中国科学: 信息科学 2013 年 第 43 卷 第 7 期: 842–852
www.scichina.com info.scichina.com
《中国科学》杂志社
SCIENCE CHINA PRESS
论 文

可验证安全外包矩阵计算及其应用

胡杏[①②③], 裴定一[①②], 唐春明[①②*], Duncan S. WONG[④]

① 广州大学数学与信息科学学院, 广州 510006
② 广东数学与交叉科学省普通高校重点实验室, 广州 510006
③ 湖南科技大学数学与计算科学学院, 湘潭 411102
④ 香港城市大学计算机科学系, 香港
* 通信作者. E-mail: ctang@gzhu.edu.cn

收稿日期: 2013–01–11; 接受日期: 2013–05–15
国家自然科学基金 (批准号: 11271003)、香港特区自然科学基金 (批准号: CityU123511)、国家教育部博士点基金 (批准号: 20094410110001)、广东省高层次人才项目和广东省自然科学基金 (批准号: S2012010009950) 资助项目

摘要 矩阵计算在科学计算和密码学领域中都有着重要的作用. 许多密码协议、科学和数值计算问题都涉及到了矩阵计算. 然而, 对那些计算能力有限的用户来说, 独立完成矩阵计算并不是件容易的事情. 云计算拥有强大的计算资源, 它使得用户的计算能力不再受限于他们的资源约束型设备, 他们可以外包工作量给云. 本文围绕矩阵计算展开研究, 针对矩阵乘积、矩阵的行列式以及矩阵的逆这 3 种运算, 分别设计了切实可行的可验证安全外包协议. 与已有的关于这 3 种可验证外包计算的协议相比, 我们的协议在效率和安全性方面都有了改进, 而且我们的协议不需要任何的密码学假设. 本文中, 还为我们的协议给出两个具体应用, 即为 "大型线性方程组的求解" 以及 "基于纠错码的密码体制的实现" 这两个问题分别构造了高效的可验证外包计算协议.

关键词 云计算 外包计算 矩阵乘积 矩阵行列式 逆矩阵

1 引言

1.1 外包计算

外包计算是云计算模式的优点之一, 它使得云用户的计算能力不再受限于各自的资源约束型设备, 通过外包工作负载给云, 云用户可以使用云提供的无限资源来完成高代价的计算.

云计算是将计算任务分布在由大量计算机构成的资源池上, 使各种应用系统能够根据需要获取计算力、存储空间和各种软件服务. 我们可以把它看作是一片用于计算的、能提供超大规模计算资源的服务器集群. 它具有可靠性、通用性、可扩展性, 可灵活、动态地支撑千变万化的实际应用, 防止数据丢失、病毒入侵, 支持多终端和数据共享, 安全方便.

尽管云计算有众多好处, 但它特有的数据和服务外包、虚拟化、多租户和跨域共享等特点, 也给云计算带来了前所未有的安全性挑战. 由于云内部的操作细节对用户是不透明的 [1], 因此, 存在着各种动机, 使得云服务器的行为不诚实. 例如, 对需要大量计算资源的计算, 如果用户无法判断云计算输

引用格式: 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

出的正确性, 云可能会为了节约资源而 “偷懒”. 此外, 还可能存在软件 bug 和恶意的外部攻击, 这些都会影响计算结果的质量. 所以, 云计算环境中的隐私安全、内容安全是云计算研究的关键问题之一.

从应用角度出发, 一个有效的外包计算协议应该满足 3 个基本条件: (1) 确保用户数据的保密性; (2) 确保用户能够验证云计算输出的正确性; (3) 确保用户端在这个协议下需要的工作量 (包括正确性验证) 少于用户独自计算的工作量, 否则用户没有必要寻求云的帮助.

1.2 外包计算模型

本文考虑的外包计算模型中有两个参与实体, 即云用户和云服务器. 云用户拥有大量的计算昂贵的矩阵计算问题 (比如矩阵乘积、矩阵行列式以及矩阵求逆), 需要外包给云, 而云服务器拥有大量的计算资源.

为了给矩阵乘积、矩阵行列式以及矩阵求逆 3 种运算设计切实有效的可验证安全外包协议, 我们使用转换技术将原问题转换为随机问题, 同时保护敏感的 I/O 信息, 然后借助云服务器来求解随机问题, 并且能够有效验证由云服务器输出的计算结果.

这个模型面对的安全性威胁主要来自于云服务器的不诚实行为, 而这些行为又分为被动和主动的两种. 云服务器的被动行为主要是指控制云服务器的攻击者虽然忠实地执行协议, 但他企图利用执行协议中得到的信息来获得一些云用户的秘密信息. 而云服务器的主动行为主要是指控制云服务器的攻击者能力非常强大, 可以篡改数据、不按照协议的要求执行、甚至单方面终止协议. 在本文我们假设云服务器的行为是主动的.

一个外包协议被称为是高效的可验证协议, 如果它满足如下的 5 个性质:

(1) 正确性: 任何诚实的按照协议执行的云服务器产生的输出必能被用户接受.

(2) 合理性: 任何不诚实的云服务器产生的错误输出都不能以不可忽略的概率被用户接受.

(3) 隐私性: 在云服务器执行协议期间, 任何有关用户私有数据的敏感信息都不能被服务器推导出来.

(4) 可验证性: 用户能以极大概率验证诚实的云服务器输出的正确性, 也能以极大概率验证不诚实的云服务器输出的不正确性.

