量, 可验证计算可以分为: 单客户端单服务器、单客户端多服务器、多客户端单服务器和多客户端多服务器 4 种模式. 基于计算复杂性理论的研究以前两种模式为主, 所以本节对前两种模式中的可验证计算方案进行分析总结. 由于同种模式下, 使用同类工具形成的方案具有某些共性, 方案之间也更具有可比性. 因此在每种模式下, 我们根据构造方案利用的主要工具又进行了分类讨论. 总体来讲, 方案中使用的工具有: 交互式证明系统、PCP 定理、CS 证明、多证明者的交互式证明系统、二次张成程序等. 这些工具的详细内容会在本节中给出.

2.1 单客户端单服务器的可验证计算方案

单客户端单服务器的模式是指单个客户端将计算任务委托给单个服务器, 客户端具有所要外包的函数和函数的输入.

2.1.1 基于交互式证明系统的可验证计算方案

Goldwasser 等 [8] 以及 Babai [9] 分别提出了交互式证明系统. 交互式证明系统突破传统证明系统, 从证明者到验证者单向传递的方式, 采取多次来回发送消息, 对于每个结论进行证明.

在理论计算机科学中, 一个问题实例的全体组成一个集合, 也称为一个语言. 要证明某个具体问题的解, 就可以表达成为证明某个元素 x 在某个语言 L 中. 所以一个交互式证明系统可以定义如下.

定义1 (交互证明系统) 一个语言 L 的交互式证明系统是一个由证明者 P 和验证者 V 组成的交互过程, 二者具有共同的输入, 并且在交互过程满足如下 3 个条件.

(1) 验证者的策略是一个概率多项式时间的过程.

(2) 证明者的计算能力没有限制.

(3) 正确性要求如下:

• 完备性. 存在一个证明策略, 当交互的输入为任意的 $x \in L$ 时, 证明者 P 至少以 2/3 的概率使得验证者接受.

• 可靠性. 当交互的输入为任意的 $x \notin L$ 时, 对于证明者的任意的策略 P^*, 至多只能以 1/3 的概率使得验证者接受.

结合可验证计算的场景, 验证者 V 把计算外包给证明者 P, P 将计算结果返回给 V 并向 V 证明结果的正确性. 交互式证明系统的可靠性要求, 保证 V 不会接受一个错误的计算结果 (或者说, 接受一个错误结果的概率是可忽略的). 但是交互证明系统很少能够应用到实际的委托计算场景中, 原因在于 P 的计算能力是没有限制的. 在实际可验证计算中, 我们只考虑概率多项式时间内可计算的函数. 同时我们要求验证者 V 运行时间要小于实际计算函数所需时间, 这样才能够达到外包计算的目的.

2008 年, Goldwasser 等 [10] 提出了 "Muggle Proof " 模型, 将证明者的能力限制为多项式时间, 验证者为准多项式时间.

可以由大小为 poly(n), 深度为 polylog(n) 的 $O(\log(n))$-space 一致电路族计算的语言, 称为 $\mathcal{L}$-uniform **NC** 类语言. 其中 n 为电路的输入长度.

定理1 对任给的属于 $\mathcal{L}$-uniform **NC** 类的语言 L, L 存在满足以下条件的交互证明:

• 证明者的时间复杂度为 poly(n);

• 验证者的运行时间为 $n \cdot \text{polylog}(n)$;

• 运行所需空间复杂度为 $O(\log(n))$;

• 协议的通信复杂度为 polylog(n).

文献 [10] 的构造具有非常重要的意义. 首先, 它考虑的是高效的证明者, 符合实际应用需要. 其次, 文中采用的技巧对之后的很多研究具有重要的参考价值. 该方案可以简要描述如下.

GKR 方案 证明者的输入为一个大小为 S, 深度为 d 的电路 $C : \{0,1\}^n \to \{0,1\}$ 和一个字符串 $x \in \{0,1\}^n$, 证明者要向验证者证明 $C(x) = 0$. 不失一般性, 假设 C 是分层电路.

(1) 对 C 的层进行升序编号, 输出层是 0 层, 输入层是 d. 每一层都有一个向量 v_i 与之对应, $0 \leqslant i \leqslant d$, 其中 v_i 包含输入为 x 时 i 层各个门的值, V_i 是 v_i 的低阶扩张 (low degree extension).

(2) 证明者只需向验证者证明 $V_0(0,\ldots,0) = 0$. (证明者将第 i 层的低阶扩张的值归约到 $i+1$ 层).

(3) 验证者只需要计算 V_d 在一点的值, 与证明者给出的值做对比.

在 GKR 方案中, 虽然证明者是高效的, 验证者的时间复杂度是准线性的, 但是在协议执行过程中证明者需要记录很多额外信息, 并且验证者要持有整个输入, 这使得程序实现存在很大障碍.

2012 年, Cormode 等 [11] 通过构造数据流模型, 对 Goldwasser 等 [10] 的构造进行改进, 然后对改进协议作了具体实现, 降低了证明者的运行时间. 对于由大小为 S 的电路所计算的语言, 证明者的运行时间为 $O(S \log S)$.

2.1.2 基于 PCP 定理的可验证计算

Fortnow 等在文献 [12] 中提出一个与多证明者的交互证明系统能力等价的模型. 该模型在后来 Arora 等 [13] 的文章中被称为可随机检查证明 (probabilistically checkable proofs, PCP), 定义如下.

定义2 (PCP) 若 M 是一个多项式时间概率图灵机 (Turing machine) 并且可以访问应答器 (oracle) O. M 接受语言 L 当且仅当:

- 存在一个应答器 O, 对于任意的 $x \in L$, M^O 以大于 $1-2^{-n}$ 的概率接受 x;
- 对任意的 $x \notin L$, 对所有的应答器 O', $M^{O'}$ 接受 x 的概率都小于 2^{-n}.

可随机检查证明中, 确定了应答器后, 对验证者问题的回答都是确定性的. 而交互式证明系统中, 证明者可以根据以前的回答来调整当前的回答. 在可随机检查证明中, 验证者使用的随机串的长度和验证者从应答器上获得信息的长度是两个非常重要的参数. 围绕如何降低这两个参数产生了一系列后续的研究.

1990 年, Shamir [14] 证明了 IP = PSPACE, 即所有 PSPACE 中的语言都可以通过一个交互证明系统高效地验证. 文中使用了两种重要技术: 低阶多项式技术和算术化方法. 它们对于 PCP 系统的性质证明起到了关键的作用. 受 Shamir 的启发, 1994 年, Arora 等 [15] 证明了 NP = PCP($O(\log(n)), O(1)$), 称之为 PCP 定理. 即证明者向验证者证明字符串 x 属于某个 NP 语言 L, 证明者可以准备一个证明, 验证者只需要访问该证明中的常数个位置, 就可以判定 x 是否属于 L. 虽然验证者访问的位置较少, 但是验证者需要保存整个证明, 证明的长度有可能很大, 以至于验证者无法存储.

为此, Kilian [16] 利用 Merkle 树构造了一个短的承诺发给验证者, 验证者可以交互地打开某些比特. 应用到可验证计算中, 即证明者将计算结果和一个短的承诺发给验证者, 验证者通过交互打开承诺的某些比特来验证计算结果的正确性. 上述讨论的协议都是交互的, 利用 Micali 的 CS 证明 [17] 和 Hash 函数的可抽取假设 [18][1)], 可以将协议变成非交互的. 由于 Killian [16] 的构造中应用了抗碰撞的 Hash 函数, 完备性和可靠性是在随机应答器模型下证明的. 证明者的时间复杂度限制在 $O(S^{1.5})$.

2.1.3 基于二次张成程序 (QSP) 的可验证计算

在讨论 NP 语言的交互证明时, 通常归约到 NP 完全问题, 如线路可满足性. PCP 证明线路可满足性时, 用到的一个非常重要的技术是电路算术化, 电路的每个门都用一个二次多项式代替. 通过验证多项式的值来检测电路的可满足性问题. 由上述讨论知道需要使用 Hash 函数将 PCP 中的长证明变短. 一个自然的问题是, 是否存在其他的算术化的方法避免使用这种计算?

Groth [19] 在公共参考串 (CRS) 模型 [20] 下构造了适用于双线性映射群的证明. 利用循环群上的双线性映射, 将多项式放在指数上来验证. 其可靠性基于 q-computational power Diffie-Hellman 假设, 但是证明者的复杂度和 CRS 的大小都是二次的. Lipmaa [21] 进一步给出了将 CRS 的大小降为准线性的方法. 但是证明者的计算复杂性仍然是二次的. 理想的协议是, 证明者的时间复杂度是关于线路大小的线性函数.

