第 34 卷第 11 期 通 信 学 报 Vol.34 No. 11
2013 年 11 月 Journal on Communications November 2013
doi:10.3969/j.issn.1000-436x.2013.11.017

基于双线性对的乐观 Mix-net 协议

李龙海，付少锋，苏锐丹
（西安电子科技大学 计算机学院，陕西 西安 710071）

摘 要：提出了一种新的基于双线性对的乐观 Mix-net 协议。利用双线性对工具简化了密钥管理，在不同的协议会话中服务器端不用重新生成密钥，并且当前会话不会为其他会话提供解密预言机服务。采用了“哑元追踪法”保证混洗过程的完整性，简化了正确性证明的构造。对 ElGamal 联合解密过程做了优化，降低了每个服务器节的指数运算量。在没有服务器作弊的情况下，对输入密文组的混洗和解密速度比其他可公开验证的 Mix-net 方案高得多。

关键词：匿名通信；乐观混合网络；双线性对；秘密混洗证明
中图分类号：TP393.08 **文献标识码：**B **文章编号：**1000-436X(2013)11-0153-09

Optimistic Mix-net protocol based on bilinear pairings

LI Long-hai, FU Shao-feng, SU Rui-dan
(School of Computer Science and Technology, Xidian University, Xi'an 710071, China)

Abstract: A novel pairing-based optimistic Mix-net scheme was proposed. The key management is made easier by employing bilinear paring primitives and there is no need for the participating mix servers to re-generate keys jointly between mix-sessions to avoid providing decryption oracle service to other mix-sessions. Integrity of messages during mixing is partially guaranteed by using "dummy messages tracing" technology resulting in a simpler construction for proofs of correctness. An optimization method for the joint E mal decryption involved in the protocol was also proposed, which can reduce the number of exponentiations computed by each mix server. The Mix-net will shuffle and decrypt input ciphertexts much faster than all previous Mix-nets with public verifiability when all mix servers execute the mixing protocol honestly.

Key words: anonymous communication; optimistic mix network; bilinear pairings; proof of secret shuffling

1 引言

Mix-net 是由 Chaum 最早提出的用于实现匿名通信的密码学协议[1]。Mix-net 协议的功能可以概括为：混洗并解密，即输入一组密文$(c_1, c_2,\cdots, c_N)$，输出解密后所得明文组$(m_1, m_2,\cdots, m_N)$的一个随机置换。如果该置换保密并且加密算法是语义安全的，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 协议已被广泛应用于匿名电子邮件[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]等需要保护用户隐私的互联网应用系统中。

Mix-net 的简单实现是只用一个 Mix 服务器完成解密和混洗操作，其缺点是用户必须完全信任该服务器。为提高安全性，在复杂实现中多个 Mix 服务器被级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。在这种级联方案中，只要有一个 Mix 服务器是诚实的，就能够保证输入密文和输出明文之间的不可关联性(unlinkability)。设计 Mix-net 协议的难点是如何保证所处理消息集合的完整性，即如何防止恶意 Mix 服务器在混洗过程中替换或篡改部分消息。针对完整性问题，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

收稿日期：2012-11-01；修回日期：2013-10-15
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)

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。精心设计的证明协议不会暴露服务器使用的秘密置换，任何人都可以验证该服务器混洗操作的正确性，验证通过之后再启动下一级Mix服务器的操作。基于零知识证明的Mix-net协议的主要问题是每个服务器构造混洗证明都需要进行 $O(N)$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 表示输入向量中的元素个数，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)，该运算还是比较耗时的。根据Sebe等的实验，“在输入向量中包含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 h”[7]。

为进一步提高效率，Golle等提出了一种基于乐观验证策略的Mix-net[8]协议。在正常情况下，该协议不要求逐级证明和验证，只做系统输入和输出端的整体验证；只有在端到端的验证失败时再做逐级证明和验证。因此，在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，其计算复杂度则略高于传统的基于零知识证明的Mix-net协议。Golle的协议被发现存在严重安全漏洞，在文献[9~11]中先后提出了多种攻击方法破坏其匿名性和健壮性。为了避免WikStrom等提出的选择密文攻击[10,11]，Golle的协议必须每执行一次都要重新生成和发布密钥。Abe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[9]。

为解决上述问题，本文设计了一种新的基于双线性对的乐观Mix-net协议。该协议具有如下特点：1) 密钥管理简单，在不同的协议会话中服务器端不用重新生成秘密钥和公开钥，并且协议的当前执行不会为其他会话(包括并发执行的会话)提供解密预言机服务，因而对文献[10]和文献[11]的选择密文攻击免疫；2) 采用“哑元追踪法”保证混洗过程的正确性，这样不但可以有效防御WikStrom的攻击[10]，而且简化了完整性证明的构造；3) 对ElGamal联合解密过程做了优化，简化了构造和验证非交互式证明的工作，为每个Mix服务器节省了大约 $(6n-3)N$ 个指数运算。在没有作弊行为的情况下，所提协议比基于零知识证明的可公开验证的Mix-net协议的计算复杂度要低得多。

2 相关工作

1981年，Chaum首次引入了Mix-net的概念[1]，并给出了一种基于RSA加密算法的Mix-net实现。Pfitzmann等[12]发现Chaum的Mix-net无法抵御选择密文攻击。1993年，Park等[13]提出了基于ElGamal体制的再加密型Mix-net实现模型。与传统的逐级解密的Mix-net相比，再加密型Mix-net具有密钥管理简单、密文长度与服务器数目无关、混洗与解密过程相互独立、容错性好等优点，因此自Park之后在此方向出现了大量的研究文献。由于提出的Mix-net也属于再加密型，因此这里对与再加密型Mix-net协议相关的工作进行重点讨论。