(5) 高效性: 外包协议中由用户端完成的本地计算量应充分小于用户独自解决原问题的计算量.

1.3 国内外研究现状

基于完全同态加密方案的研究. 2009 年, IBM 公司的 Gentry[2] 基于理想格首次构造出了完全同态加密, 这是一个具有里程碑意义的工作. 如果能够构造出高效的完全同态加密方案, 则任意的函数都能实现高效的可验证安全外包计算. 但是, 要想构造出高效的完全同态加密, 在最近几乎是不太可能的事情 (Schneier 指出, 他们的工作在理论上令人印象深刻, 但完全不切实际). 尽管文献 [3∼9] 都是改进完全同态加密的, 但是它们距离实用的完全同态加密方案, 还有很长的距离.

基于同态加密的矩阵计算. 2008 年, Atallah 等 [10] 使用语义安全的同态加密方案为昂贵的线性代数计算 (例如两个矩阵的乘积) 构造了可验证安全外包协议, 用户端的计算复杂度为 $O(n^2)$, n 是矩阵的大小. 另外, 该文提出的方案是基于 “两个不相互勾结的服务器” 这个假设的, 因而容易受到合谋攻击. 2010 年, Atallah 等 [11] 结合 Shamir 的密钥共享方案, 提出了一个单服务器的可验证外包协议, 解决了安全外包矩阵乘积 AB 的问题, 该协议的计算复杂度是 $O(t^2n^2)$, 其中 n 和 t 分别表示矩阵的大小和 Shamir 方案的门限值. 文献 [12] 基于 Yao's garbled circuits 和全同态加密方案, 提出了能计算任

意函数 F 的非交互的可验证外包方案, 客户端需要执行一个代价昂贵的预处理阶段, 其复杂度与函数 F 的 Boolean 电路复杂度有关.

为实现线性方程组 $Ax = b$ 的计算外包, Wang 等 [13] 基于 Jacobi 方法的迭代思想和语义安全的加法同态的加密方案, 为求解线性方程组 $Ax = b$ 构造了可验证安全外包协议, 虽然在他们的外包协议中, 用户端的计算复杂度是 $O(n^2)$, 但他们仅仅只能得到方程组的近似解, 而且对系数矩阵 A 也有一定的限制. 2011 年, Mohassel[14] 为 $n \times n$ 阶矩阵上的线性代数计算 (矩阵乘法、求逆、求矩阵行列式等), 提出了基于不同的同态加密方案 (例如 GM, Paillier's, El Gamal, BGN 或者 GHV) 的多个非交互的安全授权协议, 用户端的计算量至多是 $O(n^2 \log n)$, 而且他们的协议需要密码学假设.

文献 [15] 中也给出了一个矩阵乘法的可公开验证的外包协议. 该协议使用了一个分摊模型 (amortized model), 当用户将一个大规模 $(n \times d)$ 的矩阵存储于云服务器时, 用户端需要执行一个复杂度为 $O(nd)$ 的一次性的昂贵计算, 从而加快了矩阵乘法的验证 (复杂度为 $O(nd)$). 而且, 他们的协议还依赖于双线性映射和伪随机函数的使用.

基于伪装技术的外包计算研究. 除了使用密码学工具来构造安全外包协议外, 另外一种构造安全外包计算协议的主要工具是伪装技术 (或者盲化技术)[16~19]. 2001 年, Atallah 等 [16] 首次研究了数值计算和科学计算的安全外包问题, 他们提出了许多适合于科学计算 (例如, 矩阵乘法、不等式、线性方程组等) 的伪装技术, 这些技术确保了用户数据的隐私性, 但该文中并未解决计算结果的可验证性问题. 2005 年, Hohenberger 等 [17] 构造了使用两个不勾结的服务器来完成模素数的求幂运算的安全外包协议. 2012 年, Seitkulov[19] 提出了一些新的可验证的伪装方法, 解决了抽象方程、带秘密参数的 Cauchy 问题、带秘密边界条件的边值问题以及一些非线性方程的可验证安全外包问题.

1.4 本文贡献

我们使用伪装 (盲化) 技术, 为矩阵计算 (包括矩阵乘积、矩阵行列式、矩阵的逆) 构造了高效的可验证安全外包计算协议. 这些协议既保证了矩阵计算问题中输入/输出的隐私性、计算结果的可验证性, 也保证了高效性, 因为我们的协议的计算复杂度仅仅是 $O(n^2)$ (注意到, 问题的输入规模也是 $O(n^2)$).

值得强调的是, 我们首次为计算矩阵行列式和矩阵的逆构造了在云计算中高效可验证的外包计算协议.

使用我们的外包计算协议, 我们为求解线性方程组 $Ax = b$ 构造了一个高效的外包计算协议, 同时也高效地实现了基于编码的密码体制的外包计算.

1.5 本文结构

本文第 2 节研究了矩阵乘积的可验证的安全外包协议. 第 3 节提出了一个新的求矩阵行列式的可验证安全外包协议. 第 4 节研究的是矩阵求逆的问题. 第 5 节将我们的结果与现有的方案做了一个比较分析. 第 6 节给出了可验证安全外包矩阵计算协议分别在代数计算和基于编码的密码体制中的应用.

2 矩阵乘积

这一节描述两个矩阵乘积的可验证安全外包协议 MP. 文献 [16] 中提出了关于两个方阵乘积的安

全外包方案, 在该方案中, 用户无法判断服务器的输出是否正确. 我们的协议 MP 是两个方阵乘积的可验证安全外包协议, 是对文献 [16] 中的安全外包方案的一个本质上的改进.