交互式证明可以通过 Hash 函数的可抽取假设变成非交互的, 在这个构造的发展过程中, 定义了简明的非交互知识论证系统 (succinct non-interactive argument of knowledge, SNARK). 近年来很多研究工作是围绕 SNARK 进行的, 这里对它作简单介绍. SNARK 是在简洁的非交互论证系统 (succinct non-interactive argument, SNARG) 的基础上定义的. 简洁的非交互论证系统由 3 个算法 (Gen, P, V)

1) Goldwasser S, Lin H J, Rubinstein A. Delegation of computation without rejection problem from designated verifier CS-proofs. IACR Cryptology ePrint Archive, 2011: 456.

构成:

• $\mathrm{Gen}(1^\lambda) \to (crs, priv)$. 输入安全参数 λ, 输出公共参考串 crs 和私自验证信息 $priv$.

• $P(crs, u, w) \to \pi$. 使用证据 w, 证明者 P 对陈述 u 产生一个证明 π.

• $V(crs, priv, u, \pi) \to 0$ 或 1. V 验证 π, 判定是否接受陈述 u.

要注意的是, 这里证明者生成证明 π 和验证者验证证明 π 均要使用同一个随机串 crs. 在简洁的非交互论证系统基础上可以定义简明的非交互知识论证系统.

定义3 (SNARK) 满足下述条件的 $\Pi = (\mathrm{Gen}, P, V)$ 是满足关系 R 的 NP 语言 L 的 SNARK.

• 完美完备性. 对于任意的算法 A,

$$\Pr\left[\begin{array}{c|c} V(priv, u, \pi) = 1 & (crs, priv) \leftarrow \mathrm{Gen}(1^\lambda) \\ \text{if } (u, w) \in R & (u, w) \leftarrow A(crs),\ \pi \leftarrow P(crs, u, w) \end{array}\right] = 1,$$

其中, $P(crs, u, w)$ 的运行时间是 $\mathrm{poly}(\lambda, n)$, n 是 (u, w) 的长度.

• 可靠性. 对于任意的概率多项式时间算法 A, 存在可忽略函数 $\mathrm{negl}(\lambda)$, 使得

$$\Pr\left[\begin{array}{c|c} V(priv, u, \pi) = 1 & (crs, priv) \leftarrow \mathrm{Gen}(1^\lambda) \\ u \notin L & (u, \pi) \leftarrow A(1^\lambda, crs) \end{array}\right] \leqslant \mathrm{negl}(\lambda).$$

• 简洁性. 证明的长度

$$|\ \pi\ | = \mathrm{poly}(\lambda + |u|).$$

• 可抽取性. 对于任意多项式大小的 P^*, 存在多项式大小的提取器 ε_{P^*}, 对任意的 $z \in \{0,1\}^\lambda$, 存在可忽略函数 $\mathrm{negl}(\lambda)$, 使得

$$\Pr\left[\begin{array}{c|c} V(priv, u, \pi) = 1 & (crs, priv) \leftarrow \mathrm{Gen}(1^\lambda) \\ (u, w) \notin R & (u, \pi) \leftarrow P^*(crs, z),\ w \leftarrow \varepsilon_{P^*}(crs, z) \end{array}\right] \leqslant \mathrm{negl}(\lambda).$$

Gennaro 等 [22] 对 NP 复杂类, 定义了二次张成程序 (quadratic span programs, QSPs), 并用其构造 SNARK, 避免使用 PCP.

QSP 具体描述如下: QSP 由两个多项式集合 $V = \{v_1, v_2, \ldots, v_{n+m}\}$ 和 $W = \{w_1, w_2, \ldots, w_{n+m}\}$, 以及一个目标多项式 t 组成.

QSP (V, W, t) 计算输入长为 n 比特的布尔函数 F 是指, 当且仅当对任意满足 $F(x) = 1$ 的 $x = x_1, \ldots, x_n \in \{0,1\}^n$, 一定有

$$t(x) \left| \left(\sum_{i=1}^{n} a_i v_i(x)\right) \cdot \left(\sum_{i=1}^{n} b_i w_i(x)\right), \right.$$

而且满足当 $x_i = 0$ 时, $a_i = b_i = 0$.

利用 QSP 构造可验证方案.

(1) 在预处理阶段, 验证者随机选取 s, 计算并公布 g^{s^i}, $g^{v_i(s)}$, $g^{w_i(s)}$ 和 $g^{t(s)}$, 对于 s 保密.

(2) 输入 x, 证明者计算系数 a_i, b_i 和多项式 h 使其满足

$$t \cdot h = \left(\sum_{i=1}^{n} a_i v_i\right) \cdot \left(\sum_{i=1}^{n} b_i w_i\right).$$

(3) 利用验证者公布的值, 证明者可以计算 $g^{t\cdot h(s)}$, 验证者用双线性映射检验方程.

方案的安全性基于 q-power Diffie-Hellman 假设, 验证者在输入准备阶段的时间复杂度是线性的, 验证的时间是常数, 证明者是准线性的.

将上述协议应用到可验证计算中, 客户端委托服务器计算函数 $F:\{0,1\}^n\to\{0,1\}^m$, 首先构造一个函数 $f(x,w)$ 满足 $f(x,w)=1$ 当且仅当 $F(x_{[1,n]})=x_{[n+1,n+m]}$. 服务器计算完函数值后, 构造关于 f 的上述协议, 向客户端证明结果的正确性.

研究可验证计算的目的就是能够找到可以实际应用的协议. Parno 等 [23] 和 Sasson 等 [24] 应用 QSP 分别构造了实例化系统 Pinocchio 和 SNARKs-for-C.

2.2 单客户端多服务器的可验证计算方案

在单客户端单服务器的可验证计算发展过程中, 由于技术手段的限制, 很多方案都存在计算效率低, 外包函数受限和交互轮数较高等缺点. 研究者试图寻找相对高效的、具有普遍性的解决方案.

• Canetti 等 [25] 利用 Refereed Games [26] 构造了外包给两个或多个服务器的可验证计算方案. 只要有一个服务器是诚实的, 就可以保证客户端得到正确的结果. 文献 [25] 构造的协议是针对任意高效计算的函数, 协议的交互轮数为安全参数的对数.

• 文献 [27] 结合文献 [10] 中的算术化技术和 Refereed Games 构造了对数空间一致 NC 电路的一轮协议, 协议可以达到统计可靠性.

• 文献 [28] 在安全性和效率方面增强了之前的结果. 上述介绍的协议都是基于 Refereed Games 构造的, 要求至少有一个服务器是诚实的.

• Blumberg 等 [2)] 基于多证明者的交互证明系统构造了可验证计算方案. 该方案不需要假设证明者之一是诚实的. 文中构造了实例化系统 Clover.

3 基于密码学手段的可验证计算

在密码学领域, 外包计算的研究已有 20 多年的历史. 早期具有鲜明密码学特征的可验证计算协议当属文献 [29] 中的构造. 在 CRYPTO'92 上, Chaum 等 [29] 提出 "electronic wallet" 模型, 并利用盲签名构造了具体协议, 这是利用密码学手段研究可验证计算的开端.

本节仍然是根据客户端和服务器的数量来分类讨论可验证计算方案的构造. 在不同的可验证计算模式下, 以方案的特性 (方案中委托与验证的公开与否) 为依据进行总结分析. 应用到的主要密码学工具有: 全同态加密、混淆电路、基于属性的加密、同态 MAC/签名等.

由于以同态 MAC/签名方案为主要手段的可验证计算方案共性鲜明, 方案之间的对比可以看出研究成果的递进, 我们对其进行单独讨论.

3.1 单客户端单服务器的可验证计算

3.1.1 可私自验证计算

可私自验证计算是指客户端委托服务器计算函数 F 关于 x 点的值, 客户端会对函数 F 和 x 进行预处理, 处理信息的公开部分传给服务器, 保留私有部分, 并且客户端不会对第三方公开预处理的私

2) Blumberg A J, Thaler J, Walfish M, et al. Verifiable computation using multiple provers. IACR Cryptology ePrint Archive, 2014: 846.

有信息. 对于返回的计算结果, 客户端只能自己进行验证.

计算机理论方面多侧重于交互式协议的研究, 而密码学研究者试图用密码学手段构造非交互的可验证计算方案. 当然, 基于计算机理论的方案也有非交互的, 但是它们的安全性有的需要在随机应答器模型下证明, 有的基于不可证伪的假设 (non-falsifiable assumption). 此外, 基于计算机理论的可验证计算方案不考虑输入和输出结果的隐私性, 而密码学研究者会使用密码手段实现输入和输出结果的隐私性.