如何在不影响秘密性的前提下用较低的代价保证混洗消息集合的完整性是设计再加密型Mix-net的难点所在。针对该问题目前已经提出的解决方案主要包括如下几种。

1) 利用零知识证明技术。早期的基于“分割—选择”方法的Mix证明协议，如Sako的方案[14]和Ogata的方案[15]，计算和通信复杂度相当高。每个服务器构造和验证Mix证明分别需要进行 $O(N?)$次和 $O(Nn?)$次指数运算，其中 $?$ 为安全参数，一般取 $?=80$。Abe和Jakobsson分别在文献[16]和文献[17]中提出了基于“置换网络”(permutation network)[18]的Mix证明协议，使得构造Mix证明的计算复杂度降为 $O(N\log N)$。Furukawa和Sako在文献[19]中利用“置换矩阵”(permutation matrix)技术设计了一种高效的Mix证明协议，服务器在构造证明时需要进行 $10N$ 个指数运算。Neff在文献[20]中利用“指数向量点乘”技术设计了一种效率更高的Mix证明，使得每个服务器的指数计算量为 $8N$。Groth在文献[21]中将Neff的协议推广到了任意具有同态特性的公钥体制。Nguyen等将Furukawa的技术推广到了Paillier体制[22]。Kun Peng的工作[23]使构造证明指数运算量降为 $6N$，而WikStrom等提出的Mix证明协议[24]使计算量降为 $3N$ 左右，是目前已知的效率最高的证明方法。但不幸的是WikStrom的Mix-net被Kun Peng发现“在正确性、匿名性和健壮性方面存在多处安全漏洞，并且一些严重漏洞被证明是不可修复的”[25]。

2) 设置多个参照组对比混洗结果。主要思想是设置多个参照组分别对输入密文向量进行再加密和混洗，然后分别解密，最后对比各个参照组的解

密结果是否一致。如果一致，则说明结果是正确的；如果不一致，则用逆向追踪法找出作弊者，然后重复该过程。当然，为了在解密对比时不提前暴露秘密信息，应该首先对输入进行盲化。只有在确认混洗过程无误的情况下，再进行“反盲化”。基于该思想的 Mix-net 协议包括[26,27]，但这些方案都被发现存在安全漏洞[28,29]。总的来讲，在基于多参照组冗余计算的 Mix-net 系统中单服务器的计算复杂度为 $O(t\ N)$，其中 t 表示参照组数目。在 N 较大时 t 的值可以很小。

3) 利用随机抽取部分样本检验的方法。这类 Mix-net 协议[30,31]构造和验证完整性证明的代价较小，但健壮性低于其他类方案。Mix 服务器可以篡改少量的消息而不被发现，因此只适用于大规模的电子选举。

4) 基于乐观验证策略的 Mix-net。Golle 等最早提出了乐观验证型(optimistic verifiable) Mix-net[8]方案。该 Mix-net 要求输入密文作双层加密，并且在混洗过程中不作复杂的逐级验证，混洗结束后作单层解密，并直接验证最后的混洗结果是否正确。如果不正确，再利用复杂度较高基于零知识证明的 Mix-net 对只有单层加密的密文向量进行混洗。在没有服务器作弊的情况下，单服务器的运算量很低，并且主要是不可避免的再加密和联合门限解密的开销。每个服务器花在证明方面的计算量接近于常数级，与输入向量中元素个数 N 无关。另外 Sebe 的 Mix-net 协议[7]也采用了类似的乐观验证的思想。

文献[10~12]相继发现 Golle 的乐观型 Mix-Net 存在安全漏洞：1)用户输入密文具有可展性；2)每个服务器没有严格检查其输入是否为特定循环群 G 中的元素；3)协议的当前执行会为其他会话(包括并发执行的会话)提供解密预言机服务；4)逐级检验存在漏洞使得某些恶意服务器可以“抵消”其上游服务器的混洗效果并破坏匿名性。

本文设计的基于双线性对的乐观 Mix-net 协议可以有效弥补上述安全漏洞，并且简化了密钥管理：在不同的协议会话中服务器端不用重新生成秘密钥和公开钥。另外，还将乐观验证思想运用到 Mix-net 的联合解密过程中，为每个 Mix 服务器节省了大约 $(6n-3)N$ 个指数运算。

3 Mix-net 协议详细设计

3.1 系统的建立

设 G_1 和 G_2 是 2 个阶为 q 的有限循环群，q 为素数。其中，G_1 为加法群，G_2 为乘法群，Q 为 G_1 的生成元。$\hat{e}: G_1 \times G_1 \to G_2$ 是一个可接受的双线性对映射[32]，要求关于 $\hat{e}$ 双线性 Diffie-Hellman 判定(DBDH)[32]问题难解。该假设也蕴含了 G_1 上的 Diffie-Hellman 计算(CDH)问题难解和 G_2 上的 Diffie-Hellman 判定(DDH)问题难解。

系统由 N 个用户和 n 个 Mix 服务器组成，他们通过具有认证功能的可靠的广播系统(BBS 系统)进行通信。n 个服务器利用 Shamir (t,n)门限方案[33]共享秘密钥 $x \in_R Z_q$，并以 $y = xQ$ 作为系统的公开钥。设 x_i 表示服务器 S_i 所掌握的关于 x 的秘密份额，$y_i = x_i Q$ 为相应的公开钥，并且 S_i 已将 y_i 公布在了 BBS 上。