目标 MP: 设 M_1, M_2 是 F_q 上的两个 $n\times n$ 阶矩阵, M_1, M_2 都是秘密的. 求两矩阵乘积 M_1M_2. 则令

$$\delta_{x,y}=\begin{cases}1, & x=y,\\ 0, & x\neq y.\end{cases}$$

可验证安全外包协议 MP 描述如下.

可验证安全外包协议 MP:

Step1: 生成验证值.

随机选择两个数 $1\leqslant i,j\leqslant n$. 取 M_1 的第 i 行 $(a_{i_1},\ldots,a_{i_n})$ 和 M_2 的第 j 列 $(b_{1j},\ldots,b_{nj})^{\mathrm{T}}$, 计算 $c=\sum_{t=1}^{n}a_{it}b_{tj}$.

$m_{i,j}=M_1M_2(i,j)=c$ 就是验证值, 它是秘密的且保存在用户端.

Step2: 盲化 M_1 和 M_2. 用户完成以下子步骤:

1) 构造 6 个随机置换 $\pi_1,\ldots,\pi_6$, 其中 $\pi_i\in\{1,\ldots,n\}$,$i=1,2,\ldots,6$.

2) 构造 6 个非零随机数集合 $\{a_1,\ldots,a_n\}$, $\{b_1,\ldots,b_n\}$, $\{c_1,\ldots,c_n\}$, $\{d_1,\ldots,d_n\}$, $\{e_1,\ldots,e_n\}$, $\{f_1,\ldots,f_n\}$.

3) 生成 6 个矩阵 $P_1,\ldots,P_6$, 其中 $P_1(i,j)=a_i\delta_{\pi_1(i),j}$, $P_2(i,j)=b_i\delta_{\pi_2(i),j}$, $P_3(i,j)=c_i\delta_{\pi_3(i),j}$, $P_4(i,j)=d_i\delta_{\pi_4(i),j}$, $P_5(i,j)=e_i\delta_{\pi_5(i),j}$, $P_6(i,j)=f_i\delta_{\pi_6(i),j}$. 这些矩阵都是容易求逆的, 例如 $P_1^{-1}(i,j)=a_j^{-1}\delta_{\pi_1^{-1}(i),j}$.

4) 用户端计算 4 个矩阵 $X_1=P_1M_1P_2^{-1}$, $Y_1=P_2M_2P_3^{-1}$, $X_2=P_4M_1P_5^{-1}$, $Y_2=P_5M_2P_6^{-1}$.

Step3: 外包计算.

用户把 X_1,X_2,Y_1,Y_2 发送给服务器, 服务器计算 $Z_1=X_1Y_1$, $Z_2=X_2Y_2$, 并把 (Z_1,Z_2) 返回给用户.

Step4: 去盲.

用户端计算 $T_1=P_1^{-1}Z_1P_3$, $T_2=P_4^{-1}Z_2P_6$.

Step5: 验证.

若 1) $T_1=T_2$ 且 2) $T_k(i,j)=c$ $(k=1,2)$ 都成立, 说明服务器是诚实工作的, 用户端由 Z_1(或 Z_2) 得到正确的结果 $M_1M_2=T_1=P_1^{-1}Z_1P_3$; 否则, 用户端拒绝接受计算结果并终止协议.

定理 1 在单 — 云服务器模型中, 协议 MP 是两矩阵乘积的可验证安全外包实现.

证明 1) 正确性: 协议的正确性是显然的. 如果云服务器是诚实的按照协议执行的, 就一定能让用户接受它的输出.

2) 合理性: 来自云服务器的可能的不诚实行为有: (i) 改变乘积 M_1M_2 中某个元素的值. 在这种情况下, 云服务器想要改变 M_1M_2 中某个元素的值并通过 Step5 中验证式 $T_1=T_2$ 的验证, 就必须猜测出矩阵 P_1^{-1}, P_3, P_4^{-1}, P_6, 成功猜测出这 4 个矩阵的概率是 $\left(\frac{1}{n!}\right)^4\left(\frac{1}{q^n}\right)^4$. (ii) 选择一个数 $k\in F_q$, 然后返回 (kZ_1,kZ_2) 给用户. 虽然这样能使得验证式 $T_1=T_2$ 成立, 但由于验证值 c 是秘密的, 所以, 云服务器的这种不诚实行为能通过 Step5 中验证式 $T_k(i,j)=c$ $(k=1,2)$ 的概率是可忽略的.

3) 隐私性: 外包协议 MP 没有泄露 M_1、M_2 以及乘积 M_1M_2. 攻击者想要由 X_i 恢复出 M_1(或由 Y_i 恢复出 M_2), 就必须在 $(n!)^2$ 种可能的随机置换中准确的猜测出所用的两个置换, 并且找到使用的 $2n$ 个随机数.

令 X 表示 "攻击者成功恢复出某个私有数据" 这一事件, 则对一个 PPT 攻击者 $\mathcal{A}$, 有 $Pr[X] \leqslant \left(\frac{1}{n!}\right)^2\left(\frac{1}{q^n}\right)^2$.