Gennaro 等 [30] 利用 Yao's Garbled Circuit [31] 构造了非交互的可验证计算方案, 它克服了基于计算机理论方案的弊端. Yao's Garbled Circuit 可以构造一次安全的可验证计算方案, 安全性是基于 Yao's Garbled Circuit 的可靠性 (authenticity, Bellare [32] 形式化定义 garbled scheme 时定义了这一性质). 利用全同态加密 (FHE) [33,34] 将一次安全的可验证计算方案变为多次安全的. Chung 等在文献 [35] 中也应用了这一技巧. Gennaro 等 [30] 的方案可以达到输入和输出结果的隐私性.

文献 [30] 的另一个突出贡献是, 形式化定义了可验证计算方案. 之后很多研究都是基于他们的定义, 现将其定义描述如下.

定义4 (可验证计算方案) 一个可验证计算方案 $\mathcal{VC}$ 由以下定义的 4 个算法组 (KeyGen, ProbGen, Compute, Verify) 构成.

• KeyGen. 密钥生成算法是一个随机算法, 输入 (F,λ), 输出 $(\mathrm{PK},\mathrm{SK})$: 它以安全参数 λ 和函数 F 作为输入, 生成公钥 PK 和私钥 SK. PK 是 F 的编码. 这一阶段称为预处理阶段.

• ProbGen. 问题生成算法输入 SK,x, 输出 (σ_x,τ_x): 利用 SK 编译 x 得到公开值 σ_x 和私有值 τ_x.

• Compute. 输入 PK,σ_x, 输出 σ_y: 服务器利用 PK 和 σ_x , 计算 $y=F(x)$ 的编码值 σ_y.

• Verify. 输入 $\mathrm{SK},(\tau_x,\sigma_y)$, 输出 y 或 $\perp$: 客户端利用私有信息解码服务器返回的编码值, 输出 $y=F(x)$ 或者 $\perp$, 其中 $\perp$ 表示客户端拒绝服务器返回的结果 σ_y. 一个可验证计算方案最基本的要求就是正确性, 它要求对函数 F 以及 $\forall x\in\mathrm{Domain}(F)$, 客户端和服务器都正确执行上述 4 个算法, 客户端最后得到正确的函数值 $F(x)$.

根据可验证计算的目的, 方案需要能够确保客户端不受服务器欺骗, 这一性质定义为方案的安全性. 方案的安全性由客户端接受敌手返回的错误函数值的概率来衡量. 方案安全性的强弱由敌手的能力决定. 一个方案是安全的是指敌手无法令客户端接受一个错误的函数值, 也就是说接受一个错误函数值的概率是可忽略的.

可验证计算中客户端的运行时间要比计算函数所需的时间短, 否则可以自己完成计算, 这就失去了研究可验证计算的意义. 上述定义算法的效率是建立在分摊的意义上. 客户端在预处理阶段可以花费适当长的时间, 但对于同一个函数只需要进行一次预处理, 客户端通过委托服务器计算同一函数在不同输入情况下的值, 来分摊预处理阶段的时间. Gennaro 等 [30] 的方案花费 $\mathrm{poly}(T,\lambda)$ 时间生成长度为 $\mathrm{poly}(T,\lambda)$ 的公钥 (其中 T 是 F 的时间复杂度).

为了减短公钥长度从而降低通信复杂度, Chung 等 [35] 基于 FHE 构造了非交互的可验证计算方案. 该方案没有使用 Yao's Garbled Circuit, 公钥的长度为 0. 然而, 在预处理阶段, 仍然需要花费 $\mathrm{poly}(T,\lambda)$ 时间去计算函数 F 关于一些随机输入的值, 客户端将 KeyGen 算法外包给服务器, 利用通用论证系统 (universal argument) [36] 证明计算结果的正确性. 预处理阶段客户端的时间复杂度降为 $\mathrm{poly}(\log T)$. 文中方案虽然使得客户端的时间复杂度大大降低, 却是以增加服务器的工作量为代价的, 增加了外包计算的成本.

Applebaum 等 [37] 提出了一种新的构造可验证计算的方法. 他们使用消息认证码 (MAC) 有效地

把一个多方计算协议的安全性质, 转化为一个可验证计算方案的可靠性. 与文献 [30, 35] 中的方案相比, 该方案更加简单高效. 文献 [37] 中的方案是基于随机编码 (randomized encoding) 和 MAC 构造的.

从安全性角度来讲, 文献 [30, 35] 中的方案安全性都较弱, 对敌手返回的错误的计算值, 客户端不能将验证结果泄露给敌手, 否则方案会变得不安全. 敌手能否询问客户端的验证结果, 这一问题称为 "rejection problem". Gennaro 等 [30] 将能否构造抵抗 "rejection problem" 的可验证计算方案作为一个公开问题提出. Barbosa 等 [38] 构造了第一个抵抗 "rejection problem" 的可验证计算方案. 文献 [38] 提出了一个新的密码学构件, 称为代理同态加密 (delegatable homomorphic encryption, DHE). 对于能力受限的发送者和接收者, 发送者拥有数据 x, 而接收者希望得到函数 f 在 x 处的值 $f(x)$. 接收者委托服务器计算, 而且 x 是敏感数据需要对服务器和接收者保密. 文中利用函数加密 (FE), FHE 和 MAC 构造了一个 DHE 方案, 并利用 DHE 构造私自可验证计算方案. 由于方案需要安全通道和可信中心, 因此在现实中是很难实现的.

上面介绍的几个方案都是针对一般函数的, 其共同点在于所有构造都或多或少地基于全同态加密 (FHE). 众所周知, 现有的全同态加密效率低下, 这就使得目前建立在其上的可验证计算方案缺乏实用意义.

Benabbas 等 [39] 开辟了一条为特殊函数构造高效可验证计算方案的研究途径. 他们所研究的特殊函数 (主要包括各式各样的多项式函数) 在构造可验证的关键词搜索 (verifiable keyword search) 方案和可恢复性证明 (proof of retrievability) 方案等问题中都有重要的直接应用.

以一个高次多项式 $F(X) = \sum_{i=0}^{n} a_i X^i \in \mathbb{Z}_p[X]$ 为例. 当它的次数足够高时, 存储这样一个函数需要很大的空间, 计算该函数的值也需要大量的处理时间. 显然弱小的客户端既无法负担该多项式的存储任务, 也无法负担其计算任务. 对客户端来说, 一个自然的选择就是, 把存储和计算该多项式的任务委托给一个强大的云服务器. 对于这样一个特殊函数, Benabbas 等 [39] 给出了第一个高效的 (未使用 FHE) 可验证计算方案. 以下是该构造的信息论基础.

(1) 对于任意一个大素数 $p > 0$, 客户端随机地选取 $c \in \mathbb{Z}_p$, $\boldsymbol{r} = (r_0, \ldots, r_n) \in \mathbb{Z}_p^{n+1}$ 并为每个系数绑定一个标签 $t_i = ca_i + r_i$. 在服务器上存储 $pk = \{(a_i, t_i)\}_{i=0}^{n}$ 并保留 $sk = (c, \boldsymbol{r})$ 以作验证之用.

(2) 对客户端的任一输入 $x \in \mathbb{Z}_p$, 服务器返回计算结果 $y = F(x) = \sum_i a_i x^i$ 以及一个证明 $t = \sum_i t_i x^i$.

(3) 客户端验证 $t = cy + R(x)$ 是否成立, 这里 $R(x) = \sum r_i x^i$. 若所验证的等式成立则接受 y, 否则拒绝.

不诚实的服务器会返回一个伪造的结果 $\bar{y} \neq y$, 连同相应的证明 $\bar{t}$. 该伪造的结果被错误地接受, 当且仅当 $\bar{t} = c\bar{y} + R(x)$. 一旦该事件发生, 则服务器必然应实际上解得 $c = (\bar{y} - y)^{-1}(\bar{t} - t)$.

该信息论构造的安全性正是基于 "c 对于服务器是完全透明的" 这一事实. 虽然该信息论构造给出了一个安全的验证机制, 然而客户端的验证算法涉及到函数 $R(x)$ 的计算问题. 由于该计算的复杂度与计算 $F(x)$ 毫无二致, 怎样达到严格高效的验证要求就成为一个主要课题.

Benabbas 等 [39] 的一个重要创新就是, 选定一个高阶 (p 阶) 循环群 $\mathbb{G} = \langle g \rangle$, 并把信息论构造中的所有计算移到群生成元 g 的指数上. 这就使得随机向量 $\boldsymbol{r}$ 可被一个完全不随机的向量的指数函数所替代而不影响安全性. 这就导致了封闭式高效拟随机函数 (closed-form efficient PRF) 概念的诞生, 并实质上给出了严格高效的验证算法. 作为代价, 方案的安全性由信息论意义下的安全性转变为基于所用高效拟随机函数的安全性, 这又进一步转变为基于各式各样的数论假设的安全性, 如 SDDH 和 DDH 假设等等.

