doi:10.3969/j.issn.1000-436x.2014.z2.021

高效的基于混合加密的乐观 Mix-net 协议

李龙海，黄诚强，许尚妹，付少锋
（西安电子科技大学 计算机学院，陕西 西安 710071）

摘 要：提出了一种高效的基于混合加密算法的 Mix-net 协议。正常情况下，当所有 Mix 服务器都忠实地执行协议时，其运算速度优于其他所有已知的具有公开可验证性的 Mix-net。采用一种“乐观的”、基于散列运算的方法验证混洗过程的正确性，避免了构造复杂的、耗时的零知识证明，因此获得了速度上的提升。只用两轮逐元素的测试过程确保消息未被恶意服务器篡改，并且测试中仅涉及低代价的散列运算。公钥加密和对称密钥加密的有效结合也加速了混洗。这些优化措施使单个 Mix 服务器的运算量几乎和服务器数目无关，除了少量可忽略的计算任务。此外，任何人都可以通过少量的指数运算快速验证输出结果的正确性。方案也满足健壮性。这些特点使该方案非常适合用在大规模的电子选举中。

关键词：匿名通信；电子选举；混合网络；混合加密
中图分类号：TP393.08 **文献标识码**：A **文章编号**：1000-436X(2014)Z2-0154-11

Efficient hybrid-encryption-based optimistic Mix-net protocol

LI Long-hai, HUANG Cheng-qiang, XU Shang-mei, FU Shao-feng
(School of Computer Science and Technology, Xidian University, Xi’an 710071, China)

Abstract: An efficient hybrid-encryption-based Mix-net is presented that is much faster than all previous Mix-nets with public verifiability when all mix-servers execute the mixing protocol honestly (the usual case). The improvement by taking an “optimistic” and hash-based approach to verify the correctness of mixing is achieved without requiring complex and costly zero-knowledge proofs. Only two element-wise testing processes with low-cost computations of hash functions are involved to make certain messages are not manipulated by a cheating server. An efficient integration of public-key and symmetric-key operations also speeds up the mixing. As a result, the computational task of each mix-server is almost independent of the number of mix-servers except for some negligible tasks. Anyone can verify the correctness of a result rapidly by computing a few exponentiations. The scheme is robust, too. Those characteristics make it very suitable for large scale electronic voting.

Key words: anonymous communication; electronic voting; mix network; hybrid encryption

1 引言

Mix-net 是 Chaum 最早提出的用于在通信过程中实现隐私保护的密码学协议[1]。Mix-net 协议是多个用户和 Mix 服务器之间的一种多方计算协议。将一组密文(c_1, c_2,…, c_N)输入到 Mix 服务器之后，该服务器将输出解密后所得明文组(m_1, m_2, …, m_N)的一个随机置换。如果该置换保密并且加密算法是安全的，则攻击者无法确定输入与输出之间的对应关系，也就无法追踪任意消息 m_i 的发送者。Mix-net 协议在很多与隐私保护相关的应用系统中扮演了重要角色，其中包括匿名电子邮件[1]、匿名实时通信[2]、电子选举[3,4]、匿名 Web 浏览[5]、电子支付[6]等。

为了避免将信任完全建立在单个服务器上，多个 Mix 服务器被串联在一起并依次对输入密文组进行部分解密和混洗，由最后一级服务器输出明文。

收稿日期：2014-07-04
基金项目：国家自然科学基金资助项目(61101142)；中央高校基本科研基金资助项目(K50510030012)
Foundation Items: The National Natural Science Foundation of China (61101142); The Fundamental Research Funds for the Central Universities(K50510030012)

只要有一个 Mix 服务器是诚实的，就能够保证输入和输出之间的不可关联性(unlinkability)。如何在保证匿名性的前提下防止恶意 Mix 服务器在混洗过程中替换或篡改部分消息是设计 Mix-net 协议的难点。针对该问题，目前最常用的方法是利用零知识证明技术：要求每个 Mix 服务器输出混洗结果的同时，公布一个输入、输出集合元素存在一一对应关系的非交互式零知识证明。基于零知识证明的 Mix-net 协议的主要问题是计算复杂度太高，每个服务器构造零知识证明都需要进行 $O(N)$次指数运算，而验证者需要进行 $O(nN)$次指数运算才能确定整个 Mix-net 系统的正确性，其中 N 表示输入元素个数，n 表示 Mix 服务器数目。在 N 较大时(例如，在大规模电子选举中)，该运算是比较耗时的。根据文献[7]中的实验，当输入 4 000 个 ElGamal 密文时，每一级服务器构造 Neff 提出的 Mix 零知识证明[4]需要花超过 1 个小时。目前已知效率最高的 Mix 零知识证明[8]与 Neff 的证明相比，在运算量上仅降低了一倍，仍然为 $O(N)$级。

为了进一步提高效率，提出了一种高效的基于混合加密算法的 Mix-net 协议。正常情况下，当所有 Mix 服务器都忠实地执行协议时，其运算速度优于其他所有已知的具有公开可验证性的 Mix-net。受 Golle 方案的启发[9]，采用了一种“乐观的”、基于散列运算的方法验证混洗过程的正确性，避免了构造复杂的、耗时的零知识证明，因此获得了速度上的提升。只用两轮逐元素的测试过程确保消息未被恶意服务器篡改，并且测试中仅涉及低代价的散列运算。公钥加密和对称密钥加密的有效结合也加速了混洗。这些优化措施使单个 Mix 服务器的运算量几乎和服务器数目无关，除了少量可忽略的计算任务。此外，任何人都可以通过少量的指数运算快速验证输出结果的正确性。方案也满足健壮性。这些特点使该方案非常适合用在大规模的电子选举中。

2　相关工作

针对传统零知识证明 Mix-net 协议计算复杂度高的问题，Golle 等提出了一种基于乐观验证策略的 Mix-net[9]。该协议在多服务器混洗过程中只作非常简单的逐级验证，最后等混洗结束之后通过散列值或消息认证码(MAC)直接验证结果是否正确；在简单验证失败后再启动速度较慢的基于零知识证明的 Mix-net 协议。因此，在没有作弊行为的正常情况下每个服务器花在完整性证明和验证方面的计算量接近于常数级；在有服务器作弊的异常情况下，其复杂度则略高于传统的基于零知识证明的 Mix-net 协议。因此，在服务器出错概率较低的应用中，采用这种乐观验证方法是非常高效的。不幸的是，Golle 的协议被发现存在严重的安全漏洞，在文献[10~12]中先后提出了多种攻击方法破坏其匿名性和完整性。为了避免选择密文攻击[11,12]，Golle 的协议必须每执行一次都要重新生成和发布密钥。文献[10]给出了一种有效的消息替换攻击，但没有给出相应的修正方法。

文献[7]提出的专用于电子选举的 Mix-net 利用了与文献[9]类似的双层加密、和基于散列的完整性保证方法，并且外层加密采用了 ElGamal 算法，内层采用基于椭圆曲线的混合加密算法。在没有参与者作弊的情况下，其效率非常高。但该协议非常脆弱：由于没有“备份混洗”措施，一旦混洗完毕后基于散列值的完整性验证失败，所有混洗结果（即加密的选票）将不得不被丢弃。虽然此时投票者的匿名性不会受到影响，但电子选举过程被终止。因此，单个投票者故意制造无效选票就可以终止或重启整个选举活动，这在大规模选举中是不现实的。

实现乐观验证的核心技术是将消息进行双层加密，然后针对外层加密进行乐观型混洗，一旦出现错误再针对内层加密进行运算速度较慢的非乐观型的“备份混洗”(backup mixing)。Golle 的 Mix-net[9]在内外两层加密时都采用了 ElGamal 算法，这种方式不但增加了服务器端的混洗和解密负担，而且使密文长度成倍增长。为了解决 Golle 的 Mix-net[9]中存在的安全性问题并进一步提高效率，设计了一种新的基于混合加密的乐观 Mix-net 协议。该协议具有如下特点。