4) 可验证性: 假设云服务器被一个 PPT 攻击者 $\mathcal{A}$ 所收买, 则云服务器收到的数据以及计算的最后结果 $\mathcal{A}$ 都会知道, 当选择合理性中描述的第 1 种攻击行为, 想要通过验证式 $T_1 = T_2$ 的验证, 就必须要猜测出所用的 4 个随机矩阵 $P_1^{-1}, P_3, P_4^{-1}, P_6$, 猜测成功的概率 $\left(\frac{1}{n!}\right)^4\left(\frac{1}{q^n}\right)^4$. 当选择合理性中描述的第 2 种攻击行为, 想要成功伪造验证值 c 的概率是可忽略的 (设 $P_i(i = 1, \ldots, 6)$ 中各元素都取自域 F_q). 因为 $c \in F_q$, 能猜测出验证值 c 的概率是 $\frac{1}{q}$, 猜测出验证值 c 是 M_1M_2 的第 (i, j) 位置上的元素 $m_{i,j}$ 的概率是 $\frac{1}{n^2}$, 猜测出元素 $m_{i,j}$ 是由 Z_1(或 Z_2) 中元素 z_{l_t} 经过变换 (行交换和列交换) 而来的概率是 $\frac{1}{n^2}$, 最后, 还要找到一个数 $r \in F_q$, 使得 $rz_{l_t} = c$, 能准确找到这个数 r 的概率是 $\frac{1}{q}$.

令 Y 表示 "攻击者伪造结果, 成功欺骗用户" 这一事件, 则对一个 PPT 攻击者 $\mathcal{A}$, 有 $Pr[Y] \leqslant \left(\frac{1}{n!}\right)^4\left(\frac{1}{q^n}\right)^4\left(\frac{1}{q^2}\right)\left(\frac{1}{n^4}\right)$.

综上所述, 协议 MP 是两矩阵乘积的可验证安全外包实现.

定理 2 在单 — 云服务器模型中, 协议 MP 是两矩阵乘积的高效实现.

证明 对两个 $n \times n$ 阶矩阵的乘积, 直接相乘的计算复杂度是 $O(n^3)$. 在文献 [16] 中也给出了安全外包两个 $n \times n$ 阶矩阵乘积的方案, 其计算复杂度为 $O(n^2)$, 但这个方案没有考虑如何验证服务器的输出. 我们的协议 MP 是可验证的安全外包协议. 在外包协议 MP 中, 用户端的主要计算量是在盲化和去盲阶段, 两个阶段的计算复杂度都是 $O(n^2)$, 另外, 在 Step1 中生成验证值的计算复杂度也是 $O(n^2)$.

综上所述, 协议 MP 是两矩阵乘积的高效实现.

对协议 MP 稍作修改就可得到非方阵乘积的可验证安全外包协议, 其安全性和高效性可分别由定理 1 和 2 推得.

3 矩阵行列式

这一节我们提出一个新方法来安全外包计算矩阵行列式.

目标 MD: 设 A 是 F_q 上的 $n \times n$ 阶矩阵, A 是秘密的, 求 $\det(A)$.

可验证安全外包协议 MD:

Step1: 随机选择一个 $1 \leqslant i \leqslant n$, 把 A 的第 i 行 (或第 i 列) 的每个元素都分解为 $m(1 < m < n-2)$ 个元素之和, 于是得到 $\det(A) = \det(A_1) + \cdots + \det(A_m)$.

Step2: 随机选择 $1 \leqslant i, j \leqslant n$ 且 $i \neq j$, 调用协议 MP 安全外包计算 A_iA_j, 记为 $A_{m+1} = A_iA_j$.

Step3: 盲化矩阵 A 以及 $A_i(i = 1, \ldots, m + 1)$. 用户完成以下子步骤:

1) 构造一个有 $2m + 4$ 个矩阵的集合 $P = \{P_1, \ldots, P_{2m+4}\}$. $P_k(k = 1, \ldots, 2m + 4)$ 的构造方法与前面提到的方法相同, 即 $P_k(i, j) = a_i^k\delta_{\pi_k(i),j}$, 其中 $\pi_k \in \{1, \ldots, n\}$ 都是随机置换, $\{a_1^k, \ldots, a_n^k\}$ 都是随机数. 这些矩阵都是容易求逆的, 例如 $P_1^{-1}(i, j) = (a_j^1)^{-1}\delta_{\pi_1^{-1}(i),j}$.

2) 对每个 $0 \leqslant s \leqslant m + 1$, 用户从集合 P 中随机选择两个矩阵 $\{P_l, P_t\} \subset P$, 分别记为 $P_l = P_l^s, P_t = P_t^s$, 计算矩阵 $B_s = P_l^sA_sP_t^s$, 其中 $A_0 = A$ (这里对每个 B_s 选用的集合 $\{P_l, P_t\}$ 是互不相交的).

Step4: 服务器计算行列式 $\det(B_0), \ldots, \det(B_{m+1})$ 并返回结果. 注意, $B_i(i = 0, \ldots, m + 1)$ 的发送次序是随机的且秘密的, 只有用户知道. 我们也要求云服务器按照发送次序返回对应的计算结果.

Step5: 验证. 如果下面两式

$$\begin{cases} \dfrac{\det(B_0)}{\det(P_l^0 P_t^0)} = \displaystyle\sum_{s=1}^{m} \dfrac{\det(B_s)}{\det(P_l^s P_t^s)}; \\ \dfrac{\det(B_{m+1})}{\det(P_l^{m+1} P_t^{m+1})} = \dfrac{\det(B_i)}{\det(P_l^i P_t^i)} \times \dfrac{\det(B_j)}{\det(P_l^j P_t^j)}, \end{cases}$$

同时成立, 则说明服务器是诚实工作的, 用户端由 $\det(B_0)$ 计算出 $\det(A)$; 否则, 用户端拒绝接受所有结果并终止协议.