这类高阶多项式的可验证计算方案中, 输入值都是明文, 这对于隐私数据是不可接受的. Fiore

等[40] 讨论了加密数据的高效可验证计算方案. 文中引入一个新的构件, 称为同态 Hash 函数, 定义如下所示.

定义5 (同态 Hash 函数) 同态 Hash 函数族 $H:\mathcal{X}\to\mathcal{Y}$ 包含一个三元组 (KGen, H, Eval):

- KGen. 输入安全参数 1^λ, 生成函数描述 H_K.
- H. 关于输入 $\mu_1,\ldots,\mu_t\in\mathcal{X}$, 计算函数值 $H(\mu_1),\ldots,H(\mu_t)$.
- Eval. 利用输入 $H(\mu_1),\ldots,H(\mu_t)$ 和函数 f 的描述, 计算函数在 $f(\mu_1,\ldots,\mu_t)$ 处的哈希值, 即

$$\mathrm{Eval}(f,(H(\mu_1),\ldots,H(\mu_t)))=H(f(\mu_1,\ldots,\mu_t)).$$

文献 [40] 利用同态 Hash 函数, 文献 [41] 的加密方案以及文献 [42] 的同态 MAC 方案, 对多种类型的多项式函数 (线性组合、高阶单变量多项式、多变量二次多项式) 构造了高效的可验证计算方案. 基本思想是: 客户端将加密数据 $\mu=(\mu_0,\ldots,\mu_t)$ 传给服务器, 由服务器存储, 其中 μ_i 是原始数据 x_i 的密文. 对每个密文 μ_i 生成一个 MAC σ_i, 使用 MAC 的同态性质去认证密文的同态计算. 文献 [40] 委托计算的函数是 $f:\mathcal{F}_q^t\to\mathcal{F}_q$, 而 BGV [41] 计算电路是函数 $\hat{f}:\mathcal{F}_q^{2^n t}\to\mathcal{F}_q^{3n}$, 所以 BGV 同态加密得到的密文无法直接作为函数 f 的输入. 通过使用同态 Hash 函数可以压缩 BGV 密文为单个 $\mathcal{F}_q$ 中的元素. 该方案可以抵抗 "reject problem". 利用上述方案, 客户端可以外包存储大量的敏感隐私性数据给服务器, 要求服务器计算这些数据统计性质, 可以保证计算的正确性和数据的隐私性. Zhang 等[43] 通过多线性映射构造了单变量多项式函数和矩阵乘积的可验证计算方案, 方案可以保证输入和函数隐私.

3.1.2 可公开验证计算方案

可私自验证计算协议执行过程中产生的信息不对外公开. 而基于密码学手段的可私自验证方案的效率都是在分摊意义下考虑的, 只有客户端委托服务器计算同一函数关于多个不同输入的值时, 算法才有意义. 一个自然的想法是, 构造方案使得客户端不用对函数进行预处理, 直接利用其他客户端的预处理结果进行委托计算.

Parno 等[44] 提出了可公开验证计算的概念. 可公开验证计算有两个性质: 公开代理和公开验证. 公开代理是指一个客户端对函数 F 作预处理之后将信息公布, 其他客户端可以直接利用这些信息, 只需要对函数的输入作处理便可将计算任务委托给服务器. 公开验证是指任何人都可以验证服务器返回的结果. 公开可验证计算的形式化定义如下.

定义6 (可公开验证计算方案) 一个可公开验证计算方案 $\mathcal{VC}$ 由以下定义的 4 个算法 $(\mathrm{KeyGen},\mathrm{ProbGen},\mathrm{Compute},\mathrm{Verify})$ 构成,

- KeyGen. 密钥生成算法是一个随机算法. 它以安全参数 λ 和函数 F 作为输入, 输出一个短的公钥 PK_F 和公共计算的密钥 EK_F. PK_F 用于处理函数输入, EK_F 用于计算函数 F, 它的长度依赖于函数 F. 这一阶段称为预处理阶段.
- ProbGen. 问题生成算法. 利用 PK_F 编译 x 得到两个公开值 σ_x 和 VK_x.
- Compute. 服务器利用 EK_F 和 σ_x, 计算 $y=F(x)$ 的编码值 σ_y.
- Verify. 客户端利用公开验证秘钥 VK_x 和公开验证算法输出函数值 $y=F(x)$ 或 $\perp$, 其中 $\perp$ 表示客户端认为服务器返回的结果 σ_y 不是可靠的.

文献 [44] 利用密钥策略的属性加密方案 (key-policy attribute based encryption, ABE) 构造了可公开验证计算方案. 密钥策略 ABE 中, 明文的加密与属性相关. 当属性满足密文接收者的访问结构时,

接收者就可以正确解密出明文. Parno 等 [44] 将这种性质与可验证计算中的可验证性相结合. 客户端委托服务器计算函数 F 在 x 点的值, 随机选取一个明文 m, 以 x 作为属性加密, 将密文和访问结构 F 对应的私钥传给服务器, 服务器进行解密. 从而构造了可公开验证计算方案. 若服务器返回消息 m, 则表示 $F(x)=1$, 否则 $F(x)=0$. 这是因为如果 $F(x)=0$, 服务器只能以可忽略概率找到消息 m. 但是如果服务器返回给客户端一个不等于 m 的消息, 则有两种可能, 一种是 $F(x)=0$, 服务器确实不能解密出消息 m, 另一种是 $F(x)=1$, 服务器拒绝解密或者恶意欺骗客户端. 因此要防止第二种情况的出现, 观察上述协议的特点, 当 $F(x)=0$ 时, 服务器不能欺骗客户端.

文献 [44] 构造了函数 F 的补函数 $\bar{F}$ (定义为 $\bar{F}(x)=1-F(x)$), 对 F 和 $\bar{F}$, 同时运行上述协议则可以得到正确的 $F(x)$. 方案满足公开代理和公开验证两个性质, 但是只能计算单比特输出的函数, 对于多比特输出的函数需要多次运行协议, 这增加了委托成本. 此外适用于该方案的函数范围依赖于属性加密方案.

目前可公开验证计算的方案较少, 很多方案都只能满足公开代理和公开验证两个性质中的一个. Fiore 等 [45] 基于之前 Benabbas 等 [39] 的方案, 构造了针对高阶多项式函数和矩阵乘积的可以公开验证的计算方案. 方案的公开验证性是通过群上的双线性映射实现的. Catalano 等 [46] 引入了代数单向函数的概念, 并利用普通的数论假设 (如 RSA, 大数分解假设) 构造了相应的代数单向函数. 应用代数单向函数可以构造针对高阶多项式和矩阵乘积的可以公开验证的方案. 方案的安全性基于 RSA 假设, 而计算模幂乘要比计算双线性对速度快. 所以, Catalano 等 [46] 的方案要比 Fiore 等 [45] 的方案效率高.

Papamanthou 等 [47] 引入了一个云环境下动态计算的新模型 —— 正确计算的签名 (signatures of correct computation, SCC), 并构造了具体方案. SCC 比可公开验证计算要求更高, 它的算法是由一个可信方提供的, 客户端利用可信方提供的算法委托服务器对自己的数据作处理, 在验证服务器返回的结果时不需要任何关于数据的信息, 所以任何一方 (即使不信任客户端) 都可以验证计算结果. 由 SCC 方案可以构造出可公开验证计算方案. 文献 [47] 中的方案在随机应答器模型下是适应性安全的.

已知的可验证计算方案, 服务器必须存储函数的编码, 与只存储函数本身相比, 这样增加了服务器的存储负担. Zhang 等 [48] 针对多项式函数和矩阵乘积的代理计算构造了多个方案, 他们引入了一个参数来测量云端存储空间和客户端验证时间之间的关系. 方案允许客户端适当地增加验证时间来减少云存储的花费.

3.2 多客户端或多服务器的可验证计算

在单客户端的情形中, 客户端自己知道外包的函数 F 和输入值 x. 但是存在这种情形: 多个客户端, 每个客户端只拥有输入的一部分. 比如多个资源有限的节点构成的网络, 每个节点搜集的数据作为某个运算输入的一部分. 解决这类问题最直观的想法是利用之前单客户端的协议, 将所有输入都传给一个客户端然后由该客户端执行协议. 这种办法无法保护每个客户端输入的隐私性, 而且这样构成的协议是交互的.

Choi 等 [49] 提出了一个多客户端、非交互的可验证计算. n 个计算能力较弱的客户端, 每个人拥有函数 F 的部分输入 x_i, 委托服务器计算 F 在输入 $(x_1,\ldots,x_n)$ 的值 $F(x_1,\ldots,x_n)$. 除了能验证服务器返回结果的正确性之外, 客户端还希望自己的输入对于其他客户端是保密的. 文献 [49] 的构造与 Gennaro 等的方案 [30] 相似, 将双方传输协议改为适用于三方的 Proxy OT. 该构造满足较弱的安全性和隐私性: 首先, 恶意的服务器与客户端不能合谋; 其次, 客户端之间不能合谋; 最后, 客户端是半诚实的.