另外，还需要定义 3 个安全的密码学哈希函数：$H_1: \{0,1\}^* \to G_1$，$H_2: G_2 \times G_2 \to G_2$ 和 $H_3: \{0,1\}^* \to \{0,1\}^{len}$，$len$ 表示某个安全长度，例如令 len 等于 128。在安全性分析中它们都将被视为 Random Oralce[34]。

$\mathrm{NIZK}\{x_1, x_2, \mathrm{L}, x_e : A(x_1, x_2, \mathrm{L}, x_e)\}$ 表示关于秘密值 $x_1, x_2, \mathrm{L}, x_e$ 的非交互式零知识证明，A 是关于离散对数的某个逻辑命题。示证者在不暴露 $x_1, x_2, \mathrm{L}, x_e$ 的条件下证明命题 A 为真。其具体构造方法可以参考文献[35,36]。

3.2 可验证 ElGamal 门限解密 VTDec(*u*, *v*)

协议的输入为一个 ElGamal 密文对 $(u,v) = (g^s, y^s m)$，由 n 个服务器构成的服务器组对 (u,v) 进行联合解密。其具体过程如下。

1) 任选 t 个服务器，假设为 $S_1, S_2, \cdots, S_t$。

2) 服务器 S_i $(1 \le i \le t)$计算并公布 $D_i = u^{-x_i L_i}$，其中 $L_i = \prod_{1 \le j \le t, j \ne i} \frac{j}{j-i}$ 为拉格朗日参数。为表明其操作的正确性，S_i 必须同时公布非交互式证明

$$\mathrm{NIZK}\{x_i : y_i = g^{x_i} \wedge D_i = (u^{-L_i})^{x_i}\}$$

3) 等所有 t 个服务器公布完毕之后，每个服务器都验证其他服务器公布的证明是否正确。如果某服务器提供的证明验证无效，则确定其为作弊者，并由备用服务器替代他参与解密协议。如果全部有效，则输出解密明文

$$m = v\prod_{i=1}^{t} D_i = my^s u^{-\sum_{i=1}^{t} x_i L_i} = my^s g^{s(-\sum_{i=1}^{t} x_i L_i)}$$

$$= my^s g^{-sx} = my^s y^{-s}$$

3.3 会话前准备

每开始一个新的以 sid 为标识号的 Mix-net 会

话，系统的参与者首先做如下准备工作。

1) 计算仅能够用于本次会话的 ElGamal 公开钥 (Q_{sid}, Y_{sid})，其中，$Q_{sid}=\hat{e}(Q, H_1(sid))$，$Y_{sid}=\hat{e}(y, H_1(sid))=\hat{e}(xQ, H_1(sid))=Q_{sid}^{x}$。

注意每个参与者都可以计算该公开钥，不用由服务器端广播给所有用户。这是本协议简化密钥管理的关键所在。

下面用 $E_{Q_{sid}, Y_{sid}}(m)$ 表示工作在群 G_2 上的以 (Q_{sid}, Y_{sid}) 为公开钥的 ElGamal 加密函数。

2) 服务器 S_j（$j=1, 2, \cdots, n$）生成包含 D 个 6 元组的随机向量

$$\{(\alpha_{j,i}, \beta_{j,i}, \gamma_{j,i}, \delta_{j,i}, \varepsilon_{j,i}, \phi_{j,i})\}_{i=1}^{D} \in_R G_2^{6D}$$

上述向量用于构造哑元，S_i 将该向量保密。此时只需公布承诺向量

$$\{C_{j,i}\}_{i=1}^{D}=\{H_3(\alpha_{j,i} \| \beta_{j,i} \| \gamma_{j,i} \| \delta_{j,i} \| \varepsilon_{j,i} \| \phi_{j,i})\}_{i=1}^{D}$$

关于哑元数目 D 的计算方法将在 4.1 节中进行讨论。

3.4 详细的混洗过程

1) 用户发送消息

为发送消息 $m \in G_2$，用户 P_i 任取 $s, r_1, r_2, r_3 \in_R Z_q$，并计算

$$(u, v)=E_{Q_{sid}, Y_{sid}}(m)=(Q_{sid}^{s}, Y_{sid}^{s} m) \quad w=H_2(u, v)$$

$$q_i=((a, b), (c, d), (e, f))$$

$$=(E_{Q_{sid}, Y_{sid}}(u), E_{Q_{sid}, Y_{sid}}(v), E_{Q_{sid}, Y_{sid}}(w))$$

$$=((Q_{sid}^{r_1}, Y_{sid}^{r_1} Q_{sid}^{s}), (Q_{sid}^{r_2}, Y_{sid}^{r_2} Y_{sid}^{s} m), (Q_{sid}^{r_3}, Y_{sid}^{r_3} w))$$

为防御选择密文攻击，且使用户输入密文不具有可展性，P_i 必须同时构造非交互式零知识证明：NIZK $\{s, r_1, r_2, r_3 : a=Q_{sid}^{r_1} \wedge b=Y_{sid}^{r_1} Q_{sid}^{s} \wedge c=Q_{sid}^{r_2} \wedge e=Q_{sid}^{r_3}\}$。

最后，P_i 将 q_i 和上面的零知识证明公布在 BBS 上作为自己的输入。

2) 插入哑元

等所有用户公布完毕或到达某个时间限之后，服务器组首先对 BBS 中的每个用户输入所附带的非交互证明进行验证，并去掉无效的输入。设经过验证之后获得的有效 6 元组向量为 $\{?_1, ?_2, \cdots, ?_N\}$。具体的插入哑元过程如下。