1) 在对用户消息进行外层加密时采用了对称密钥加密算法，而用户和服务器之间的共享对称密钥利用公钥密钥体制建立。在混洗时服务器对输入密文进行逐层解密，对称解密算法的高效性大大提高了混洗速度，尤其是在处理长消息时速度优势更加明显。

2) 用户输入密文长度固定，与 Mix 服务器数目无关。

3) 在外层加密时封装了关于内层解密参数的散列值，大大降低了内层解密时各个 Mix 服务器构

造解密有效性证明和验证证明的工作量。

4) 在没有服务器作弊的情况下，此方案几乎完全实现了单个Mix服务器运算量与服务器数目无关的特性(只有很少的运算量和服务器数目相关)，即计算复杂度为 $O(N)$级，而其他所有具有公开可验证性的 Mix-net 方案的计算复杂度都为 $O(nN)$级。

5) 任何第三方都可以验证混洗、解密结果的正确性，并且在没有服务器作弊的情况下，验证者主要做简单的散列运算，其运算量几乎和 Mix 服务器数目无关。而目前只有极少数的 Mix-net 能够实现该特性。

所提出的乐观 Mix-net 协议非常适合用在服务器作弊代价较高、而普通用户作弊代价较低的应用中。

3 模型与安全性定义

3.1 协议模型

Mix-net 协议参与者包括 N 个用户$\{P_1, \cdots, P_N\}$和 n 个 Mix 服务器$\{S_1, \cdots, S_n\}$，它们都被视为具有多项式计算能力的交互式图灵机。

如图 1 所示，Mix-net 的功能可以概括为：用户 P_i将消息 m_i加密后得到密文 C_i，然后将 C_i发送给 Mix 服务器组。Mix 服务器组收到所有用户的密文$(C_1, C_2, \cdots, C_N)$后，经过随机排序和联合解密输出另一组明文$\{m'_1,\cdots,m'_N\}$。如果输出$\{m'_1,\cdots,m'_N\}$是输入$\{m_1,\cdots,m_N\}$的一个随机置换，则称该 Mix-net 协议是正确的。

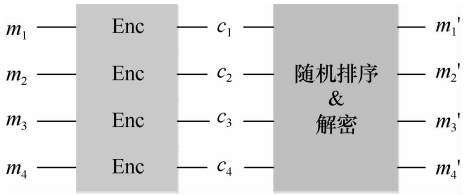


图 1 Mix-net 模型

通信模型定义 假设存在具有认证功能的可靠广播系统(BBS 系统)。在协议运行的任意时刻，任意参与者 P_i都能可靠地向 BBS 上写入消息 msg，任何人都能够从 BBS 上读到消息$(count, P_i, msg)$，其中 $count$ 是不断自增的计数值。消息一旦写入 BBS 就不能修改，用户只能向 BBS 追加消息。几乎所有的 Mix-net 协议都需要建立在上述可靠 BBS 假设之上。

下面给出 Mix-net 的严格定义。

定义 1(Mix-net) Mix-net 是一个三元组$(\mathcal{G}^{\mathrm{MIX}}, \mathcal{E}^{\mathrm{MIX}}, m)$，其中：

$\mathcal{G}^{\mathrm{MIX}}$ 是一个概率性密钥生成算法，其输入包括一个安全参数 K 和一个秘密共享控制结构 Γ，输出包括一个公开钥集合 Y 和一个秘密钥集合 X。X 中包括 n 个子密钥，分别被 n 个 Mix 服务器持有。符合 Γ 定义的若干个 Mix 服务器可以联合恢复其他 Mix 服务器的子密钥。

$\mathcal{E}^{\mathrm{MIX}}$ 是一个概率性加密算法，其输入为公钥集合 Y 和用户的消息明文 m，输出为密文 C。

M 是一个由 n 个 Mix 服务器参与的安全多方计算协议，不同的服务器之间利用 BBS 相互通信，M 的输出也公布在 BBS 上。M 的公共输入包括 BBS 上用户的密文列表$(C_1, C_2, \cdots, C_N)$和公钥集合 Y，各个 Mix 服务器的秘密输入为集合 X 中对应的子密钥。M 的输出为一组明文$\{m'_1,\cdots,m'_N\}$。

定义 2(完备性) 设诚实用户的消息输入为 $L=\{m_1, \cdots ,m_h\}$，Mix-net 运行结束后的输出为 $L'=\{m'_1,\cdots,m'_N\}$。如果满足 $L\subseteq L'$，则称该 Mix-net 是完备的，或称输出 L' 是正确的。

3.2 安全性定义

定义 3(攻击模型定义) 对攻击者 A 的能力定义如下：A 只具有多项式计算能力。A 最多能够控制 $N-1$ 个用户和 $t-1$ 个 Mix 服务器$(2t-1\leqslant n)$，并且要求 A 能够控制的恶意用户集合和恶意服务器集合在协议运行期间不能改变。A 能够获得被控制用户和 Mix 服务器的所有秘密输入，A 能够通过 BBS 获得任意参与者的公开输出。

针对以上定义的攻击者 A，设计目标是是使 Mix-net 协议在 A 的攻击下满足如下特性。

定义 4(可验证性) 针对定义 3 规定的攻击者 A，如果 Mix-net 在运行结束后产生不正确输出而不被诚实参与者发现的概率是 K 的可忽略函数，则称 Mix-net 满足可验证性。

定义 5(健壮性) 针对定义 3 规定的攻击者 A，如果 Mix-net 总是能在多项式时间内以压倒性概率运行结束并产生正确输出，则称 Mix-net 满足健壮性。

对任意正整数 h，设 Σ_h表示$\{1, 2, \cdots, h\}$上所有置换构成的集合。

定义 6(匿名性) 定义攻击者 A 和 Mix-net 之间的一个游戏 Game ANON(Mix-net, A)如下。

1) 运行 $\mathcal{G}^{\mathrm{MIX}}$ 生成公开钥集合 Y 和秘密钥集合 X，并将 Y 公布到 BBS 上。

2) A 根据 BBS 上公布的内容生成 2 个明文列表 $L_0=\{m_1,\cdots,m_h\}$ 和 $L_1=\{m_{\pi(1)},\cdots,m_{\pi(h)}\}$，其中 $\pi\in\Sigma_h$。

3) 任取 $b\in\{0,1\}$，然后以 L_b 作为 h 个诚实用户的输入(恶意用户输入由 A 任选)运行 $\mathcal{E}^{\text{MIX}}$ 和 M。A 可以通过 BBS 读取 Mix-net 运行期间所有公开的中间结果和最后输出。

4) 最后 A 输出 $b'\in\{0,1\}$。

A 赢得上述游戏的优势被定义为

$$Adv^{\text{ANON}}_{\text{Mix-net}}(A)=|\Pr(b'=b)-1/2|$$

如果对于定义 3 规定的攻击者 A，$Adv^{\text{ANON}}_{\text{Mix-net}}(A)$ 是关于 K 的可忽略函数，则称该 Mix-net 协议满足匿名性。

4　协议详细设计

4.1　符号与假设

设 G 表示一个有限循环群，g 为其生成元，$q=|G|$为足够大的素数，使在群 G 中 DDH (decisional diffie hellman problem)问题难解。

$(\mathcal{E},\mathcal{D},\mathcal{K},\mathcal{M},\mathcal{C})$ 表示一个对称加密方案，其中 $\mathcal{E}$、$\mathcal{D}$ 分别表示加密和解密算法，$\mathcal{K}$、$\mathcal{M}$、$\mathcal{C}$ 分别表示密钥空间、消息空间和密文空间。$\mathcal{E}_K(x)$ 表示用密钥 K 对消息 x 的加密结果，$\mathcal{D}_K(x)$ 表示用密钥 K 对密文 x 的解密结果。该对称加密方案具有长度保持特性，即 $\mathcal{M}=\mathcal{C}=\{0,1\}^{l_{\text{msg}}}$。为了在 Mix-net 方案中实现输入与输出的不可关联性，必须要求该对称加密方案满足如下定义的不可区分性(indistinguishability)。