Goldwasser 等 [50] 构造了多输入的函数加密 (multi-input function encryption) 方案, 作为应用, 讨论了如何构造多客户端的可验证计算的方案. 由于文中多输入的函数加密方案是利用证据不可区分的零知识证明和不可区分的混淆 (indistinguishability obfuscation) 构造的, 所以方案依赖于不可证伪的假设 (non-falsifiable assumption) 或者亚指数困难性假设 (sub-exponential hardness assumption). Boneh 等 [51] 改进了文献 [50] 的结果, 利用多线性映射代替不可区分的混淆, 提高了方案的效率.

近来, Gordon 等 3) 讨论了多客户端的可验证计算中更强的安全性和隐私性模型, 利用基于属性的加密, 全同态加密和 Yao's Garbled Circuit 构造了多客户端的可验证计算方案, 并证明了在客户端和服务器之间没有合谋, 客户端之间可以合谋的情况下, 该方案是安全的. 文中的构造是对 Parno 等 [44] 和 Goldwasser 等 [52] 结果的推广. 此外 Gordon 等 3) 证明了当客户端和服务器可以任意合谋时, 构造针对一般函数的多客户端安全可验证计算方案是不可能的. 目前, 可验证计算方案的效率都依赖于客户端的个数. 所以未来一个有趣的研究方向就是构造这样一类方案, 其客户端的线上计算和通信复杂度不依赖于客户端的个数和函数的深度, 并且安全性依赖于一般假设.

多服务器的可验证计算是指客户端将计算任务委托给多个服务器. Chaum 等 [29] 定义了 "wallets with observer". 这是一种由第三方安装在客户端上的安全硬件, 它帮助客户端做复杂的计算但客户端并不需要信任这个硬件, 客户端通过分析硬件与第三方的通信记录来验证计算结果的正确性. Hohenberger 等 [53] 形式化定义了这一模型, 并构造了适用于模幂乘运算的协议, 协议使用两个服务器将模幂运算的复杂性降为 $O(\log^2 n)$, 但是该协议无法抵抗合谋攻击. 随后文献 [54~56] 4) 改进了 Hohenberger 等 [53] 的结果, 使得计算更加高效并且安全性能更好.

Atallah 等 [57] 首次提出了科学计算的安全外包问题, 例如矩阵乘法、矩阵求逆和差分方程等等. 最近, 文献 [58,59] 分别提出了安全的外包矩阵乘算法. 前者的安全性基于没有合谋的服务器, 后者利用秘密分享技术达到了可证明安全性, 但是这个算法仍然需要客户端作大量的计算. 随后文献 [2,60,61] 改进了他们的结果, 使得客户端的效率更高. 为了解大规模的线性规划 $Ax = b$ 问题, Wang 等 [62] 利用 Jacobi 方法的迭代思想和语义安全的加法同态加密构造了可验证安全的外包协议, 然而他们的方案得到的只是一个近似解, 随后有大量的工作基于求解线性规划问题 [63~67].

Zhang 等 [68] 提出了一个新的多服务器局部可验证计算 (verifiable local computation) 模型, 在这个模型中, 可以外包数据模块 $m = (m_1, \ldots, m_n)$ 给多个服务器然后验证任意模块的计算. 在合谋的服务器数量小于阈值时, 他们的方案可以达到数据隐私和安全性. 文献 [68] 构造了两个方案, 一个方案仅基于伪随机函数, 非常高效; 另一个方案使用了双线性映射, 效率建立在分摊的意义上.

有时多服务器的可验证计算模式可能会消除单服务器模式下的弊端, 目前基于密码学手段的单服务器的可验证计算方案中, 一个很大的弊端是需要有预处理阶段, 而且预处理阶段的时间复杂度与计算函数本身是差不多的. 这就要求客户端有能力自己计算函数或者客户端委托给可信服务器作预处理. 此外, 方案的效率建立在分摊的意义上, 就要求服务器对同一个函数做多次操作. Ananth 等 [69] 构造了多服务器的可验证计算方案, 该方案不需要预处理阶段, 并且方案在不使用 FHE 的条件下做到了输入隐私. 但是方案的安全性要求至少有一个服务器是诚实的, 所以方案不能抵抗合谋攻击.

3) Gordon S D, Katz J, Liu F H, et al. Multi-client verifiable computation with stronger security guarantees. IACR Cryptology ePrint Archive, 2015: 142.

4) Kiraz M S, Uzunkol O. Efficient and verifiable algorithms for secure outsourcing of cryptographic computations. IACR Cryptology ePrint Archive, 2014: 748.

3.3 基于同态 MAC/签名的可验证计算

目前为止我们介绍的可验证计算都是由于客户端的计算能力有限, 需要把 $F(x)$ 委托给服务器. 但是存在另外一种情形: 客户端的存储能力有限使得它不得不把计算委托给服务器. 由于客户端无法存储外包的函数或者输入, 所以之前的方案不能适用于这种情形.

同态消息认证码使得服务器可以对认证数据做任意操作并产生一个短的标签来证明计算结果的正确性. 具体描述如下, 客户端需要计算 $F(D)$, 而存储 D 需要占用很大的空间, 客户端把数据 $D = D_1D_2\cdots D_n$ 和 $t_i = \mathrm{MAC}_k(D_i)$ 传给服务器, 由服务器存储, 客户端只存储私钥 k. 服务器计算 $y = F(D)$ 和标签 t 并传给客户端, 当且仅当 $t = \mathrm{MAC}_k(y)$ 时客户端接受 y.

Gennaro 等 [70] 形式化定义了同态消息认证 (fully message homomorphic authenticator) 方案, 其中涉及带标签的程序, 先给出说明.

带标签的程序 (labeled-program) 是指由标签来定义程序的输入, 一个带标签的程序 $\mathcal{P} = (f, \tau_1, \ldots, \tau_k)$ 包含一个电路 $f : \{0,1\}^k \to \{0,1\}$ 和一组各不相同的输入标签 $(\tau_1, \ldots, \tau_k)$, τ_i 表示第 i 个输入的标签.

定义7 (同态消息认证 [70]) 一个同态消息认证方案 HMAC=(KeyGen, Auth, Ver, Eval) 由以下定义的 4 个算法构成.

• KeyGen. 密钥生成算法是一个随机算法, 输入安全参数 λ, 输出一个私钥 sk 和一个计算密钥 evk.

• Auth. 算法输入私钥 sk, 认证的比特 $b \in \{0,1\}$ 以及 b 的标记 $\tau \in \{0,1\}^*$, 输出一个标签 σ.

• Eval. 确定性算法, 以标签向量 $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_k)$ 和电路 $f : \{0,1\}^k \to \{0,1\}$ 作为输入, 输出一个新的标签 ψ. 如果 σ_i 是带标签的程序 $\mathcal{P}_i$ 的输出 b_i 的标签, 则 ψ 是组合程序 $\mathcal{P}^* = f(\mathcal{P}_1, \ldots, \mathcal{P}_k)$ 的输出 $b^* = f(b_1, \ldots, b_k)$ 的标签.

• Ver. 确定性验证算法, 使用标签 ψ 检验 $e \in \{0,1\}$ 是否是程序 $\mathcal{P}$ 关于已认证的数据的输出.

一个同态 MAC 方案需要满足的最基本的条件是安全性: 一个概率多项式时间的敌手不能伪造一个有效的 MAC. 一个方案是可组合的是指, 部分认证计算结果的输出可以作为下次计算的输入. 也就是说, 如果标签 $\psi_1, \ldots, \psi_t$ 分别认证标记程序 $\mathcal{P}_1, \ldots, \mathcal{P}_t$ 的输出 $b_1, \ldots, b_t$, 则标签 $\psi^* = \mathrm{Eval}(\mathcal{P}^*, \psi_1, \ldots, \psi_t)$ 应当认证组合程序的输出 $b^* = \mathcal{P}^*(b_1, \ldots, b_t)$. 方案的标签是简短的是指标签的规模 (tag-size) 具有一个关于安全参数 λ 的多项式的界, 和计算电路的规模 (size) 或者输入的长度无关.

Agrawal 等 [71] 构造了一个只适用于线性函数计算的同态消息认证方案. Gennaro 等 [70] 利用全同态加密构造了适合任意函数的同态 MAC 方案, 但该方案只是解决了计算的正确性和验证的可靠性问题, 验证效率非常低, 并且每次只能对单个比特进行认证, 因此, 仅是理论可行并不具有实际可操作性. 之后的很多研究都致力于提高同态消息认证方案的效率.