① 服务器 S_j（$j=1, 2, \cdots, n$）公布哑元向量 $\{(\alpha_{j,i}, \beta_{j,i}, \gamma_{j,i}, \delta_{j,i}, \varepsilon_{j,i}, \phi_{j,i})\}_{i=1}^{D}$。

② 等所有服务器公布完毕之后，验证各个服务器公布的哑元向量与会话准备阶段公布的承诺向量 $\{C_{j,i}\}_{i=1}^{D}$ 是否一致。如果某服务器无法通过验证，则将其视为作弊者从系统中删除。他公布的哑元向量也将被忽略。现假设所有服务器都是诚实的。

③ 任意服务器都可以计算联合生成的哑元向量

$$\text{Dummy}=\{(\prod_{j=1}^{n}\alpha_{j,i}, \prod_{j=1}^{n}\beta_{j,i}, \prod_{j=1}^{n}\gamma_{j,i}, \prod_{j=1}^{n}\delta_{j,i},$$

$$\prod_{j=1}^{n}\varepsilon_{j,i}, \prod_{j=1}^{n}\phi_{j,i})\}_{i=1}^{D}$$

④ 最后将向量 Dummy 中的 D 个 6 元组全部插入到向量 $\{?_1, ?_2, \cdots, ?_N\}$ 的最尾部，得到下一阶段的输入 $\boldsymbol{L}_0=\{(a_{0,i}, b_{0,i}, c_{0,i}, d_{0,i}, e_{0,i}, f_{0,i})\}_{i=1}^{N+D}$。

3) 再加密和混洗

n 个 Mix 服务器按照如下步骤对 L_0 进行再加密和混洗。

① 每个服务器 S_j 首先从 BBS 上读取前一个服务器的输出

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

② 服务器 S_j 任取 $3N$ 个随机数 $r_{ji}, s_{ji}, t_{ji} \in_R Z_q$ $(1 \le i \le N)$，对 $\boldsymbol{L}_{j-1}$ 进行再加密得到向量

$$\{(Q_{sid}^{r_{ji}} a_{j-1,i}, Y_{sid}^{r_{ji}} b_{j-1,i}, Q_{sid}^{s_{ji}} c_{j-1,i}, Y_{sid}^{s_{ji}} d_{j-1,i},$$

$$Q_{sid}^{t_{ji}} e_{j-1,i}, Y_{sid}^{t_{ji}} f_{j-1,i})\}_{i=1}^{N+D}$$

然后 S_j 任取置换 $p_j: \{1, \cdots, N+D\} \to \{1, \cdots, N+D\}$，并按照 p_j 对上述向量进行重排序，得到输出向量：$\boldsymbol{L}_j=\{(a_{j,i}, b_{j,i}, c_{j,i}, d_{j,i}, e_{j,i}, f_{j,i})\}_{i=1}^{N+D}$。

③ 设 $e_j=\prod_{i=1}^{N+D} e_{j,i}$，$f_j=\prod_{i=1}^{N+D} f_{j,i}$，$e_{j-1}=\prod_{i=1}^{N+D} e_{j-1,i}$，$f_{j-1}=\prod_{i=1}^{N+D} f_{j-1,i}$。$S_j$ 构造非交互式知识证明

$$s_j=\text{NIZK}\{t_j : e_j e_{j-1}^{-1}=Q_{sid}^{t_j} \wedge f_j f_{j-1}^{-1}=Y_{sid}^{t_j}\}$$

S_j 在构造 s_j 时使用的秘密输入 (Witness) 为 $t_j=\sum_{i=1}^{N} t_{ji}$。

④ 最后，S_j 将 L_j 公布在 BBS 上作为下一个服务器的输入。S_j 同时公布 s_j 让其他服务器验证。如果 s_j 被验证无效，则确定 S_j 为作弊者，将 S_j 从系统中删除，然后 S_{j+1} 以 L_{j-1} 为输入进行再加密和混洗操作。

最后一个服务器 S_n 的输出 L_n 是再加密和混洗

阶段的最终输出。

4) 追踪哑元

从服务器 S_1 到 S_n 依次公布关于 D 个哑元的再加密参数和置换路径。设 $\{q_{N+i}\}_{i=1}^{D}=\{(a_{0,N+i},b_{0,N+i},c_{0,N+i},d_{0,N+i},e_{0,N+i},f_{0,N+i})\}_{i=1}^{D}$ 为 L_0 中的 D 个哑元，则具体的追踪过程如下。

① 设 $\{k_{0,1},\cdots,k_{0,D}\}=\{N+1,\cdots,N+D\}$，对 $j=1,2,\cdots,n$，服务器 S_j 依次公布向量

$$\{k_{j,1},k_{j,2},\cdots,k_{j,D}\}$$

$$=\{p_j(k_{j-1,1}),p_j(k_{j-1,2}),\cdots,p_j(k_{j-1,D})\}$$

$$\{(r'_{j,i},s'_{j,i},t'_{j,i})\}_{i=1}^{D}=\{(r_{j,k_{j-1,i}},s_{j,k_{j-1,i}},t_{j,k_{j-1,i}})\}_{i=1}^{D}$$

② 等所有服务器公布完毕之后，验证：对任意的 $i\in\{1,2,\cdots,D\}$，满足

$$a_{n,k_{n,i}}=Q_{sid}^{\sum_{j=1}^{n}r'_{j,i}}a_{0,N+i},\quad b_{n,k_{n,i}}=Y_{sid}^{\sum_{j=1}^{n}r'_{j,i}}b_{0,N+i},$$

$$c_{n,k_{n,i}}=Q_{sid}^{\sum_{j=1}^{n}s'_{j,i}}c_{0,N+i},\quad d_{n,k_{n,i}}=Y_{sid}^{\sum_{j=1}^{n}s'_{j,i}}d_{0,N+i}$$

$$e_{n,k_{n,i}}=Q_{sid}^{\sum_{j=1}^{n}t'_{j,i}}e_{0,N+i},\quad f_{n,k_{n,i}}=Y_{sid}^{\sum_{j=1}^{n}t'_{j,i}}f_{0,N+i}$$