定义 7　设 k_0、k_1 是从 $\mathcal{K}$ 中任取的 2 个密钥，A 表示任意的具有多项式计算能力的攻击者。进行如下实验：首先，由 A 任取 2 个等长的消息 $m_0,m_1\in\{0,1\}^*$，然后任取 $b\in\{0,1\}$ 并计算 $c_0=\mathcal{E}_{k_0}(m_b)$，$c_1=\mathcal{E}_{k_1}(m_{1-b})$，最后将 c_0、c_1 交给 A 并由 A 猜测 b 的值。如果 A 猜中 b 的概率与 1/2 相比只具有可忽略的优势，则称该对称加密方案具有不可区分性。

满足上述特性的对称加密算法完全可以用具有不可区分性的伪随机数生成器(pseudo random number generator)构造。

另外，还需要假设 3 个理想的散列函数：$H_1:G\to\{0,1\}^{len}$，$H_2:\{0,1\}^*\to G$，$H_3:G\to\mathcal{K}$，其中 len 表示某个安全长度，例如令 len 等于 128 或 160。在安全性分析中它们都将被视为 random oracle[13]。

$\text{NIZK}\{x_1,x_2,\cdots,x_e:A(x_1,x_2,\cdots,x_e)\}$ 表示关于秘密值 $x_1,x_2,\cdots,x_e$ 的非交互式零知识证明(non-interactive zero knowledge proof)，A 是关于离散对数的某个逻辑命题。示证者在不暴露 $x_1,x_2,\cdots,x_e$ 的条件下证明命题 A 为真。其具体构造方法可以参考文献[14,15]。

4.2　密钥生成算法 $\mathcal{G}^{\text{MIX}}$

由 n 个 Mix 服务器首先根据安全参数 K 共同选定有限循环群 G、生成元 g、对称加密方案 $(\mathcal{E},\mathcal{D},\mathcal{K},\mathcal{M},\mathcal{C})$、散列函数 H_1, H_2, H_3。秘密共享控制结构 Γ 定义为 n 个 Mix 服务器中任取 t 个服务器。然后按照如下过程生成服务器端的公开钥和秘密钥。

1) 为 ElGamal 系统建立密钥：每个服务器 S_i $(i=1,2,\cdots,n)$任取 $x_i\in_R Z_q^*$ 作为自己的子密钥，并公布 $y_i=g^{x_i}$ 和非交互式证明 $\text{NIZK}\{x_i:y_i=g^{x_i}\}$；$S_i$ 同时利用可验证的(t,n)门限秘密共享协议[16]将 x_i 共享给其他服务器；等所有服务器公布完毕之后，服务器端计算 $y=\prod_{i=1}^{n}y_i$。

2) 为混合加密系统建立密钥：设 $g_0=g$，对 $i=1,2,\cdots,n$，服务器 S_i 依次计算并公布 $g_i=g_{i-1}^{\alpha_i}$，$z_i=g_i^{\beta_i}$，其中 α_i 和 β_i 是 S_i 从 Z_q^* 中任取的子密钥，S_i 将它们秘密保存。S_i 同时需要利用可验证的(t,n)门限秘密共享协议[16]将 α_i 和 β_i 共享给其他服务器。

最后在 BBS 上公布的公钥集合 $Y=\{G,g,(\mathcal{E},\mathcal{D},\mathcal{K},\mathcal{M},\mathcal{C}),H_1,H_2,H_3,y,(g_1,z_1),(g_2,z_2),\cdots,(g_n,z_n)\}$。每个 Mix 服务器 S_i 的子密钥为 $\{x_i,\alpha_i,\beta_i\}$。

另外，为了避免基于多轮 Mix-net 运行的选择密文(CCA)攻击[11]，假定每开始一个新的 Mix 会话都要重新运行 $\mathcal{G}^{\text{MIX}}$ 生成相关密钥。这在电子选举应用中是一个合理的假设。

4.3　用户消息加密算法 $\mathcal{E}^{\text{MIX}}$

设 $E_y(x)$表示用公钥 y 对 x 的 ElGamal 加密。为发送消息 $m\in G$，发送者 P_i 执行如下定义的加密算法 $\mathcal{E}^{\text{MIX}}$ 生成密文 C。

任取 $r_1,r_2,s\in_R Z_q^*$，并计算

$$(u,v)=(g^s,y^sm),\quad h_1=H_1(y^s),\quad h_2=H_2(u\|v\|h_1),$$

$$K_i=H_3(z_i^{r_1})\ (i=1,2,\cdots,n),$$

$$a=g^{r_1},\quad b=\mathcal{E}_{K_1}\mathcal{E}_{K_2}\cdots\mathcal{E}_{K_n}(u\|v\|h_1),$$

$$(c,d)=E_y(h_2)=(g^{r_2},y^{r_2}h_2)$$

为防御恶意用户的选择密文攻击(CCA)，要求 P_i 必须构造非交互式证明

$$\mathrm{NIZK}\{r_1, r_2 : a = g^{r_1} \wedge c = g^{r_2}\}$$

如果 P_i 提交密文 (a,b,c,d) 时能同时提供有效证明，则表明 P_i 一定知道密文 b 中包含的明文 $u \| v \| h_1$ 以及密文 d 中包含的明文 h_2。即上述混合加密算法满足明文可意识性(plaintext awareness)。

最后，P_i 将密文 C=(a,b,c,d) 和上面的零知识证明公布在 BBS 上作为自己的加密输出。

4.4 混洗协议 *M*

等所有用户的密文公布完毕或到达某个时间限之后，Mix 服务器组共同执行 M 协议对输入密文组进行混洗和解密。具体过程定义如下。

1) 第一轮部分解密和混洗

Mix 服务器组首先对 BBS 中的每个用户输入所附带的非交互式证明进行验证，并去掉无效的输入。设经过验证之后获得的有效 4 元组向量为 $\boldsymbol{L}_0 = \{(a_i, b_i, c_i, d_i)\}_{i=1}^N$。$n$ 个 Mix 服务器依次按照如下步骤对 $\boldsymbol{L}_0$ 进行处理。

① 每个服务器 S_j 首先从 BBS 上读取前一个服务器的输出

$$\boldsymbol{L}_{j-1} = \{(a_{j-1,i}, b_{j-1,i}, c_{j-1,i}, d_{j-1,i})\}_{i=1}^N$$

② S_j 对 $\boldsymbol{L}_{j-1}$ 中每个四元组的 $(a_{j-1,i}, b_{j-1,i})$ 分量进行部分解密。对 i = 1, 2, …, N，S_j 计算

$$\hat{a}_{j,i} = a_{j-1,i}^{\alpha_j},\ K_{j,i} = H_3(\hat{a}_{j,i}^{\beta_j}),\ \hat{b}_{j,i} = \mathcal{D}_{K_{j,i}}(b_{j-1,i})$$

③ S_j 对 $\boldsymbol{L}_{j-1}$ 中每个四元组的 $(c_{j-1,i}, d_{j-1,i})$ 分量进行 ElGamal 再加密。对 i = 1, 2, …, N，S_j 任取 $r_{ji} \in_R Z_q^*$ 然后计算

$$\hat{c}_{j,i} = g^{r_{j,i}} c_{j-1,i},\ \hat{d}_{j,i} = y^{r_{j,i}} d_{j-1,i}$$

④ S_j 任取置换 $\pi_j : \{1,\cdots,N\} \to \{1,\cdots,N\}$，并按照 π_j 对 $\{(\hat{a}_{j,i}, \hat{b}_{j,i}, \hat{c}_{j,i}, \hat{d}_{j,i})\}_{i=1}^N$ 进行重排序，得到输出向量：$\boldsymbol{L}_j = \{(a_{j,i}, b_{j,i}, c_{j,i}, d_{j,i})\}_{i=1}^N$。

⑤ 设 $c_j = \prod_{i=1}^N c_{j,i}$，$d_j = \prod_{i=1}^N d_{j,i}$，$c_{j-1} = \prod_{i=1}^N c_{j-1,i}$，$d_{j-1} = \prod_{i=1}^N d_{j-1,i}$。$S_j$ 构造非交互式知识证明

$$\sigma_j = \mathrm{NIZK}\{r_j : c_j c_{j-1}^{-1} = g^{r_j} \wedge d_j d_{j-1}^{-1} = y^{r_j}\}$$