定理 3 在单 — 云服务器模型中, 协议 MD 是求矩阵行列式的可验证安全外包实现.

证明 1) 正确性: 协议的正确性是显然的. 如果云服务器是诚实的按照协议执行的, 就一定能让用户接受它的输出.

2) 合理性: 来自云服务器的可能的不诚实行为是选择一个数 $k \in F_q$, 然后返回 $k\det(B_0), \ldots, k\det(B_{m+1})$ 给用户. 但由于 $B_i(i = 0, \ldots, m+1)$ 的发送次序是随机的且秘密的, 所以, 云服务器的这种不诚实行为能同时通过 Step5 中两个验证式的概率是可忽略的.

3) 隐私性: 外包协议 MP 没有泄露 A 和 $\det(A)$. 在 Step3 中对 A 和 $A_i(i = 1, \ldots, m)$ 进行了盲化, 攻击者若想要恢复出其中某个矩阵, 就必须在 $(n!)^2$ 种可能的随机置换中准确的猜测出所用的两个置换, 并且找到所用的随机数 a_i^k.

令 X 表示 "攻击者成功恢复出某个私有数据" 这一事件, 则对一个 PPT 攻击者 $\mathcal{A}$, 有 $Pr[X] \leqslant \left(\frac{1}{n!}\right)^2 \left(\frac{1}{q^n}\right)^2$.

另外, 用不同的矩阵分别盲化 A 和 $A_i(i = 1, \ldots, m)$, 在只使用了 1 个服务器的情况下, 这种处理方式有效地增加了攻击者企图通过 $A_s(s = 0, ..., m)$ 获得原问题相关信息的难度. 如果使用的是多个服务器, 这种方法就更加安全了.

4) 可验证性: 假设云服务器被一个 PPT 攻击者 $\mathcal{A}$ 所收买, 它想要成功欺骗用户, 就要在 $m+2$ 个行列式 $\det(B_0), \ldots, \det(B_{m+1})$ 中找出分别对应于 A 和 $A_i A_j$ 的两个行列式, 然后在剩余的 m 个行列式值中找出两个值 $\det(B_p), \det(B_q)$ 并找到一个数 $\alpha \in F_q$, 来满足 Step5 中的两个验证式.

令 Y 表示 "攻击者伪造结果, 成功欺骗用户" 这一事件, 则对一个 PPT 攻击者 $\mathcal{A}$, 有 $Pr[Y] \leqslant \left(\frac{1}{m+2}\right)\left(\frac{1}{m+1}\right)\left(\frac{1}{C_m^2}\right)\left(\frac{1}{q}\right)$.

综上所述, 协议 MD 是求矩阵行列式的可验证安全外包实现.

定理 4 在单 — 云服务器模型中, 协议 MD 是求矩阵行列式的高效实现.

证明 对 $n \times n$ 阶矩阵 A, 直接计算 $\det(A)$ 是困难的问题. 在我们的外包协议 MD 中, 用户端的计算量主要在盲化阶段, 盲化 A 和 $A_i(i = 1, \ldots, m)$ 的总计算量是 $O(n^2(m+2))$, 通过合适的选择 m, 这个计算量仍然是很小的. 另外, 调用可验证的安全外包协议 MP 安全外包计算 $A_i A_j$ 的计算复杂度是 $O(n^2)$.

综上所述, 协议 MD 是求矩阵行列式的高效实现.

4 矩阵的逆

这一节我们介绍方阵求逆的可验证安全外包协议.

目标 MI: 设 A 是 F_q 上的 $n \times n$ 阶矩阵, A 是秘密的, 求 A^{-1}.

可验证安全外包协议 MI:

Step1: 盲化矩阵 A. 用户完成以下子步骤:

1) 构造 4 个随机置换 $\pi_1,\ldots,\pi_4$, 其中 $\pi_i\in\{1,\ldots,n\}$, $i=1,\ldots,4$.

2) 构造 4 个非零随机数集合 $\{a_1,\ldots,a_n\}$, $\{b_1,\ldots,b_n\}$, $\{c_1,\ldots,c_n\}$, $\{d_1,\ldots,d_n\}$.

3) 生成 4 个矩阵 $P_1,\ldots,P_4$, 其中 $P_1(i,j)=a_i\delta_{\pi_1(i),j}$, $P_2(i,j)=b_i\delta_{\pi_2(i),j}$, $P_3(i,j)=c_i\delta_{\pi_3(i),j}$, $P_4(i,j)=d_i\delta_{\pi_4(i),j}$. 这些矩阵都是容易求逆的, 例如 $P_1^{-1}(i,j)=a_j^{-1}\delta_{\pi_1^{-1}(i),j}$.

4) 用户端计算 2 个矩阵 $X=P_1AP_2^{-1}, Y=P_3AP_4^{-1}$.

Step2: 把 X 和 Y 发送给服务器, 服务器计算 X^{-1} 和 Y^{-1}. 如果 X 和 Y 都可逆, 则返回 (X^{-1},Y^{-1}); 否则, 服务器返回 "X 不可逆", "Y 不可逆".

Step3: 为 X^{-1} 和 Y^{-1} 去盲. 用户端计算 $T_1=P_2^{-1}X^{-1}P_1$, $T_2=P_4^{-1}Y^{-1}P_3$.