③ 如果发现对某个值 i，不能满足上面的等式，那么根据各个服务器公布的关于哑元的再加密参数、置换路径以及该服务器在混洗阶段的输入、输出，从 S_1 到 S_n 依次验证各个服务器是否对哑元进行了正确的再加密和混洗。由于篇幅所限，其具体过程就不再给出了。经过这样的逐级验证，就可找出不诚实的服务器。将这些服务器删除之后，重新回到主协议第 2)步(插入哑元阶段)运行，当然此时也要重新生成哑元。

④ 如果验证通过，则从 L_n 去掉元素 $\{(a_{n,k_{n,j}},b_{n,k_{n,j}},c_{n,k_{n,j}},d_{n,k_{n,j}},e_{n,k_{n,j}},f_{n,k_{n,j}})\}_{i=1}^{D}$ 得到只包含 N 个 6 元组的向量 $L'_n=\{(\overline{a}_i,\overline{b}_i,\overline{c}_i,\overline{d}_i,\overline{e}_i,\overline{f}_i)\}_{i=1}^{N}$。

5) 外层 ElGamal 解密

对 L'_n 进行外层 ElGamal 解密时，没有直接采用 VTDec 协议，而是采用了一种优化的联合解密方法。

设 $L_j=\prod_{1\le i\le t,i\ne j}\frac{i}{i-j}$，$\overline{e}=\prod_{i=1}^{N}\overline{e}_i$，$\overline{f}=\prod_{i=1}^{N}\overline{f}_i$，服务器组执行以下操作。

① 任选 t 个服务器，假设为 $S_1,S_2,\cdots,S_t$。

② 服务器 S_j $(1\le j\le t)$ 计算并公布下列内容：$\{(D_{a,j,i},D_{c,j,i},D_{e,j,i})\}_{i=1}^{N}=\{(\overline{a}_i^{-x_jL_j},\overline{c}_i^{-x_jL_j},\overline{e}_i^{-x_jL_j})\}_{i=1}^{N}$，$D_{e,j}=\prod_{i=1}^{N}D_{e,j,i}$，$d_j=\mathrm{NIZK}\{x_j:D_{e,j}=(\overline{e}^{-L_j})^{x_j}\wedge y_j=x_jQ\}$。

③ 等所有 t 个服务器公布完毕之后，任意服务器计算：$\overline{w}=\overline{f}\prod_{j=1}^{t}D_{e,j}$，$\{(\overline{u}_i,\overline{v}_i,\overline{w}_i)\}_{i=1}^{N}=\{(\overline{b}_i\prod_{j=1}^{t}D_{a,j,i},\overline{d}_i\prod_{j=1}^{t}D_{c,j,i},\overline{f}_i\prod_{j=1}^{t}D_{e,j,i})\}_{i=1}^{N}$。

④ 服务器组验证 $\overline{w}=\prod_{i=1}^{N}\overline{w}_i$ 是否成立。如果不成立，则说明有服务器在该阶段作弊，此时再运行 VTDec 协议对 L'_n 进行复杂度较高的解密操作。如果成立，则以 $\overline{L}=\{(\overline{u}_i,\overline{v}_i,\overline{w}_i)\}_{i=1}^{N}$ 为解密输出进入下一阶段。

只做上述简单验证就能保证解密正确性的原因在定理 1 的证明中会做具体分析。

注意在以上优化的联合解密方法中，每个服务器只需公布一个简单的证明 d_j，而在传统的 VTDec 协议中，服务器针对密文向量中的每一个元素都要公布一个类似的证明，因此需要广播的证明长度缩短了 N 倍，大大降低了计算和通信复杂度。

6) 验证混洗结果

服务器组对 $\overline{L}$ 中的任意 3 元组 $(\overline{u}_i,\overline{v}_i,\overline{w}_i)$ 验证 $\overline{w}_i=H_2(\overline{u}_i,\overline{v}_i)$ 是否成立，如果不成立，则对 $(\overline{u}_i,\overline{v}_i,\overline{w}_i)$ 进行反向追踪，即从 S_n 到 S_1 依次要求各个服务器公布与 $(\overline{u}_i,\overline{v}_i,\overline{w}_i)$ 相关的部分置换路径和再加密参数，然后由大家共同验证每个服务器 S_j 公布的内容与其输入 L_{j-1} 和输出 L_j 是否一致。

如果反向追踪过程表明 $(\overline{u}_i,\overline{v}_i,\overline{w}_i)$ 是良性的(即所有服务器对 $(\overline{u}_i,\overline{v}_i,\overline{w}_i)$ 都执行了正确操作，该 3 元组的无效是发送用户本身造成的)，则忽略该 3 元组，并继续程序的执行。如果反向追踪过程表明 $(\overline{u}_i,\overline{v}_i,\overline{w}_i)$ 是恶性的，即有一个或多个服务器对该 3 元组执行了错误操作，则将这些恶意服务器删除之后，跳转到第 8)步执行备份混洗程序。

7) 内层 ElGamal 解密

如果经过第 7)步的检验发现 $\overline{L}$ 中每个 3 元组都是有效的，则服务器组利用 VTDec 协议对 $\overline{L}$ 中的每个密文对 $(\overline{u}_i,\overline{v}_i)$ 进行门限解密，获得最终的明文输出向量 $L_M=\{m_1,\cdots,m_N\}$。