Catalano 等 [72] 构造了低次多项式的同态 MAC 方案, 在生成标签阶段只需要计算一个拟随机函数, 验证阶段的复杂度与函数 f 的长度和次数有关. 文献 [72] 中的第一个方案依赖于单向函数的存在性, 该方案使用一阶多项式 $y(z) \in \mathbb{Z}_p[z]$ 认证消息 $m \in \mathbb{Z}_p$, y 满足 $y(0) = m$, $y(x) = r_\tau$, 其中 $r_\tau = F_x(\tau)$ 是一个拟随机函数值, x 是私钥. 设 f 是一个算术电路, 计算加法门和乘法门时, 分别做多项式的加法和乘法. 可以观察到同态性质: 如果有两个标签 $y^{(1)}, y^{(2)}$(多项式的系数) 使得 $y^{(1)}(0) = m_1$, $y^{(2)}(0) = m_2$, 则对 $y = y^{(1)} + y^{(2)}$ 或 $y = y^{(1)} \cdot y^{(2)}$, 我们可以得到 $y(0) = m_1 + m_2$ 或 $y(0) = m_1 \cdot m_2$. 在随机点 x 处, 同样可以得到 $y(x) = r_{\tau_1} + r_{\tau_2}$ 或 $y(x) = r_{\tau_1} \cdot r_{\tau_2}$. 通过对整个电路 f 的计算, 就可以得到带标记程序 $\mathcal{P} = (f, \tau_1, \ldots, \tau_n)$ 的输出 m' 的标签 y, 通过验证 $m' = y(0)$ 和 $f(r_{\tau_1}, \ldots, r_{\tau_n}) = y(x)$, 可以验证计

表 1 同态 MAC 方案效率及性质比较

Table 1 Comparison of homomorphic MACs

Scheme	Tag size	Composability	Supported computation	Assumption	Verify queries
GW13 [70]	$O(\lambda)$	Yes	Arbitrary Circuits	FHE	No
CF13-1 [72]	$O(d)$	Yes	Degree$-d$ circuits	OWF	Yes
CF13-2 [72]	$O(1)$	No	Degree$-D$ circuits	D-DHI	Yes
ZS [73]	$O(d)$	Yes	Degree$-d$ circuits	l-DL	Yes
BFR [42]	$O(1)$	Yes	Degree-2 circuits	DL	Yes
CFGN [74]	$O(1)$	Yes	Degree$-k$ circuits $\forall k: \frac{k}{p} < \frac{1}{2}$	$(1,k)$-MDHI	Yes

算结果的正确性. 该方案支持标签的任意组合, 但是生成标签的规模会随着多项式次数的增大而增加, 因此只适用于次数较低的多项式. 文献 [72] 中第二个方案通过把计算移到指数上解决了这个问题, 生成的标签较短, 但不支持标签的任意组合.

Catalano 等 [72] 提出的方案中, 验证函数 f 对应的标签需要计算 $W = f(r_{\tau_1}, \ldots, r_{\tau_n})$, 这将花费和直接计算 f 一样多的时间. Backes 等 [42] 针对这个问题提出了高效同态 MAC 概念, 他要求运行验证算法要比计算函数更高效. 方案 [42] 的主要思想有两点: (1) 阐明了一个实际、安全、可再次使用的标记的模型; (2) 构造了一个拟随机函数, 利用这个拟随机函数可以预先计算一个和部分标签相独立的值 ω_f, 使得通过利用 ω_f 可以高效地计算 W. 这个方案的效率是建立在分摊的意义上的, 但他们的方案仅限于阶数小于等于 2 的多项式. 基于判定线性假设该文章提出了一个原始的构造, 并表明了算法的实用性. Zhang 等 [73] 改进了他们的方案, 针对阶数大于 2 的多项式, 提出了一个新的解决方案. Catalano 等 [74] 利用分级编码技术构造了一个新的同态 MAC, 支持任意多项式深度电路的计算. 表 1 中对比了上文中提到的同态 MAC 方案.

同态签名的概念最早由 Johnson 等在文献 [75] 中提出, 利用同态签名方案, 客户端可以将数据和签名都存储在服务器, 任何人都可以验证服务器返回的计算结果的正确性. 此后若干年内人们构造了大量的允许线性函数计算的同态签名方案 [76~84]. Boneh 等 [85] 构造了第一个允许确定阶数多项式函数计算的同态签名方案, 但是 Boneh 和 Freeman 的构造是基于格的而且它是在随机应答器模型下可证明是安全的.

Catalano 等 [86] 采用分层分级编码的技术构造了新的适用于确定阶数多元多项式的同态签名方案, 方案的可证明安全性不需要在随机应答器模型下证明, 且提高了签名的验证效率. 构造的基本思想是: 对消息 m 的签名是分级编码第一层的元素, 其形式为

$$(\Lambda = g^{(r-mx)b}, \Gamma = g^{(r-mx)ab}),$$

其中, g^r 是公共信息, a, b 是签名私钥, g, g^x, g^b, g^{abx} 是验证公钥. 给定原始消息 m_1, m_2 的签名 (Λ_1, Γ_1), (Λ_2, Γ_2) 时, 计算 $m_1 + m_2$ 的签名为 $(\Lambda = \Lambda_1 \cdot \Lambda_2, \Gamma = \Gamma_1 \cdot \Gamma_2)$. 计算 $m_1 \cdot m_2$ 时, 应用多线性映射计算

$$e(\Lambda_1, \Gamma_2) = g_2^{[r_1 r_2 - (r_1 m_2 + r_2 m_1)x + m_1 m_2 x^2]ab^2},$$

其中, g_2 是群 $\mathbb{G}_2$ 的生成元. 这就出现了一个问题, 每做一次乘法运算, 指数上中间项 $(r_1 m_2 + r_2 m_1)x$ 都会变长一次. 通过 g^{abx} 可以 "清除" 中间项, 也就是计算

$$e(\Lambda_1, \Gamma_2) \cdot e(\Lambda_1, g^{abxm_2}) \cdot e(g^{abxm_1}, \Lambda_2)$$

得到 $g_2^{[r_1r_2-m_1m_2x^2]ab^2}$. 通过这种方法可以获得原始消息经过多项式运算得到的消息 $f(\boldsymbol{m})$ 的签名 $g_i^{[f(\boldsymbol{r})-f(\boldsymbol{m})x^i]a^{i-1}b^i}$, 其中 i 是多项式 f 的次数, g_i 是 $\mathbb{G}_i$ 的生成元, $\boldsymbol{m}$ 是原始消息向量, $\boldsymbol{r}$ 是关于公共信息 g^{r_i} 的指数 r_i 的向量. 该方案的潜在缺点有两个: (1) 分级编码技术的运算效率比较低; (2) 相应的安全性假设需要更长时间的密码分析才能最终确立其可信性. 能够克服上述两个缺点的新的构造, 是研究的热点之一.

4 总结

综上所述, 由于采用的研究手段和工具不同, 不同研究团体的可验证计算方案形成了各自不同的特色. 基于计算复杂性理论的方案不需要预处理, 但是方案多是交互式的, 非交互的方案安全性是在随机应答器模型下或者基于不可证伪的假设, 此外方案不考虑数据的隐私性. 基于密码学手段的可验证计算方案大部分需要预处理, 预处理的时间复杂度与计算函数相当, 所以效率都是建立在分摊的意义上. 方案会采用全同态加密、加法 (或乘法) 同态加密和伪装技术等去保护数据的隐私性.

虽然可验证计算的研究已经取得了很多重要成果, 但由于具有非常重要的实际应用背景, 该研究方向还需要进行大量的研究工作. 本文已经指出了一些研究内容, 下面几点也是可验证计算未来研究的热点方向.

• 实用的可验证计算方案设计. 研究可验证计算的目的就是为了满足现实需求. 尽管已经存在一些针对特殊函数的较实用的方案, 但是如何提高方案的效率, 以及如何构造更广泛函数类的实用方案, 仍然是今后长期的重要的研究方向.

• 发展新的模型. 针对不同的应用场景, 提出不同的模型, 并构造出与之相应的解决方案. 例如, 对于服务器返回的错误结果, 建立一种既能够公开服务器的不诚实行为, 又使得服务器无法否认自己的不诚实行为的机制是实际中非常必要的.

• 利用特殊函数的可验证计算方案, 解决更多应用问题.