Step4: 验证. 验证分为两个方面 1) $T_1=T_2$; 2) 用户随机选择一个数 $1\leqslant i\leqslant n$, 取 A 的第 i 行 $(a_{i1},\ldots,a_{in})$ (或第 i 列 $(a_{1i},\ldots,a_{ni})^{\mathrm{T}}$), 取 T_1(或 T_2) 的第 i 列 $(t_{1i},\ldots,t_{ni})^{\mathrm{T}}$(或第 i 行 $(t_{i1},\ldots,t_{in})$), 验证 $\sum_{j=1}^{n}a_{ij}t_{ji}=1$. 若 1) 和 2) 都成立, 说明服务器是诚实工作的, 用户端由 X^{-1} 或 Y^{-1} 得出 A^{-1}; 否则, 用户端拒绝接受并终止协议.

定理 5 在单 — 云服务器模型中, 协议 MI 是矩阵求逆的可验证安全外包实现.

证明 1) 正确性: 协议的正确性是显然的. 如果云服务器是诚实的按照协议执行的, 就一定能让用户接受它的输出.

2) 合理性: 来自云服务器的可能的不诚实行为有: (i) 改变 A^{-1} 中某个元素的值. 在这种情况下, 云服务器想要改变 A^{-1} 中某个元素的值并通过 Step4 中验证式 $T_1=T_2$ 的验证, 就必须猜测出矩阵 $P_2^{-1},P_1,P_4^{-1},P_3$, 成功猜测出这 4 个矩阵的概率是 $\left(\frac{1}{n!}\right)^4\left(\frac{1}{q^n}\right)^4$. (ii) 选择一个数 $k\in F_q$, 然后返回 (kX^{-1},kY^{-1}) 给用户. 由于 A 的秘密性以及 i 的随机性, 云服务器的这种不诚实行为能通过 Step4 中验证式 2) 的概率是可忽略的.

3) 隐私性: 外包协议 MI 没有泄露 A 和 A^{-1}. 在 Step1 中对 A 进行了盲化, 攻击者若想要由 X(或 Y) 恢复出 A, 就必须在 $(n!)^2$ 种可能的随机置换中准确的猜测出所用的两个置换, 并且找到所用的 $2n$ 个随机数.

令 X 表示 "攻击者成功恢复出某个私有数据" 这一事件, 则对一个 PPT 攻击者 $\mathcal{A}$, 有 $Pr[X]\leqslant\left(\frac{1}{n!}\right)^2\left(\frac{1}{q^n}\right)^2$.

另外, 用 4 个矩阵 $P_1,\ldots,P_4$ 分两次盲化 A 得到两个不同的盲化矩阵 X 和 Y, 在只使用了 1 个服务器的情况下, 这种处理方式有效地增加了攻击者企图通过 X 和 Y 获得原问题相关信息的难度. 如果使用的是多个服务器, 这种方法就更加安全了.

4) 可验证性: 假设云服务器被一个 PPT 攻击者 $\mathcal{A}$ 所收买, 当选择合理性中描述的第 1 种攻击行为, 想要通过验证式 $T_1=T_2$ 的验证, 就必须要猜测出所用的 4 个随机矩阵 $P_1^{-1},P_2,P_3^{-1},P_4$, 猜测成功的概率是 $\left(\frac{1}{n!}\right)^4\left(\frac{1}{q^n}\right)^4$. 当选择合理性中描述的第 2 种攻击行为, 想要伪造计算结果, 就必须确定出 A 的第 i 行 $(a_{i1},\ldots,a_{in})$ (或第 i 列 $(a_{1i},\ldots,a_{ni})^{\mathrm{T}}$), 攻击者的这种行为成功的概率是 $\left(\frac{1}{n}\right)\left(\frac{1}{q^n}\right)$.

令 Y 表示 "攻击者伪造结果, 成功欺骗用户" 这一事件, 则对一个 PPT 攻击者 $\mathcal{A}$, 有 $Pr[Y]\leqslant\left(\frac{1}{n}\right)\left(\frac{1}{n!}\right)^4\left(\frac{1}{q^n}\right)^5$.

综上所述, 协议 MI 是矩阵求逆的可验证安全外包实现.

定理 6 在单 — 云服务器模型中, 协议 MI 是矩阵求逆的高效实现.

证明 对 $n\times n$ 阶矩阵 A, 直接计算 A^{-1} 的时间复杂度是 $O(n^3)$. 在我们的外包协议 MI 中, 用户端的计算量主要在盲化和去盲阶段, 因此, 协议执行过程中加载于用户端的计算复杂度是 $O(n^2)$.

表 1 与其他协议的比较

Table 1 Comparison of other proposed protocols vs. ours

Protocol derived from	Operations on matrix	Complexity (the input size is $n \times n$)	Necessary for cryptographic assumptions	The number of server	Verfiable	Other problems and shortcomings
Ref.[10]	Multiplication	$O(n^2)$	Y	2	Y	Vulnerable to colluding attacks
Ref.[11]	Multiplication	$O(t^2n^2)$ $t^2 < n^2$ t is the threshold	Y	1 or 2	Y	High computation and communication complexity
Ref.[14]	Multiplication inversion determinant	$O(n^2logn)$	Y	1	Y	Their verification algorithm working with all but negligible probability
Ref.[15]	Multiplication	$O(n^2)$	Y	1	Y	The client invests an expensive pre-computation phase
Ref.[16]	Multiplication inversion	$O(n^2)$	N	1	N	–
Ours	Multiplication inversion determinant	$O(n^2)$	N	1	Y	–

综上所述, 协议 MI 是矩阵求逆的高效实现.