8) 备份混洗

① 如果发现 $\overline{L}$ 中的某个 3 元组 $(\overline{u}_i,\overline{v}_i,\overline{w}_i)$ 是恶性的，则通过反向追踪找到与之对应的输入 $((a_k,b_k),(c_k,d_k),(e_k,f_k))$，然后任选 t 个服务器对该输入进行门限解密得到 $(\%_i,\%_i,\%_i)$，并用 $(\%_i,\%_i,\%_i)$ 替换 $\overline{L}$ 中的 $(\overline{u}_i,\overline{v}_i,\overline{w}_i)$。

② 设 $\tilde{L}=\{(\tilde{u}_i,\tilde{v}_i)\}_{i=1}^{N}$ 表示对 $\bar{L}$ 修正完毕之后得到的只有单层加密的密文向量。以 $\tilde{L}$ 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 $L_M=\{m_1,\cdots,m_N\}$。

4 分析

4.1 安全性分析

1) 完整性

引理1 设散列函数 H 的输出为群 G 中的元素，如果能够将 H 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 $\{(u_i,v_i)\}_{i=1}^{N}\neq\{(\bar{u}_i,\bar{v}_i)\}_{i=1}^{N}$ 满足等式

$$\prod_{i=1}^{N}H(u_i,v_i)=\prod_{i=1}^{N}H(\bar{u}_i,\bar{v}_i)$$

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[8]中对该定理也给出了简单的证明。

定理1 在上述Mix-net协议中，Mix服务器在执行混洗操作时能够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。

证明(概要) 设 $L_0=\{(E(u_i),E(v_i),E(w_i))\}$，$L_n'=\{(E(u_i'),E(v_i'),E(w_i'))\}$，$\bar{L}=\{(\bar{u}_i,\bar{v}_i,\bar{w}_i)\}$。混洗过程中服务器提供的有效证明 $s_1,s_2,\cdots,s_n$ 保证了 $\prod w_i=\prod w_i'$ 成立。在外层 ElGamal 解密阶段的证明 d_j 和该阶段第4)步中的验证操作保证了 $\prod w_i'=\prod\bar{w}_i$ 成立。因此 $\prod w_i=\prod\bar{w}_i$ 成立。

如果混洗结束后经过检验发现对任意的 i $(1\le i\le N)$ 都满足 $\bar{w}_i=H_2(\bar{u}_i,\bar{v}_i)$，则必然满足等式 $\prod H_2(u_i,v_i)=\prod H_2(\bar{u}_i,\bar{v}_i)$。

根据引理1，2个序列 $\{(u_i,v_i)\}_{i=1}^{N}$ 和 $\{(\bar{u}_i,\bar{v}_i)\}_{i=1}^{N}$ 必然以压倒性概率相等。换句话说，如果有服务器在混洗和外层ElGamal解密阶段改变了用户的原始输入，在必然存在某个 i 使得 $\bar{w}_i\neq H(\bar{u}_i,\bar{v}_i)$，因而一定会以压倒性概率被发现。

在内层 ElGamal 解密阶段，用户的原始输入序列 $\{(u_i,v_i)\}_{i=1}^{N}$ 利用可公开验证的联合解密协议 VTDec 进行解密，最终获得明文输出 $L_M=\{m_1,\cdots,m_N\}$。此时，服务器能够非法解密某个诚实用户的密文消息而不被发现的概率是可忽略的。

上述完整性保证使得诚实用户输入密文所对应的明文以压倒性概率全部出现在输出 L_M 中。如果某个不诚实用户的输入(E(u), E(v), E(w))不满足 $w=H_2(u,v)$，那么该输入会在协议的第6)步中被忽略。

另外，任意的观察者都可以验证 $s_1,s_2,\cdots,s_n$ 的有效性以及 $\bar{w}_i=H_2(\bar{u}_i,\bar{v}_i)$ 是否成立，因此所提方案的完整性是可公开验证的。

2) 匿名性

设 $q_i\in\{?_1,?_2,\cdots,?_N\}$ $(1\le i\le N)$ 是某个诚实用户发送给 Mix-net 系统的密文消息，并且 Mix-net 的输出 $L_M=(m_1,m_2,\cdots,m_N)$ 中没有相互重复的消息出现。设 h 表示参与当前 Mix-net 会话的诚实用户的个数。如果攻击者能够获得 $j(1\le j\le N)$ 满足 $\mathrm{Dec}(q_i)=m_j$ 的概率与 $1/h$ 相比只具有可忽略的优势，则称该 Mix-net 具有匿名性。即如果攻击者能够发现输入密文与输出明文之间对应关系的概率不优于随机猜测，则称 Mix-net 具有匿名性。

在这里只分析没有服务器作弊的情况。在有服务器作弊时，匿名性是由基于零知识证明的 Mix-net 协议[4]保证的。

主协议第3)步的“再加密和混洗”过程对密文向量做了随机排序，隐藏了输入与输出的对应关系。对于被动攻击者而言，若直接通过观察向量 L_i $(i=0,1,\cdots,n)$ 以及 $\bar{L}$ 猜测输入与输出的对应关系，则必将面临解 G_2 上的 DDH 问题。混洗阶段另外的输出的 s_j 具有零知识性，也不会暴露输入输出对应关系。