S_j 在构造 σ_j 时的秘密输入为：$r_j = \sum_{i=1}^N r_{j,i}$。

⑥ S_j 将 $\boldsymbol{L}_j$ 公布在 BBS 上作为下一个服务器的输入。S_j 同时公布 σ_j 让其他服务器验证。如果 σ_j 被验证无效，则确定 S_j 为作弊者，将 S_j 从系统中删除。其他服务器利用自己掌握的关于 α_i 和 β_i 的子秘密代替 S_j 执行上述操作。

上述过程进行完毕之后，最后一个服务器 S_n 的输出 $\boldsymbol{L}_n$ 作为本阶段的最终输出。

2) 第一次 ElGamal 解密

设 $\boldsymbol{L}_n = \{(\overline{a}_i, \overline{b}_i, \overline{c}_i, \overline{d}_i)\}_{i=1}^N$。在本阶段，$n$ 个服务器利用自己的密钥 x_j（j = 1, 2, …, n）对 $\boldsymbol{L}_n$ 中每个元素的 $(\overline{c}_i, \overline{d}_i)$ 分量进行联合 ElGamal 解密。实际上也可以将该解密操作集成到主协议第 2 步的混洗阶段，这样可以节省一轮交互过程。为了使协议过程的表述更加清晰，分成了 2 个独立的阶段进行描述。

① 设 $\overline{c} = \prod_{i=1}^N \overline{c}_i$，$\overline{d} = \prod_{i=1}^N \overline{d}_i$。任意服务器 S_j ($1 \leqslant j \leqslant n$)计算并公布

$$\{D_{j,i}\}_{i=1}^N = \{\overline{c}_i^{-x_j}\}_{i=1}^N,\ D_j = \prod_{i=1}^N D_{j,i},$$

$$\delta_j = \mathrm{NIZK}\{x_j : D_j = (\overline{c})^{-x_j} \wedge y_j = g^{x_j}\}$$

② 等所有服务器公布完毕之后，任意服务器计算

$$\{\overline{h}_{2,i}\}_{i=1}^N = \{\overline{d}_i \prod_{j=1}^n D_{j,i}\}_{i=1}^N,\ \overline{h}_2 = \overline{d} \prod_{j=1}^n D_j$$

然后验证 $\overline{h}_2 = \prod_{i=1}^N \overline{h}_{2,i}$ 是否成立。如果成立，则以 $\boldsymbol{L}_D = \{(\overline{a}_i, \overline{b}_i, \overline{h}_{2,i})\}_{i=1}^N$ 为解密输出进入下一阶段。如果不成立则要求每个服务器 S_j 都公布证明

$$\mathrm{NIZK}\{x_j : y_j = g^{x_j} \wedge D_{j,1} = \overline{c}_1^{-x_j} \wedge \cdots \wedge D_{j,N} = \overline{c}_N^{-x_j}\}$$

如果某个服务器 S_k 无法提供上述有效证明，则证明该服务器作弊，并由其他服务器利用自己掌握的关于 x_k 的子秘密代替 S_k 执行解密操作。

3) 验证混洗结果的有效性

该阶段包括如下 3 个步骤。

① 设 G 的每个元素用 l_G bit 表示。首先将 $\boldsymbol{L}_D$ 中的每个三元组 $(\overline{a}_i, \overline{b}_i, \overline{h}_{2,i})$ 中的 $\overline{b}_i$ 解释为 $\overline{u}_i \| \overline{v}_i \| \overline{h}_{1,i}$ 的形式，即令 $\overline{b}_i$ 的最高的 l_G bit 表示 $\overline{u}_i$，中间的 l_G bit 表示 $\overline{v}_i$，剩余的 bit 表示 $\overline{h}_{1,i}$。

② 验证对任意的 i，$\overline{h}_{2,i} = H_2(\overline{b}_i) = H_2(\overline{u}_i \| \overline{v}_i \| \overline{h}_{1,i})$ 是否成立。如果对某个 i 该式不成立，则认为三元组 $(\overline{u}_i, \overline{v}_i, \overline{h}_{1,i})$ 是无效的。这种无效的密文可能是由恶意用户造成的，也可能是由恶意服务器造成的。这时需要以 $(\overline{a}_i, \overline{b}_i, \overline{c}_i, \overline{d}_i)$ 为输入参数执行第 5 步“反向追踪子协议”来确定无效密文的来源。

根据“反向追踪子协议”的执行结果又分为 2 种情况。

a) 如果反向追踪过程表明 $(\overline{u}_i,\overline{v}_i,\overline{h}_{1,i})$ 是良性的，即所有服务器对该输入项都执行了正确操作，说明该三元组的无效是发送用户本身造成的，则忽略该三元组，并继续下一个 3 元组的验证。

b) 如果反向追踪过程表明 $(\overline{u}_i,\overline{v}_i,\overline{h}_{1,i})$ 是恶性的，即有一个或多个服务器对该三元组执行了错误操作，则首先将这些恶意服务器删除，然后由剩余的服务器相互合作，利用他们手中的子秘密共同恢复这些恶意服务器在混洗阶段和第一次 ElGamal 解密阶段的错误。设 $\tilde{\boldsymbol{L}}=\{(\tilde{u}_i,\tilde{v}_i)\}_{i=1}^{N}$ 表示错误恢复之后得到的只有单层加密的密文向量。以 $\tilde{\boldsymbol{L}}$ 为输入执行基于零知识证明的 Mix-net 协议(如 Neff[4]的方案)获得明文输出 $\boldsymbol{L}_{\boldsymbol{M}}=\{m_1,\cdots,m_N\}$，即以 $\tilde{\boldsymbol{L}}$ 为输入进行“备份混洗”。

③ 如果第②步的验证表明 $\{(\overline{u}_i,\overline{v}_i,\overline{h}_{1,i})\}_{i=1}^{N}$ 中的三元组都是有效的或良性的，则将其中良性的三元组删除之后执行下一阶段的针对 $\{(\overline{u}_i,\overline{v}_i)\}_{i=1}^{N}$ 的内层 ElGamal 解密操作。为了方便说明，现假设本阶段最终输出 $\boldsymbol{L}_E=\{(\overline{u}_i,\overline{v}_i,\overline{h}_{1,i})\}_{i=1}^{N}$ 中的三元组都是有效的。

4) 第二次 ElGamal 解密

设第二次 ElGamal 解密阶段的输入为 $\boldsymbol{L}_E=\{(\overline{u}_i,\overline{v}_i,\overline{h}_{1,i})\}_{i=1}^{N}$。

① 任意服务器 S_j $(1\leqslant j\leqslant n)$公布：$\{F_{j,i}\}_{i=1}^{N}=\{(\overline{u}_i)^{-x_j}\}_{i=1}^{N}$。

② 等所有服务器公布完毕之后，验证对任意的 $i(1\leqslant i\leqslant N)$，$\overline{h}_{1,i}=H_1(\prod_{j=1}^{n}F_{j,i})$ 是否成立。

③ 如果对某个 i，第②步的验证不能通过，则要求每个服务器 S_j 都公布证明 $\mathrm{NIZK}\{x_j:F_{j,i}=(\overline{u}_i)^{-x_j}\wedge y_j=g^{x_j}\}$。如果某服务器的证明是无效的，则认为该服务器是作弊者，由其他服务器利用共享秘密恢复协议代替该恶意服务器生成解密参数。如果所有的服务器的证明都是有效的，则认为解密参数 $F_{1,i},F_{2,i},\cdots,F_{n,i}$ 都是正确的，而 $\overline{h}_{1,i}$ 是错误的(是由发送用户造成的错误)。针对这种有问题的三元组 $(\overline{u}_i,\overline{v}_i,\overline{h}_{1,i})$ 又有 2 种处理方式。一种是温和处理方式，即直接忽略该密文；另一种是严厉处理方式，即对 $(\overline{u}_i,\overline{v}_i,\overline{h}_{1,i})$ 进行反向追踪找到该消息的发送者，然后对该发送者采取某种惩罚措施。采用严厉处理方式可以避免恶意用户故意利用这类错误加重服务器端的负载，即制造拒绝服务攻击，但同时也破坏了用户的匿名性。所以应该根据实际的应用情况选择合适的处理方式。