参考文献

1 Sun Microsystems, Inc. Building Customer Trust in Cloud Computing with Transparent Security. 2009

2 Hu X, Pei D Y, Tang C M, et al. Verifiable and secure outosurcing of matrix calculation and its application. Sci Sin Inform, 2013, 43: 842–852 [胡杏, 裴定一, 唐春明, 等. 可验证安全外包矩阵计算及其应用. 中国科学: 信息科学, 2013, 43: 842–852]

3 Tang C M, Hu X. Advances in research of outsourcing computation. Dev Rep Chin Cryptol, 2012. 59–79 [唐春明, 胡杏. 外包计算的研究进展. 中国密码学发展报告, 2012. 59–79]

4 Belenkiy M, Chase M, Erway C C, et al. Incentivizing outsourced computation. In: Proceedings of the Workshop on Economics of Networked Systems (NetEcon), New York, 2008. 85–90

5 Monrose F, Wyckoff P, Rubin A D. Distributed execution with remote audit. In: Proceedings of the 1999 ISOC Network and Distributed System Security Symposium (NDSS), San Diego, 1999. 103–113

6 Smith S, Weingart S. Building a high-performance, programmable secure coprocessor. Comput Netw, 1999, 31: 831–960

7 Yee B, Tygar J D. Secure coprocessors in electronic commerce applications. In: Proceedings of the 1st USENIX Workshop on Electronic Commerce, New York, 1995

8 Goldwasser S, Micali S, Rackoff C. The knowledge complexity of interactive proof-systems. SIAM J Comput, 1989, 18: 186–208

9 Babai L. Trading group theory for randomness. In: Proceedings of the ACM Sympositum on the Theory of Computing (STOC), New York, 1985. 421–429

10 Goldwasser S, Kalai Y T, Rothblum G N. Delegating computation: interactive proofs for muggles. In: Proceedings of the ACM Sympositum on the Theory of Computing (STOC), New York, 2008. 113–122
11 Cormode G, Mitzenmacher M, Thaler J. Practical verified computation with streaming interactive proofs. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS), New York, 2012. 90–112
12 Fortnow L, Rompel J, Sipser M. On the power of multi-prover interactive protocols. Theory Comput Sci, 1994, 134: 545–557
13 Arora S, Safra S. Probabilistic checking of proofs: a new characterization of NP. J ACM, 1998, 45: 70–122
14 Shamir A. IP=PSPACE. JACM, 1992, 39: 869–877
15 Arora S, Lund C, Motwani R, et al. Proof verification and the hardness of approximation problems. J ACM, 1998, 45: 501–555
16 Kilian J. Improved efficient arguments. In: Proceedings of CRYPTO. Berlin: Springer, 1995. 311–324
17 Micali S. CS proofs (extended abstract). In: Proceedings of the IEEE Symposium on Foundations of Computer Science, New York, 1994. 436–453
18 Bitansky N, Canetti R, Chiesa A, et al. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, New York, 2012. 326–349
19 Groth J. Short pairing-based non-interactive zero-knowledge arguments. In: Proceedings of ASIACRYPT. Berlin: Springer, 2010. 321–340
20 Blum M, Santis A D, Micali S, et al. Noninteractive zero-knowledge. SIAM J Comput, 1991, 20: 1084–1118
21 Lipmaa H. Progression-free sets and sublinear paring-based non-interactive zero-knowledge arguments. In: Proceedings of TCC. Berlin: Springer, 2012. 169–189
22 Gennaro R, Gentry C, Parno B, et al. Quadratic span programs and succinct NIZKS without PCPs. In: Proceedings of EUROCRYPT. Berlin: Springer, 2013. 626–645
23 Parno B, Gentry C. Pinocchio: nearly practical verifiable computation. In: Proceedings of the 2013 IEEE Symposium on Security and Privacy (S&P), Washington, 2013. 238–252
24 Sasson E B, Chiesa A, Genkin D, et al. SNARKs for C: verifying program executions succinctly and in zero knowledge. In: Proceedings of CRYPTO. Berlin: Springer, 2013. 90–108
25 Canetti R, Riva B, Rothblum G N. Practical delegation of computation using multiple servers. In: Proceedings of the 18th ACM Conference on Computer and Communications Security, New York, 2011. 445–454
26 Feige U, Kilian J. Making games short. In: Proceedings of the 29th Annual ACM Symposium on Theory of Computing, New York, 1997. 506–516
27 Canetti R, Riva B, Rothblum G N. Two protocols for delegation of computation. In: Proceedings of Information Theoretic Security. Berlin: Springer, 2012. 37–61
28 Canetti R, Riva B, Rothblum G N. Refereed delegation of computation. Inf Comput, 2013, 226: 16–36
29 Chaum D, Pedersen T. Wallet databases with observers. In: Proceedings of CRYPTO. Berlin: Springer, 1993. 89–105
30 Gennaro R, Gentry C, Parno B. Non-interactive verifiable computing: outsourcing computation to untrusted workers. In: Proceedings of CRYPTO. Berlin: Springer, 2010. 465–482
31 Yao A. Protocols for secure computations. In: Proceedings of the 23rd Annual Symposium on Foundations of Computer Science, Washington, 1982. 160–164
32 Bellare B M, Hoang V T, Rogaway P. Foundations of garbled circuits. In: Proceedings of the 2012 ACM Conference on Computer and Communications Security, New York, 2012. 784–796
33 Gentry C. A fully homomorphic encryption scheme. Dissertation for Ph.D. Degree. Stanford: Stanford University, 2009
34 Gentry C. Fully homomorphic encryption using ideal lattices. In: Proceedings of the ACM Symposium on the Theory of Computing (STOC), New York, 2009. 169–178
35 Chung K M, Kalai Y, Vadhan S. Improved delegation of computation using fully homomorphic encryption. In: Proceedings of CRYPTO. Berlin: Springer, 2010. 483–501
36 Barak B, Goldreich O. Universal arguments and their applications. In: Proceedings of 17th IEEE Annual Conference on Computational Complexity, Washington, 2002. 194–203
37 Applebaum B, Ishai Y, Kushilevitz E. From secrecy to soundness: efficient verification via secure computation. In:

Proceedings of 37th International Colloquium in Automata, Languages and Programming. Berlin: Springer, 2010. 152–163
38 Barbosa M, Farshim P. Delegatable homomorphic encryption with applications to secure outsourcing of computation. In: Proceedings of CT-RSA. Berlin: Springer, 2012. 296–312
39 Benabbas S, Gennaro R, Vahlis Y. Verifiable delegation of computation over large datasets. In: Proceedings of the 31st Annual Conference on Advances in Cryptology. Berlin: Springer, 2011. 111–131
40 Fiore D, Gennaro R, Pastro V. Efficiently verifiable computation on encrypted data. In: Proceedings of the 2014 ACM Conference on Computer and Communications Security, New York, 2014. 844–855
41 Brakerski Z, Gentry G, Vaikuntanathan V. (Leveled) fully homomorphic encryption without bootstrapping. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, New York, 2011. 309–325
42 Backes M, Fiore D, Reischuk R M. Verifiable delegation of computation on outsourced data. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer Communications Security, New York, 2013. 863–874
43 Zhang L F, Safavi-Naini R. Private outsourcing of polynomial evaluation and matrix multiplication using multilinear maps. In: Proceedings of Cryptology and Network Security. Berlin: Springer, 2013. 329–348
44 Parno B, Raykova M, Vaikuntanathan V. How to delegate and verify in public: verifiable computation from attribute-based encryption. In: Proceedings of Theory of Cryptography. Berlin: Springer, 2012. 422–439
45 Fiore D, Gennaro R. Publicly verifiable delegation of large polynomials and matrix computations, with applications. In: Proceedings of the 2012 ACM Conference on Computer and Communications Security, New York, 2012. 501–512
46 Catalano D, Fiore D, Gennaro R, et al. Algebraic (trapdoor) one-way functions and their applications. In: Proceedings of Theory of Cryptography. Berlin: Springer, 2013. 680–699
47 Papamanthou C, Shi E, Tamassia R. Signatures of correct computation. In: Proceedings of Theory of Cryptography. Berlin: Springer, 2013. 222–242
48 Zhang L F, Safavi-Naini R. Verifiable delegation of computations with storage-verification trade-off. In: Proceedings of ESORICS. Berlin: Springer, 2014. 112–129
49 Choi S G, Katz J, Kumaresan R, et al. Multi-client non-interactive verifiable computation. In: Proceedings of Theory of Cryptography. Berlin: Springer, 2013. 499–518
50 Goldwasser S, Goyal V, Jain A, et al. Multi-input function encryption. In: Proceedings of EUROCRYPT. Berlin: Springer, 2014. 578–602
51 Boneh D, Lewi K, Raykova M, et al. Semantically secure order-revealing encryption: multi-input functional encryption without obfuscation. In: Proceedings of 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 2015. 563–594
52 Goldwasser S, Kalai Y, Popa R A, et al. Reusable garbled circuits and succinct functional encryption. In: Proceedings of the ACM Sympositum on the Theory of Computing (STOC), New York, 2013. 1–10
53 Hohenberger S, Lysyanskaya A. How to securely outsource cryptographic computations. In: Proceedings of Theory of Cryptography. Berlin: Springer, 2005. 264–282
54 Chen X F, Li J, Ma J F, et al. New algorithms for secure outsourcing of modular exponentiations. In: Proceedings of ESORICS. Berlin: Springer, 2012. 541–556
55 Liu J, Yang B, Du Z G. Outsourcing of verifiable composite modular exponentiations. In: Proceedings of the 5th International Conference on Intelligent Networking and Collaborative Systems, Washington, 2013. 546–551
56 Ma X, Li J, Zhang F G. Outsourcing computation of modular exponentiations in cloud computing. Cluster Comput, 2013, 16: 787–796
57 Atallah M J, Pantazopoulos K N, Rice J R, et al. Secure outsourcing of scientific computations. Adv Comput, 2002, 54: 216–272
58 Benjamin D, Atallah M J. Private and cheating-free outsourcing of algebraic computations. In: Proceedings of 6th Annual Conference on Privacy, Security and Trust, New York, 2008. 240–245
59 Atallah M J, Frikken K B. Securely outsourcing linear algebra computations. In: Proceedings of 5th ACM Symposium on Information, Computer and Communications Security, New York, 2010. 48–59
60 Lei X, Liao X, Huang T, et al. Outsourcing large matrix inversion computation to a public cloud. IEEE Trans Cloud Comput, 2013, 1: 78–87
61 Lei X, Liao X, Huang T, et al. Achieving security, robust cheating resistance, and high-efficiency for outsourcing large