现在分析存在主动攻击者的如下情况。

① 用户输入中的非交互式证明保证了输入是不可展的，因此恶意用户无法进行主动攻击。

② “会话前准备”阶段的承诺向量 $\{C_{j,i}\}_{i=1}^{D}$ 使得每个服务器在“插入哑元”阶段无法根据其他成员公布的部分哑元信息随意修改自己的哑元信息，即相当于所有服务器“同时”公布了哑元信息。因此只要有一个服务器随机选取了自己的部分哑元，就能够保证最终生成的哑元是随机的。因此，恶意服务器无法在哑元中插入选择性密文。

③ 在混洗和外层解密阶段(③~⑤步)，如果恶意服务器只篡改输入向量中的部分密文，则服务器提供的证明 s_j 和 d_j 保证了这种主动攻击一定会被发现。

④ Wikstrom 的攻击[10]表明，恶意服务器 S_j 在混洗过程中虽然无法用完全不同的向量 $\{(u_i',v_i',w_i')\}_{i=1}^{N}$ 替换 L_{j-1} 中封装的向量 $\{(u_i,v_i,w_i)\}$，但他可以将 L_{j-1} “整体替换”为 L_k $(0\le k<j-1)$ 的再加密形式。该漏洞虽然不会影响完整性，但可以

影响匿名性。在方案中设置D个哑元的主要目的就是为了防止恶意服务器的这种“整体替换”操作。如果某恶意服务器 S_j 要想将 $\boldsymbol{L}_{j-1}$ “整体替换”为 $\boldsymbol{L}_k$ $(0 \le k < j-1)$的再加密形式，为了不被“哑元追踪”过程发现，他必须使 $\boldsymbol{L}_{j-1}$ 中的 D 个哑元保持不变，因此他必须知道这些哑元在 $\boldsymbol{L}_{j-1}$ 中的位置。只要 $S_1, S_2, \text{L}, S_{j-1}$ 中有一个是诚实的，那么 S_j 能够正确猜中这些位置的概率为

$$\frac{1}{N(N-1)\text{L}(N-D+1)}$$

如果 $S_1, S_2, \text{L}, S_{j-1}$ 全部是不诚实的，那么 S_j 知道这 D 个哑元在 $\boldsymbol{L}_{j-1}$ 中的位置，但 S_j 也同时知道 $\boldsymbol{L}_k$ $(0 \le k < j-1)$与 $\boldsymbol{L}_0$ 的元素对应关系。因此，虽然 S_j 能够将 $\boldsymbol{L}_{j-1}$ “整体替换”为 $\boldsymbol{L}_k$ $(0 \le k < j-1)$，但这种替换对破坏匿名性没有任何帮助，也不会影响系统正确性。

因此，为了使 Mix-net 的匿名性遭到破坏的概率小于 e，哑元数 D 必须满足

$$\frac{1}{N(N-1)\text{L}(N-D+1)} \le e$$

例如在 $N = 10^6$ 时，为获得 2^{-80} 的错误概率，D 至少等于 4。

⑤ sid 的不同使得不同会话中的加密密钥 $(\hat{e}(Q, H_1(sid)), \hat{e}(y, H_1(sid)))$也是不同的。因此如果恶意服务器插入与其他会话具有相关性的选择性密文，那么由于协议在不同会话中的外层 ElGamal 解密操作是针对不同的加密密钥的，所以解密结果对攻击者猜测输入与输出的匹配关系没有任何帮助。

4.2　效率分析

1) 参数的选择

主要将所提方案与 Golle 的原方案[8]在计算复杂度方面作一个比较。为了使 2 个方案具有可比性，必须对两者的参数选择作出假设。

Golle 方案所需的群 G 一般取自某个模整数群 Z_p^* 的 q 阶子群。为获得 2^{80} 级别的安全性，p 的长度至少为 1 024bit。本方案所需的双线性对映射 $\hat{e}: G_1 \times G_1 \to G_2$ 假设采用最常见的构造方法：设 E(F_{p_2})表示定义在有限域 F_{p_2} 上的某个椭圆曲线群，取 E(F_{p_2})的一个 q 阶子群作为 G_1。取有限域 $E(F_{p_2^k})$ 的一个 q 阶子群作为 G_2，k 称为嵌入度。$\hat{e}$ 采用 Weil 或 Tate 对[32]。为获得 2^{80} 级别的安全性，一般令 $k = 2$，$p_2 \approx 2^{512}$。因此在同样的安全级别下，2 个方案(分别工作在群 G 和 G_2 上)中的一个 ElGamal 密文对的编码长度大致是相同(2 048bit)。为了使指数运算也具有可比性，令 2 个方案中的 q 都取相同的宽度。在下面运算量的分析中，只考虑了作为主要负载的指数运算量。文献[37]的实验数据表明 2 个群上的指数运算时间也大致是相同的。

2) 客户端的复杂度

所提方案中的用户每发送一个消息都要 9 个指数运算计算密文，用 5 个指数运算计算有效性证明，另外还要用 2 个双线性对运算计算 Q_{sid} 和 Y_{sid}。而 Golle 方案的用户只需用 14 个指数运算。因此新方案客户端的计算复杂度略高于原方案。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。2 个方案的用户发送密文的长度是相同的。

3) 服务器端的复杂度

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的情况下，所提方案的每个服务器在混洗阶段(不包括哑元追踪阶段)需用 $6N+6D$ 个指数运算对密文向量进行再加密，为构造 s_j 需要用 2 个指数运算，验证其他服务器的 s 证明用 $4(n-1)$ 个指数运算。在外层 ElGamal 解密阶段，解密、构造 d_j 和验证其他服务器的证明分别要用 $3N$、2 和 $4(n-1)$ 个指数运算。在哑元追踪阶段，每个服务器验证哑元的正确性用 $6D$ 个指数运算。在内层 ElGamal 解密阶段，解密、构造非交互证明和验证其他服务器的证明分布要用 N、N 和 $2(n-1)N$ 个指数运算。