④ 经过上面的验证和错误恢复过程之后，现在已经能够确认任意的解密参数 $F_{j,i}$ 都是正确的。服务器组计算最终的明文输出向量

$$\boldsymbol{L}_M=\{m_i\}_{i=1}^{N}=\{\overline{v}_i\prod_{j=1}^{n}F_{j,i}\}_{i=1}^{N}$$

5) 反向追踪子协议

设要追踪的四元组为 $(\overline{a}_i,\overline{b}_i,\overline{c}_i,\overline{d}_i)=(a_{n,i},b_{n,i},c_{n,i},d_{n,i})\in\boldsymbol{L}_n$。该协议必须在不泄漏服务器的秘密 α_i 和 β_i 的条件下实现对 $(\overline{a}_i,\overline{b}_i,\overline{c}_i,\overline{d}_i)$ 反向追踪，其具体过程如下。

设 $k_{n+1}=i$，对 $j=n,n\text{-}1,\cdots,1$，服务器 S_j 依次公布

$$k_j=\pi_j^{-1}(k_{j+1}),\quad R_j=r_{j,k_j},\quad \rho_j=(a_{j,k_{j+1}})^{\beta_j}$$

$$\mathrm{NIZK}\{\alpha_j:a_{j,k_{j+1}}=(a_{j-1,k_j})^{\alpha_j}\wedge g_j=g_{j-1}^{\alpha_j}\}$$

$$\mathrm{NIZK}\{\beta_j:\rho_j=(a_{j,k_{j+1}})^{\beta_j}\wedge z_j=g_j^{\beta_j}\}$$

其他服务器首先验证上面的 2 个非交互式证明是否有效，然后再验证下列各式是否成立

$$c_{j,k_{j+1}}=g^{R_j}c_{j-1,k_j},\quad d_{j,k_{j+1}}=y^{R_j}d_{j-1,k_j}$$

$$b_{j,k_{j+1}}=\mathcal{D}_{H_3(\rho_j)}(b_{j-1,k_j})$$

如果上面的验证有一项不能通过，则认为 S_j 是作弊者且 $(\overline{a}_i,\overline{b}_i,\overline{c}_i,\overline{d}_i)$ 是恶性的。如果所有的服务器都是诚实的，则认为该错误是用户造成的且 $(\overline{a}_i,\overline{b}_i,\overline{c}_i,\overline{d}_i)$ 是良性的。

5　分析

5.1　安全性分析

1) 完备性

定理 1　在第 4 节定义的 Mix-net 协议中，如果所有 Mix 服务器都诚实地执行了协议，则 Mix-net 能够产生正确输出，即所有诚实用户输入密文所对应的明文会全部出现在 Mix-net 的输出向量 $\boldsymbol{L}_M$ 中。

证明　可以将诚实用户 P_i 的输入表示为

$$(a_{0,i},b_{0,i},c_{0,i},d_{0,i})=(g^r,\mathcal{E}_{H_3(z_1^r)}\mathcal{E}_{H_3(z_2^r)}\cdots\mathcal{E}_{H_3(z_n^r)}(E_y(m_i)\|h_1),E_y(h_2))$$

服务器 S_1 在混洗阶段针对 $(a_{0,i},b_{0,i},c_{0,i},d_{0,i})$ 做了如下计算

$$\hat{a}_{1,i}=a_{0,i}^{\alpha_1}=g^{r\alpha_1}=g_1^r$$

$$K_{1,i}=H_3(\hat{a}_{1,i}^{\beta_1})=H_3(g_1^{r\beta_1})=H_3((g_1^{\beta_1})^r)=H_3(z_1^r)$$

$$\begin{aligned}\hat{b}_{1,i}&=\mathcal{D}_{K_{1,i}}(b_{0,i})\\&=\mathcal{D}_{H_3(z_1^r)}(\mathcal{E}_{H_3(z_1^r)}\mathcal{E}_{H_3(z_2^r)}\cdots\mathcal{E}_{H_3(z_n^r)}(E_y(m)\,\|\,h_1))\\&=\mathcal{E}_{H_3(z_2^r)}\cdots\mathcal{E}_{H_3(z_n^r)}(E_y(m_i)\,\|\,h_1)\end{aligned}$$

因此，S_1 成功地对 $b_{0,i}$ 做了“一层”解密。然后，S_1 又对 $(c_{0,i},d_{0,i})$ 做了 ElGamal 再加密操作得到 $(\hat{c}_{1,i},\hat{d}_{1,i})=E_y(h_2)$。由于 ElGamal 算法具有同态性，因此，再加密操作不会改变明文。

设 $(\hat{a}_{1,i},\hat{b}_{1,i},\hat{c}_{1,i},\hat{d}_{1,i})$ 经过 S_1 重排序之后与 $(a_{1,i},b_{1,i},c_{1,i},d_{1,i})$ 对应，则 $(a_{1,i},b_{1,i},c_{1,i},d_{1,i})$ 可以表示为

$$(a_{1,i},b_{1,i},c_{1,i},d_{1,i})=(g_1^r,\mathcal{E}_{H_3(z_2^r)}\cdots\mathcal{E}_{H_3(z_n^r)}(E_y(m_i)\,\|\,h_1),E_y(h_2))$$

依次类推，服务器 $S_j(1<j<n)$的对应输出项可以表示为

$$(a_{j,i},b_{j,i},c_{j,i},d_{j,i})=(g_j^r,\mathcal{E}_{H_3(z_{j+1}^r)}\cdots\mathcal{E}_{H_3(z_n^r)}(E_y(m_i)\,\|\,h_1),E_y(h_2))$$

最后，S_n 的输出为 $(a_{n,i},b_{n,i},c_{n,i},d_{n,i})=(g_n^r,E_y(m_i)\,\|\,h_1,E_y(h_2))$。

在对 $(c_{n,i},d_{n,i})$ 进行第 1 次 ElGamal 解密后得到 h_2。由于 P_i 忠实地按照协议要求构造了输入，因此必然有 $h_2=H_2(E_y(m_i)\,\|\,h_1)$，$(E_y(m_i),h_1,h_2)$ 被认为是有效输入。

最后在第 2 次 ElGamal 解密阶段 $E_y(m_i)$ 被解密，Mix 服务器组获得了明文消息 m_i，并将 m_i 和 $\boldsymbol{L}_M$ 中的其他明文共同输出在 BBS 上。

因此，如果所有服务器都是诚实的，则任意诚实用户输入密文所对应的明文都会出现在 Mix-net 的输出 $\boldsymbol{L}_M$ 中，证毕。

2) 可验证性

引理 1 如果将散列函数 $H_2:\{0,1\}^*\to G$ 视为 random oracle，并且在 G 中离散对数问题难解，则能够找到 2 组值 $\{x_i\}_{i=1}^N$ 和 $\{x_i'\}_{i=1}^N$，满足 $\{x_i\}_{i=1}^N\neq\{x_i'\}_{i=1}^N$ 且 $\prod_{i=1}^N H_2(x_i)=\prod_{i=1}^N H_2(x_i')$ 的概率是可忽略的。