5 与现有方案的比较

我们把与现有方案的比较列表如表 1 所示.

从表 1 中可知, 我们的协议在没有任何密码学假设的情形下, 实现了复杂度最低的可验证外包计算协议.

6 应用

6.1 解线性方程组

在这一小节, 我们将使用第 4 节中给出的求矩阵逆的外包协议 MI 来判定 $Ax = b$ 是否有解, 如果有解, 就利用外包协议 MI 的返回结果来获得 $Ax = b$ 的解, 重要的是, 用我们的协议得到的是方程组的精确解.

令 A 是 $n \times n$ 阶矩阵, b 是 n 维列向量. 用户端想要获得方程组 $Ax = b$ 的一个解 x, 但又不想公开 A, b 以及 x.

Step1: 调用协议 MI 安全外包计算 A^{-1}.

Step2: 如果 A 可逆, 则协议 MI 会返回 A^{-1}, 用户计算 $x = A^{-1}b$. 否则, 协议中断, 方程组 $Ax = b$ 无解.

上述方案的安全性和效率可以由协议 MI 得到. 而且, 用户的计算复杂度仅仅是 $O(n^2)$.

6.2 基于纠错码的公钥密码体制

这一小节介绍了矩阵计算的可验证安全外包协议在基于纠错码的公钥密码体制中的一个应用实例.

McEliece 在文献 [20] 中利用一般线性码的译码问题是一个 NPC 问题和 Goppa 码有快速译码算法的特点提出了一个基于纠错码的公钥密码体制.

设 G 是二元 $[n,k,d]$Goppa 码的生成矩阵, 其中 $n=2^m, d=t+1, k=n-mt$. 明文集合是 $GF(2)^k$, 密文集合是 $GF(2)^n$. S,G,P 是私钥, $G'=SGP$ 是公钥.

基于 McEliece 公钥密码体制的安全外包协议按照以下过程执行:

密钥生成阶段:

Step1: 随机选取 $GF(2)$ 上的一个 $k\times k$ 阶可逆矩阵 S 和一个 $n\times n$ 阶置换矩阵 P.

Step2: 安全外包计算 $T=SG$. 这里调用可验证的安全外包协议 MP 来计算 SG, 因此没有泄露 S,G 和 SG.

Step3: 用户计算公钥 $G'=TP$.

加密阶段: 对任意一个明文 $m\in GF(2)^k$.

Step1: 选择 $GF(2)^n$ 上重量为 t 的随机向量 z.

Step2: 计算密文 $c=mG'+z$.

解密阶段: 对密文 c.

Step1: 计算 cP^{-1}.

Step2: 利用 Goppa 的快速译码算法将 cP^{-1} 译码成 $m'=mS$.

Step3: 调用可验证协议 MI 安全外包计算 S^{-1}.

Step4: 计算明文 $m=m'S^{-1}$.

上述外包协议的安全性可由可验证安全外包协议 MP 和 MI 得到.

效率分析: 在没有外包的情况下, 用户端执行 McEliece 公钥密码体制的计算负载主要是计算 G' 和 S^{-1}, 计算复杂度分别为 $O(nk^2+kn)$ 和 $O(k^3)$. 采用上述外包方案可以减少用户端的计算量, 计算 G' 和 S^{-1} 的复杂度相应的降到 $O(k^2+kn)$ 和 $O(k^2)$.

7 结论

本文提出的所有的可验证安全外包协议都是可实现的, 且给出了它们在基于纠错码的公钥密码体制中的一个应用实例, 实际上, 这些安全外包协议适用于许多基于编码的密码体制. 另外, 我们的协议的设计思想还可用于为矩阵特征多项式问题构造可验证的安全外包计算协议.

我们设计的所有的可验证安全外包协议是能够满足实际的密码协议的需求的, 本文也从理论角度分别对各协议的安全性和高效性进行了证明和分析. 而且, 所有的方法都没有泄露原问题中的秘密信息. 本文中提出的关于矩阵计算的 3 个可验证的安全外包协议, 是可以抵抗密码分析的许多方法对秘密数据的攻击.

参考文献

1 Sun Microsystems, Inc. Building customer trust in cloud computing with transparent security. 2009. https://www.sun.com/offers/details/sun_transparency.xml