matrix multiplication computation to a malicious cloud. Inf Sci, 2014, 280: 205–217
62 Wang C, Ren K, Wang J, et al. Harnessing the cloud for securely solving large-scale systems of linear equations. In: Proceedings of 31st International Conference on Distributed Computing Syestems, Minneapolis, 2011. 549–558
63 Dieier J, Kerschbaum F. Practical privacy-preserving multiparty linear programming based on problem transformation. In: Proceedings of 3th International Conference on Social Computing, New York, 2011. 916–924
64 Hong Y, Vaidya J. An inference-proof approach to privacy-preserving horizontally partitioned linear programs. Optim Lett, 2014, 8: 267–277
65 Li W, Li H, Deng C. Privacy-preserving horizontally partitioned linear programs with inequality constraints. Optim Lett, 2013, 7: 137–144
66 Nie H, Chen X, Li J, et al. Efficient and verifiable algorithm for secure outsourcing of large-scale linear programming. In: Proceedings of 28th International Conference on Advanced Information Networking and Applications, Victoria, 2014. 591–596
67 Wang C, Ren K, Wang J. Secure and practical outsourcing of linear programming in cloud computing. In: Proceedings of 30th IEEE International Conference on Computer Communications, New York, 2011. 820–828
68 Zhang L F, Safavi-Naini R, Liu X W. Verifiable local computation on distributed data. In: Proceedings of the 2nd International Workshop on Security in Cloud Computing, New York, 2014. 3–10
69 Ananth P, Chandran N, Goyal V, et al. Achieving privacy in verifiable computation with multiple servers-without FHE and without pre-processing. In: Proceedings of 17th International Conference on Practice and Theory in Public-Key Cryptography. Berlin: Springer, 2014. 149–166
70 Gennaro R, Wichs D. Fully homomorphic message authenticators. In: Proceedings of ASIACRYPT. Berlin: Springer, 2013. 301–320
71 Agrawal S, Boneh D. Homomorphic MACs: MAC-based integrity for network coding. In: Proceedings of the 7th International Conference on Applied Cryptography and Network Security. Berlin: Springer, 2009. 292–305
72 Catalano D, Fiore D. Practical homomorphic MACs for arithmetic circuits. In: Proceedings of EUROCRYPT 2013. Heidelberg: Springer, 2013. 336–352
73 Zhang L F, Safavi-Naini R. Generalized homomorphic macs with effcient verification. In: Proceedings of the 2nd ACM Workshop on ASIA Public-Key Cryptography, New York, 2014. 3–12
74 Catalano D, Fiore D, Gennaro R, et al. Generalizing homomorphic MACs for arithmetic circuits. In: Proceedings of Public Key Cryptography. Berlin: Springer, 2014. 538–555
75 Johnson R, Molnar D, Song D, et al. Homomorphic signature schemes. In: Proceedings of CT-RSA. Berlin: Springer, 2002. 244–262
76 Ahn J H, Boneh D, Camenisch J, et al. Computing on authenticated data. In: Proceedings of 9th Theory of Cryptography Conference in TCC. Berlin: Springer, 2012. 1–20
77 Attrapadung N, Libert B. Homomorphic network coding signatures in the standard model. In: Proceedings of 14th International Conference on Practice and Theory in Public Key Cryptography. Berlin: Springer, 2011. 17–34
78 Attrapadung N, Libert B, Peters T. Computing on authenticated data: new privacy definitions and constructions. In: Proceedings of 18th International Conference on the Theory and Application of Cryptology and Information Security. Berlin: Springer, 2012. 367–385
79 Attrapadung N, Libert B, Peters T. Efficient completely context-hiding quotable and linearly homomorphic signatures. In: Proceedings of Public Key Cryptography. Berlin: Springer, 2013. 386–404
80 Boneh D, Freeman D M. Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In: Proceedings of Public Key Cryptography. Berlin: Springer, 2011. 1–16
81 Boneh D, Freeman D M, Katz J, et al. Signing a linear subspace: signature schemes for network coding. In: Proceedings of Public Key Cryptography. Berlin: Springer, 2009. 68–87
82 Catalano D, Fiore D, Warinschi B. Efficient network coding signatures in the standard model. In: Proceedings of Public Key Cryptography. Berlin: Springer, 2012. 680–696
83 Deiseroth B, Fehr V, Fischlin M, et al. Computing on authenticated data for adjustable predicates. In: Proceedings of the 11th International Conference on Applied Cryptography and Network Security. Berlin: Springer, 2013. 53–68
84 Gennaro R, Katz J, Krawczyk H, et al. Secure network coding over the integers. In: Proceedings of Public Key Cryptography. Berlin: Springer, 2010. 142–160

85 Boneh D, Freeman D M. Homomorphic signatures for polynomial functions. In: Proceedings of EUROCRYPT. Berlin: Springer, 2011. 149–168
86 Catalano D, Fiore D, Warinschi B. Homomorphic signatures with efficient verification for polynomial functions. In: Proceedings of CRYPTO. Berlin: Springer, 2014. 371–389

Progress in verifiable computation

XUE Rui[1*], WU Ying[1], LIU MuHua[1], ZHANG LiangFeng[2] & ZHANG Rui[1]

1 *State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing* 100093*, China*;
2 *School of Information Science and Technology, ShanghaiTech University, Shanghai* 200031*, China*
*E-mail: xuerui@iie.ac.cn

Abstract Verifiable computation is an important mechanism for ensuring the soundness (reliability) of function evaluation results produced when delegating computation in distributed computing or cloud computing. We summarize the important achievements about verifiable computation, in particular, from the perspectives of cryptography and computational theory. For verifiable computation in computational theory, we summarize the connection between interactive proofs, PCP theorems, and CS proofs and describe their development and applications. With regard cryptography, we discuss the state-of-the-art verifiable computation schemes that were constructed by applying cryptography tools. In addition, we survey verifiable computation schemes in the outsourced storage setting. Finally, we conclude with a discussion on the future development direction for verifiable computation.

Keywords verifiable computation, interactive proof, fully homomorphic encryption, homomorphic MAC, homomorphic signature

XUE Rui was born in 1963. He received his M.Sc. and Ph.D. in Mathematics from the Beijing Normal University. He is currently a full research professor and vice director at the State Key Laboratory of Information Security with the Institute of Information Engineering, Chinese Academy of Sciences. His research interests include information security and privacy in data and information systems, with a focus on public-key encryption and cryptographic protocols. Currently, he serves as the vice director member of security protocols association of the Cryptographic Association in China. He is also a member of the IEEE and a member of the ACM.

WU Ying was born in 1988. She is a Ph.D. candidate at the State Key Laboratory of Information Security with the Institute of Information Engineering, Chinese Academy of Sciences. Her research interests include, among others, cloud data security and security protocols.

LIU MuHua was born in 1987. He is a Ph.D. candidate at the State Key Laboratory of Information Security with the Institute of Information Engineering, Chinese Academy of Sciences. His research interests include cloud data security and security protocols.

ZHANG LiangFeng was born in 1982. He received the Ph.D. degree in cryptography and information security from the Nanyang Technological University, Singapore, in 2012. Currently, he is an assistant professor at the Shanghai Tech University. His research interests include modern cryptography and algebraic coding theory.