上述定理中描述的难解问题也被称为“广义生日攻击问题”，在文献[9]中对该定理给出了简单的证明。

定理 2 第 4 节定义的 Mix-net 协议满足可验证性，即恶意服务器能够篡改某个诚实用户的消息而不被发现的概率是可忽略的。

证明(概要) ① 首先，分析没有进入“备份混洗”的情况，此时混洗过程中每个服务器提供的零知识证明 $\sigma_1,\sigma_2,\cdots,\sigma_n$ 以及 $\delta_1,\delta_2,\cdots,\delta_n$ 全部有效，并且关于诚实用户输入密文的散列验证全部通过。设用户输入 $\boldsymbol{L}_0$、M 协议第 1 步的输出 $\boldsymbol{L}_n$ 以及 M 协议第 2 步的输出 $\boldsymbol{L}_D$ 分别为

$$\begin{aligned}\boldsymbol{L}_0&=\{(a_{0,i},b_{0,i},c_{0,i},d_{0,i})\}\\&=\{(g^{r_i},\mathcal{E}_{H_3(z_1^{r_i})}\mathcal{E}_{H_3(z_2^{r_i})}\cdots\mathcal{E}_{H_3(z_n^{r_i})}(E_y(m_i)\,\|\,h_{1,i}),E_y(h_{2,i}))\}\\\boldsymbol{L}_n&=\{(a_{n,i},b_{n,i},c_{n,i},d_{n,i})\}\\&=\{(g_n^{\overline{r}_i},E_y(\overline{m}_i)\,\|\,\overline{h}_{1,i},E_y(h'_{2,i}))\}\\\boldsymbol{L}_D&=\{(g_n^{\overline{r}_i},E_y(\overline{m}_i)\,\|\,\overline{h}_{1,i},\overline{h}_{2,i})\}\end{aligned}$$

则 M 协议第 1 步中服务器提供的有效证明 $\sigma_1,\sigma_2,\cdots,\sigma_n$ 保证了 $\prod h_{2,i}=\prod h'_{2,i}$ 成立。针对 $\{E_y(h'_{2,i})\}$ 的 ElGamal 解密(M 协议第 2 步)过程中的证明 $\delta_1,\delta_2,\cdots,\delta_n$ 保证了 $\prod h'_{2,i}=\prod\overline{h}_{2,i}$ 成立。因此 $\prod h_{2,i}=\prod\overline{h}_{2,i}$ 成立。如果经过检验发现对任意的 i $(1\leqslant i\leqslant N)$都满足 $\overline{h}_{2,i}=H_2(E_y(\overline{m}_i)\,\|\,\overline{h}_{1,i})$，则必然有

$$\prod H_2(E_y(\overline{m}_i)\,\|\,\overline{h}_{1,i})=\prod H_2(E_y(m_i)\,\|\,h_{1,i})$$

根据引理 1，2 个序列 $\{E_y(\overline{m}_i)\,\|\,\overline{h}_{1,i}\}_{i=1}^N$ 和 $\{E_y(m_i)\,\|\,h_{1,i}\}_{i=1}^N$ 必然以压倒性概率相等(经过重新排序之后)。换句话说，如果有服务器在混洗中的第 1、2、3 步篡改了用户的原始输入 $E_y(m_i)\,\|\,h_{1,i}$，在必然存在某个 i 使 $\overline{h}_{2,i}\neq H_2(E_y(\overline{m}_i)\,\|\,\overline{h}_{1,i})$，因而一定会以压倒性概率被发现。

在 M 协议第 4 步对序列 $\{E_y(\overline{m}_i)\,\|\,\overline{h}_{1,i}\}_{i=1}^N$ 即 $\{E_y(m_i)\,\|\,h_{1,i}\}_{i=1}^N$ 进行联合解密时，对任意的密文 $E_y(m_i)\,\|\,h_{1,i}$，假设发送者诚实地构造了 $h_{1,i}$，那么，如果存在服务器公布了不正确的解密参数 $F_{j,i}$，则能够满足 $h_{1,i}=H_1(\prod_{j=1}^n F_{j,i})$ 概率接近于 0，因此该服务器的欺骗行为一定会被发现。

因此，只要协议规定的验证过程都能通过，就能够以压倒性概率保证诚实用户输入的明文 m_i 全部出现在 Mix-net 最终的输出 $\boldsymbol{L}_M$ 中。

② 其次，分析在混洗过程中由于某些验证失败而使系统进入“备份混洗”的情况。备份混洗中采用的秘密混洗零知识证明协议[13,17~20]的完备性

和可验证性保证了恶意服务器篡改某个诚实用户的密文消息而不被发现的概率是可忽略的，证毕。

很容易看出，协议规定的所有验证操作都不依赖于任何秘密信息和任何服务器的诚实性，因此任何第三方都可以通过 BBS 上公开的相关证明和散列值验证输出结果 $\boldsymbol{L}_M$ 的正确性，即所设计的 Mix-net 满足可公开验证性。

3) 健壮性

定理 3　针对定义 3 规定的攻击者 A，第 4 节定义的 Mix-net 协议满足健壮性。

证明　(概要)如果 n 个 Mix 服务器中最多有 $t-1$ ($2t-1 \leqslant n$)个不诚实的服务器，定理 2 保证了恶意服务器的作弊行为一定会被发现。只要恶意服务器数目小于 t，剩余的服务器就可以利用秘密共享协议共同恢复恶意服务器的密钥，并消除恶意服务器作弊行为带来的错误，使系统进入“备份混洗”程序，并最终输出正确结果。

4) 匿名性

Mix-net 协议的匿名性依赖于以下 2 个问题的困难性假设：① 群 G 上的 DDH(decisional diffie hellman problem)问题，即给定(g^x, g^r, e_b，e_{1-b})，其中，$b \in_R \{0,1\}$，$e_1 = g^{xr}$，$e_0 = g^{\theta}$ ($\theta \neq xr$)。要求判断 b 的值；② MFG-MDH(matching find guess – matching diffie hellman)联合问题[21]，其具体定义如下。

定义 8　MFG-MDH 联合问题：输入 $\{(\mathcal{E}, \mathcal{D}, \mathcal{K}, \mathcal{M}, \mathcal{C}), G, H, g^{\alpha}, g^{\alpha\beta}, f_0, g_0, f_1, g_1, s_b, g_b', s_{1-b}, g_{1-b}'\}$，其中 $(\mathcal{E}, \mathcal{D}, \mathcal{K}, \mathcal{M}, \mathcal{C})$ 表示一个对称加密方案，G=<g>为一阶为 q 的有限循环群，$H: G \to \mathcal{K}$ 为散列函数，$\alpha, \beta \in_R Z_q^*$，$b \in_R \{0,1\}$，$g_0, g_1 \in_R G$，$f_0$，$f_1 \in_R \mathcal{C}$，$g_0' = g_0^{\alpha}$，$g_1' = g_1^{\alpha}$，$s_0 = \mathcal{D}_{H(g_0'^{\beta})}(f_0)$，$s_1 = \mathcal{D}_{H(g_1'^{\beta})}(f_1)$。输出 b。

文献[21]已经证明如果满足：① 散列函数 H 能够被视为 random oracle；② 对称加密算法 $(\mathcal{E}, \mathcal{D}, \mathcal{K}, \mathcal{M}, \mathcal{C})$ 满足定义 7 的不可区分性；③ 群 G 中 DDH 问题是困难问题，那么上述 MFG-MDH 也是困难问题。

定理 4　针对定义 3 规定的攻击者 A，基于随机语言机模型、DDH 问题和 MFG-MDH 联合问题困难性假设，提出的 Mix-net 协议满足匿名性。

证明　(概要)首先，将定义 6 中的游戏转化为另一个等价游戏。对于 A 选定的 L_0、L_1 和 π，总能找到 h 个置换 $\pi_1, \cdots, \pi_h \in \Sigma_h$，满足 $\pi = \pi_1 \pi_2 \cdots \pi_h$，且 π_j 和 π_{j+1} 之间只有 2 个元素发生了位置互换。定义 h 个新游戏 $\text{Game}_1, \cdots, \text{Game}_h$。设 $L_0^{(0)} = L_0 = \{m_1, \cdots, m_h\}$，则 Game_j ($1 \leqslant j \leqslant H$)和定义 6 的游戏是相同的，除了在 Game_j 中 A 选定的 2 个明文列表为 $L_0^{(j)} = \{m_{\pi_1 \cdots \pi_{j-1}(1)}, \cdots, m_{\pi_1 \cdots \pi_{j-1}(h)}\}$ 和 $L_1^{(j)} = \{m_{\pi_1 \cdots \pi_j(1)}, \cdots, m_{\pi_1 \cdots \pi_j(h)}\}$。显然 $L_1^{(H)} = L_1 = \{m_{\pi(1)}, \cdots, m_{\pi(h)}\}$。设 A 在 Game_j 中的优势为 Adv_j，则有如下不等式成立

$$Adv_M^{\text{ANON-CCA}} \leqslant \sum_{j=1}^{H} Adv_j \tag{1}$$

现假定存在攻击者 A 能够在定义 6 的游戏中获得不可忽略的优势，则根据式(1)，必存在 j ($1 \leqslant j \leqslant H$)使得 A 在 Game_j 中获得不可忽略的优势。