2 Gentry C. Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing. Maryland, 2009. 169–178
3 Gentry C. Toward basing fully homomorphic encryption on worst-case hardness. In: Proceedings of the 30th Annual Cryptology Conference. Santa Barbara, 2010. 116–137
4 van Dijk M, Gentry C, Halevi S, et al. Fully homomorphic encryption over integers, In: Proceedings of the 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques. Riviera, 2010. 24–43
5 Smart N P, Vercauteren F. Fully homomorphic encryption with relatively small key and ciphertext sizes. In: Proceedings of the 13th International Conference on Practice and Theory in Public Key Cryptography. Paris, 2010. 420–443
6 Stehle D, Steinfeld R. Faster fully homomorphic encryption. In: Proceedings of the 16th International Conference on the Theory and Application of Cryptology and Information Security. Singapore, 2010. 377–394
7 Lyubashevsky V, Peikert C, Regev O. On ideal lattices and learning with errors over rings. In: Proceedings of the 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques. Riviera, 2010. 1–23
8 Brakerski Z, Vaikuntanathan V. Efficient fully homomorphic encryption (standard) LWE. In: IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS). Palm Springs, 2011. 97–106
9 Brakerski Z, Vaikuntanathan V. Efficient fully homomorphic encryption from ring-LWE and security for key dependent messages. In: Proceedings of the 31st Annual Cryptology Conference. Santa Barbara, 2011. 501–521
10 Benjamin D, Atallah M J. Private and cheating-free outsourcing of algebraic computations. In: Proceedings of the 6th Conference on Privacy, Security, and Trust (PST). New Brunswick, 2008. 240–245
11 Atallah M J, Frikken K. Securely outsourcing linear algebra computations. In: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security. Beijing, 2010. 48–59
12 Gennaro R, Gentry C, Parno B. Non-interactive verifiable computing: outsourcing computation to untrusted workers. In: Proceedings of the 30th Annual Cryptology Conference. Santa Barbara, 2010. 465–482
13 Wang C, Ren K, Wang J, et al. Harnessing the cloud for securely solving large-scale systems of linear equations. In: Proceedings of the 31st International Conference on Distributed Computing Systems (ICDCS). Minneapolis, 2011. 549–558
14 Mohassel P. Efficient and secure delegation of linear algebra. Cryptology ePrint Archive, Report 2011/605, 2011
15 Fiore D, Gennaro R. Publicly verifiable delegation of large polynomials and matrix computations, with applications. In: ACM Conference on Computer and Communications Security. Chicago, 2012. 501–512
16 Atallah M J, Pantazopoulos K N, Rice J R, et al. Secure outsourcing of scientific computations. Adv Comput, 2001, 54: 216–272
17 Hohenberger S, Lysyanskaya A. How to securely outsource cryptographic computations. In: Proceedings of the 2nd Theory of Cryptography Conference. Cambridge, 2005. 264–282.
18 Dwaine C, Srinivas D, van Dijk Marten, et al. Speeding up Exponentiation using an Untrusted Computational Resource. Technical Report Memo 469, MIT CSAIL Computation Structures Group, August 2003
19 Seitkulov Y N. New Methods of Secure Outsourcing of Scientific Computations. Springer Science+Business Media, LLC, 2012. 1–14
20 McEliece R J. A public-key cryptosystem based on algebraic coding theory. Jet Propulsion Laboratory DSN Progress Report 42-44, 114-116, 1978

Verifiable and secure outsourcing of matrix calculation and its application

HU Xing[1,2,3], PEI DingYi[1,2], TANG ChunMing[1,2*] & Duncan S. WONG[4]

1 *School of Mathematics and Information Science, Guangzhou University, Guangzhou* 510006*, China;*
2 *Key Laboratory of Mathematics and Interdisciplinary Sciences of Guangdong Higher Education Institutes,*

Guangzhou University, Guangzhou 510006*, China;*
3 *School of Mathematics and Computational Science, Hunan University of Science and Technology, Xiangtan* 411102*, China;*
4 *Department of computer science, City University of Hongkong, Hong Kong, China*
*E-mail: ctang@gzhu.edu.cn

Abstract Matrix calculation plays an important role in both scientific computation and cryptography. Many cryptographic protocols, scientific computations, and numerical computations are based on matrix calculation. However, it is difficult to finish matrix calculation independently for these customers whose computation abilities are limited. Cloud Computing has a great deal of computational resources, which enable customers with limited computational resources to outsource their mass computing to the cloud. In this paper we design secure, verifiable, and practical outsourcing protocols for matrix calculation, including matrix multiplication, computing the determinant and inverse of a matrix. Compared with those existing outsourcing protocols, our protocols have obvious improvement concerning both efficiency and security. Furthermore, no cryptographic assumption is needed in our protocols. Finally, we give two applications for our outsourcing protocols; one is to construct an outsourcing protocol for solving the large-scale systems of linear equations, and the other is to design an outsourcing protocol for realizing a cryptosystem based on error-correction code.

Keywords cloud computing, outsourcing computation, matrix multiplication, determinant of a matrix, matrix inversion

HU Xing was born in 1981. She received the Master's degree in School of Mathematics and Computational Science from Hunan University of Science and Technology, Xiangtan, in 2009. Now she is a doctor candidate of the Guangzhou University. Her research interests include cryptography and cloud computing. HU Xing is a member of Chinese Association for Cryptologic Research.

TANG ChunMing was born in 1972. He received the Ph. D. degree in Academy of Mathematics and Systems Science from Chinese Academy of Sciences, Beijing, in 2004. Currently, he is a Professor at Guangzhou University. His research interests include Cryptography, Information Security, and Cloud Computing. Dr. TANG is a member of Chinese Association for Cryptologic Research.

PEI DingYi was born in 1941. He received the Master's degree in department of Mathematics from University of Science and Technology of China, Hefei, in 1968. Currently, he is a Professor at Guangzhou University. His research interests include Cryptography and Information Security. Dr. PEI is a member of Chinese Association for Cryptologic Research, as well as Chair.

Duncan S. WONG was born in 1970. He received the Ph.D. degree in Computer Science from Northeastern University, Boston, MA, USA in 2002. Currently, he is an Associate Researcher at City University of Hong Kong. His research interests include applied cryptography, in particular, cryptographic protocols, encryption and signature schemes, and anonymous systems.