模拟器按如下关键步骤为 A 模拟运行环境。

① 设有 w 个诚实服务器不受 A 控制，不妨设这些服务器为$\{S_1, \cdots, S_w\}$。其他诚实服务器分布情况可以等价转化为这种典型情况。

② 运行算法 $\mathcal{G}^{\text{MIX}}$ 生成各个 Mix 服务器的公钥和私钥，并将诚实服务器 S_1 的公钥替换成$\{y_1 = g^x, g^{\alpha}, g^{\alpha\beta}\}$。$S_1$ 需要公布的与其公钥相关的零知识证明通过控制 random oracle 的输出模拟生成。最后模拟器将修改后的公钥集合 Y 及所有恶意服务器的私钥告知 A。

③ 在 Game_j 游戏的第 2 步 A 确定了 $L_0^{(j)}$ 和 $L_1^{(j)}$ 之后，对比 $L_0^{(j)}$ 和 $L_0^{(j)}$，只存在 2 个元素位置发生了互换，不妨设为 m_0 和 m_1，对应的 2 个诚实发送者为 P_0 和 P_1。模拟器任取 $t_0, t_1 \in Z_q$，计算

$$u_0 \| v_0 \| h_{1,0} = \mathcal{D}_{g_0^{\alpha_n \beta_n}} \cdots \mathcal{D}_{g_0^{\alpha_2 \beta_2}}(s_b)$$

$$u_1 \| v_1 \| h_{1,1} = \mathcal{D}_{g_1^{\alpha_n \beta_n}} \cdots \mathcal{D}_{g_1^{\alpha_2 \beta_2}}(s_{1-b})$$

$$h_{2,0} = H_2(u_0 \| v_0 \| h_{1,0})$$

$$h_{2,1} = H_2(u_1 \| v_1 \| h_{1,1})$$

$$a_0 = g_0, b_0 = f_0$$

$$(c_0, d_0) = (g^{rt_0}, e_b^{t_0} h_{2,0})$$

$$a_1 = g_1, b_1 = f_1$$

$$(c_1, d_1) = (g^{r_{\tilde{1}}}, e_b^{t_{\tilde{1}}} h_{2,1})$$

模拟器令用户 P_0 和 P_1 的密文输出分别为以上计算所得的(a_0, b_0, c_0, d_0)和(a_1, b_1, c_1, d_1)。用户密文相关的非交互式零知识证明通过控制 random oracle 的输出模拟生成。

④ 在 M 协议第 1 步，令服务器 S_1 输出的关于 b_0 和 b_1 的部分解密结果为 s_b 和 s_{1-b}。对 c_0, d_0 和 c_1, d_1 及其他用户密文的处理方法不变。其他诚实服务器严格按照协议规定执行。

⑤ 在 M 协议第 4 步，针对 $u_0 \| v_0 \| h_{1,0}$ 和 $u_1 \| v_1 \| h_{1,1}$ 诚实服务器联合起来通过控制解密因子 $F_{j,i}$ 使解密输出分别为 m_0 和 m_1。关于解密因子 $F_{j,i}$ 的有效性证明需要通过控制 random oracle 的输出模拟生成。

模拟器在上述模拟过程中的输出全部发送给 A。最后将 A 的输出 b 作为对 DDH 问题或 MFG-MDH 联合问题的求解。

以上模拟实际上是将 DDH 问题和 MFG-MDH 问题嵌入到了 Mix-net 协议中。A 要以不可忽略的概率优势区分 b，必将面临解 2 个难题之一，证毕。

补充说明 在关于匿名性的定义 6 中没有考虑 A 能够实施选择密文攻击的情况，这是因为协议要求每开始一个新的 Mix-net 会话都要重新生成密钥。因此 Mix-net 不会被 A 用作解密预言机。

5.2 效率分析

1) 通信复杂度

设 G 中的每个元素都要用 l_G bit 表示。为发送消息 $m \in G$，Golle 的乐观 Mix-net 方案[9]需要构造 $6l_G$ bit 的密文，因此，每个服务器在混洗阶段处理的密文向量长为 $6Nl_G$ bit。在本方案中，加密 $m \in G$ 之后的密文长度为 $5l_G$+128 bit。一般情况下 l_G>128。2 种方案的通信复杂度是基本相当的。但在发送长消息时，本方案优势更为明显。设长消息必须用 k 个 G 中的元素表示。在加密该长消息时，Golle 的方案所构造的密文长为$(4k+2)$ l_G bit，而本方案构造的密文长度为$(2k+3)$ l_G+128 bit。在 k 较大时，通信量大约可以减少 1/2。这主要是因为对称加密算法在加密时长度保持不变，而 ElGamal 加密后的密文长度是明文长度的 2 倍。在新方案中减少了 ElGamal 加密算法的使用次数。

在混洗过程中，除了混洗输出结果 $\boldsymbol{L}_j$，每个 Mix 服务器 S_j 只需在 BBS 上公布非常简单的完整性证明 σ_j 和 δ_j (其长度和 N 无关)，因此，在没有服务器作弊的情况下，服务器端的通信复杂度比传统的基于零知识证明的 Mix-net 低得多。

2) 客户端的计算复杂度

因为实际的对称加密或解密算法都是非常高效的，所以在下面的计算复杂度分析中，只考虑了作为主要负载的指数运算量。将 G 上的一个指数运算记为一个 Exp。

方案中的用户每发送一个消息都要用 5+n 个 Exp 计算密文，要用 2 个 Exp 计算证明 NIZK$\{r_1, r_2 : a = g^{r_1} \wedge c = g^{r_2}\}$。因此，用户的运算量与 Mix 服务器数目成正比。这是该方案最大的缺点。但由于 Mix-net 系统的主要负载在服务器端，因此，该缺点不会对整体性能造成太大影响。

3) 服务器端的计算复杂度

为了便于比较和说明在下面的分析中没有考虑运算优化问题。在未出现服务器作弊行为的情况下，所提方案的每个服务器在混洗阶段需用 $2N$ 个 Exp 计算对称解密密钥，用 $2N$ 个 Exp 执行 ElGamal 再加密运算，为构造 σ_j 需要用 2 个 Exp，验证其他服务器的 σ 证明用 $4(n-1)$个 Exp。在第 1 次 ElGamal 解密阶段，解密、构造 δ_j 和验证其他服务器的证明分别要用 N、2 和 $4(n-1)$个 Exp。在第 2 次 ElGamal 解密阶段需要用 N 个 Exp。因此，在未出现作弊行为时每个 Mix 服务器共计需要进行 $6N+8n-4$ 个 G 上的指数运算。一般情况下 $N \gg n$，因此，所提方案单服务器的运算量几乎和服务器数目无关，即计算复杂度为 $O(N)$级。而其他所有具有公开可验证性的 Mix-net 方案的计算复杂度都为 $O(nN)$级。

表 1 给出了本节所提方案和其他具有公开可验证性的 Mix-net 方案在指数运算量方面的比较。其中 FMOS[20], Groth[19], Furukawa04[13]及 WikStrom[8]方案都属于解密混洗型 Mix-net，即将再加密、部分解密和随机排序集成在一个过程中进行。

表 1 的比较结果表明，在没有用户和服务器作弊的情况下所提出的 Mix-net 方案的计算复杂度是最低的。一旦有参与者作弊，尤其是 Mix 服务器作弊，就会使系统运行效率大幅度下降。因此，该 Mix-net 适用于服务器出错概率较低的应用中。

表 1 与其他 Mix-net 协议的单服务器指数运算量比较

协议	再加密	构造完整性证明与验证	解密
OKST97[22]	2*N*	642*Nn*	(2+4*n*)*N*
Abe99[23]	2*N*	7*N*log*N*(2*n*-1)	(2+4*n*)*N*
FS01[17]	2*N*	10*N*(2*n*-1)	(2+4*n*)*N*
Neff01[4]	2*N*	8*N*(2*n*-1)	(2+4*n*)*N*
FMOS[20]	2*N*	(10*n*-1)*N*	*N*
Groth[19]	2*N*	(8*n*-1)*N*	*N*
Furukawa04[13]	2*N*	(6*n*+2)*N*	*N*
WikStrom[8]	2*N*	(2*n*+3)*N*	*N*
Golle[9]	6*N*	12*n*-6	8*nN*
所提协议	2*N*	4*n*-2	4*N*+4*n*-2

4) 验证者的计算复杂度

由于 Mix-net 具有可公开验证性，因此，混洗及解密结束之后，任何第三方都可以通过 BBS 上公布的协议中间结果、Mix 服务器的相关零知识证明和散列值验证输出结果 $\boldsymbol{L}_M$ 的正确性。在没有作弊行为的情况下，验证者需要：① 检查 $\sigma_1,\sigma_2,\cdots,\sigma_n$ 以及 $\delta_1,\delta_2,\cdots,\delta_n$ 的有效性；② 针对协议第 4 步的输出，进行 N 次散列运算(计算 H_2 函数)；③ 针对协议第 5 步的输出，进行 N 次散列运算(计算 H_1 函数)。因此，验证者共需进行 $8n$ 次指数运算和 $2N$ 次散列运算。而此前的所有能够公开验证的方案(包括 Golle 的 Mix-net[9])，验证者都需要进行 $O(nN)$次指数运算。本方案验证运算复杂度低的特点，在大规模电子选举中非常有用。该特点使在选举结束之后，任何人都可以利用非常简单的计算设备(如 PC 机)快速检验选举结果是否可靠。

6 结束语

Mix-net 协议作为一种实现匿名性的最基本的密码学工具获得了研究者的持续关注，近十年来出现了大量相关的研究文献，也有许多高效而安全的方案被提出。但是在大规模电子选举应用中，Mix-net 协议的效率还是不能令人满意，还有很大的提升空间。而乐观型 Mix-net 协议在没有服务器作弊的情况下，每个服务器花在构造完整性证明和验证证明方面的计算量和通信量接近于常数级，验证者的计算复杂度也接近于常数级。因此，在服务器出错概率较低的应用中，采用这种方法是非常高效的。在大规模电子选举中，完全可以用加重惩罚力度的方法降低服务器作弊的概率。所提方案也存在一些缺点，例如服务器之间交互次数较多，因此，不适合用于实时匿名通信。

参考文献：

[1] CHAUM D. Untraceable electronic mail, return addresses, and digital pseudonyms[J]. Communications of the ACM, 1981, 24(2): 84-88.

[2] DINGLEDINE R, MATHEWSON N, SYVERSON P. Tor: the second-generation onion router[A]. Proceedings of the 13th USENIX Security Symposium[C]. San Antonio, 2004. 03-320.

[3] FUJIOKA A, OKAMOTO T, OHTA K. A practical secret voting scheme for large scale elections[A]. Cryptology-Asiacrypt'92[C]. Queensland, Australia, 1992. 244-251.

[4] NEFF A. A verifiable secret shuffle and its application to e-voting[A]. Proceedings of ACM CCS '01[C]. New York, USA, 2001. 116-125.

[5] GABBER E, BIBBONS P, MATIAS Y. How to make personalized Web browsing simple, secure, and anonymous[A]. Financial Cryptography '97[C]. Anguilla, 1997.17-31.

[6] JAKOBSSON M, RAIHI D. Mix-based electronic payments[A]. Proceedings of SAC '98[C]. Kingston, Canada, 1998.157-173.

[7] SEBE F, MIRET J, PUJOLIS J, *et al*. Simple and efficient hash-based verifiable mixing for remote electronic voting[J]. Computer Communications, 2010, 33(6): 667-675.

[8] WIKSTROM D. A sender verifiable Mix-net and a new proof of a shuffle[A]. Advances in Cryptology-Asiacrypt '05[C]. Chennai (Madras), India, 2005. 273-292.

[9] GOLLE P, ZHONG S, BONEH D, *et al*. Optimistic mixing for exit-polls[A]. Advances in Cryptology-Asiacrypt '02[C]. Queenstown, New Zealand, 2002. 451-465.

[10] ABE M. Flaws in some robust optimistic Mix-nets[A]. Proceedings of Information Security and Privacy, 8th Australasian Conference[C]. Wollongong, Australia, 2003.39-50.

[11] WIKSTROM D. Five practical attacks for "optimistic mixing for exit-polls"[A]. Proceedings of Selected Areas of Cryptography (SAC)[C]. Ottawa, Canada, 2003. 160-174.

[12] LONGHAI L, SHAOFENG F, XIANGQUAN C. A new relation attack on the optimistic Mix-net[A]. International Symposium on Computer Network and Multimedia Technology(CNMT 2009)[C]. Wuhan, 2009.1-4.

[13] FURUKAWA J. Efficient, verifiable shuffle decryption and its requirements of unlinkability[A]. Proceedings of PKC 2004[C]. Singapore, 2004. 319-332.

[14] CRAMER R, DAMGAARD I, SCHOENMAKERS B. Proofs of partial knowledge and simplified design of witness hiding protocols[A]. Cryptology–Crypto'94[C]. California, USA, 1994.174-187.

[15] FIAT A, SHAMIR A. How to prove yourself: practical solutions to identification and signature problems[A]. Cryptology-Crypto '86[C]. California, USA, 1987. 186-194.

[16] PEDERSEN P. Non-interactive and information theoretic secure verifiable secret sharing[A]. Advances in Cryptology: Crypto'91[C]. California, USA, 1991. 129-140.

[17] FURUKAWA J, SAKO K. An efficient scheme for proving a shuffle[A]. Proceedings of Crypto' 2001[C]. California, USA, 2001. 368-387.

[18] NEFF A. A verifiable secret shuffle and its application to e-voting[A]. Proceedings of ACM CCS '01[C]. New York, USA, 2001. 116-125.

[19] GROTH J. A verifiable secret shuffle of homomorphic encryptions[J]. Journal of Cryptology, 2010, 23(4): 546-579.

[20] FURUKAWA J, MIYAUCHI H, MORI K. An implementation of a universally verifiable electronic voting scheme based on shuffling[A]. Proceedings of Financial Cryptography'02. [C] Southampton, Bermuda, 2002. 16-30.

[21] OHKUBO M, ABE M. A length-invariant hybrid mix[A]. Cryptology-Asiacrypt'00[C]. Kyoto, Japan, 2000. 178-191.

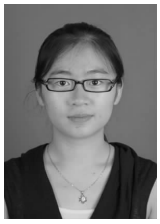
[22] OGATA W, KUROSAWA K, SAKO K, *et al*. Fault tolerant anonymous channel[A]. Proceedings of. ICICS '97[C]. South Africa, 1997. 440-444.

[23] ABE M. Mix-networks on permutation networks[A]. Cryptology-Asiacrypt'99[C]. Singapore, 1999. 258-273.

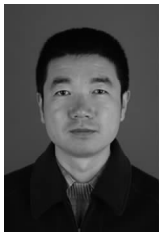
作者简介：

李龙海（1976-），男，河北冀州人，博士，西安电子科技大学副教授、硕士生导师，主要研究方向为匿名通信、隐私保护技术和计算机网络安全。

黄诚强（1989-），男，福建福州人，西安电子科技大学硕士生，主要研究方向为计算机与网络安全。



许尚妹（1990-），女，浙江杭州人，西安电子科技大学硕士生，主要研究方向为计算机与网络安全。



付少锋（1975-），男，陕西户县人，西安电子科技大学副教授，主要研究方向为计算机网络安全和嵌入式系统。