一般的再加密型 Mix-net 包含 2 轮服务器间的交互过程，1 轮用于逐级地再加密和混洗(包括验证其他服务器的混洗证明)，1 轮用于联合解密。而本协议需要 3 轮交互，增加的 1 轮用于追踪哑元。这是协议的一个缺点。但在该阶段，每个服务器仅需广播 $(3\text{lb}q + 1\text{lb}N)D$ bit 的数据，仅需计算 $6D$ 个指数运算，如果服务器间通过高速网络相连，则对系统输入输出延时的影响不大。

表 1 给出了所提方案与 Golle 方案的单个 Mix 服务器在协议的各个阶段需要花费的指数运算量(在未出现服务器作弊的情况下)。其中都未考虑对用户输入的验证工作量，因为该工作是可以和用户输入过程并行进行的。

表 1 单服务器指数运算量比较

协议	再加密	混洗证明的构造与验证	外层解密	内层解密
Golle 的协议[8]	$6N$	$12n$-6	$6nN$	$2nN$
所提协议	$6N+6D$	$6D+4n$-2	$3N+4n$-2	$2nN$

2 个方案的主要差别是在外层解密阶段。新方案用简单的证明 d_j 和引理 1 保证解密的正确性，因而不需要针对密文向量中每个元素构造和验证非交互式证明，所以节约了大约 $(6n-3)N$ 个指数运算。虽然在混洗阶段的复杂度略高于原方案，但在 N 远大于 n 和 D 的条件下，这些差别可以忽略不计。另外，新方案的服务器还要用 2 个双线性对运算计算 Q_{sid} 和 Y_{sid}，但在整个会话中只需进行一次，因而也可以忽略不计。

除了外层解密阶段，2 个方案的通信复杂度也基本相同。由于在外层解密时不需要针对密文向量中每个元素构造和验证非交互式证明，因此新方案大约节省了 $2nN\,\mathrm{lb}q$ bit 的通信量。

这 2 个方案与其他再加密型 Mix-net 方案的效率比较如表 2 所示。一些被发现存在安全漏洞的 Mix-net 未列入表中。其中效率最高的是 Furukawa 等的方案[39]，每个服务器共需进行 $6nN+5N$ 个指数运算。而本文方案共需进行 $2nN+9N+12D+8n$-4 个指数运算。在 N 远大于 n 和 D 的条件下，$12D+8n$-4 个指数运算量可忽略不计。两者相比较，本方案大约节省了 $4(n-1)N$ 个指数运算。服务器数目越多，则节省的运算量越多。一般的 Mix-net 系统中，服务器数目 n 至少为 3，因此这里至少节省了 $8N$ 个指数运算。如果只有一个 Mix 服务器，本方案的优化设计反而增加了计算复杂度。

表 2 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 Mix 证明与验证	解密
OKST97[15]	$2N$	$642Nn$	$(2+4n)N$
Abe99[16], JJ99[17]	$2N$	$7N\log N(2n-1)$	$(2+4n)N$
FS01[19]	$2N$	$10N(2n-1)$	$(2+4n)N$
Neff01[4]	$2N$	$8N(2n-1)$	$(2+4n)N$
FMOS[38]	$2N$	$(10n-1)N$	N
Groth[21]	$2N$	$(8n-1)N$	N
Furukawa04[39]	$2N$	$(6n+2)N$	N
Golle[8]	$6N$	$12n-6$	$8nN$
所提协议	$6N+6D$	$6D+4n-2$	$(3+2n)N+4n-2$

通过比较可以看出，我们的方案效率高于目前已知的所有基于零知识证明的和基于多参照组对比的再加密型 Mix-net 方案，尤其在消息数目 N 较大并且由多台服务器构成混洗网络的情况下。

5 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 10 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。而本文提出的乐观型 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。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。另外，提出的基于双线性对的乐观型 Mix-net 还具有密钥管理简单，解密速度快等优点。该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, USA, 2004. 303-320.

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

[4] NEFF A. A verifiable secret shuffle and its applicatio 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]. Financia Cryptography '97[C]. Anguilla, UK, 1997. 17-31.

[6] JAKOBSSON M, RAIHI D. Mix-based electronic payments[A]. Proceedings of SAC '98[C]. London, UK, 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] GOLLE P, ZHONG S, BONEH D, *et al*. Optimistic mixing for exit-polls[A]. Cryptology-Asiacrypt '02[C]. Queenstown, New Zealand, 2002. 451-465.

[9] ABE M. Flaws in some robust optimistic Mix-nets[A]. Proceedings of the 8th Australasian Conference on Information Security and Privacy[C]. Wollongong, Australia, 2003. 39-50.

[10] 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.

[11] LI L H, FU S F, CHE X Q. A new relation attack on the imistic Mix-net[A]. International Symposium on Computer Network and Multimedia Technology(CNMT 2009)[C]. Wuhan, China, 2009.1-4.

[12] PFITZMANN A, PFITZMANN B. How to break the direct RSA-implementation of mixes[A]. Cryptology- Eurocrypt '89[C]. Houthalen, Belgium, 1989. 373-381.

[13] PARK C, ITOH K, KUROSAWA K. Efficient anoymous channel and all/nothing election scheme[A]. Cryptology-Eurocrypt'93[C]. Lofthus, Norway, 1994. 248-259.

[14] SAKO K, KILIAN J. Receipt-free mix-type voting scheme[A]. Cryptology-Eurocrypt '95 [C]. Saint-Malo France, 1995. 393-403.

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

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

[17] JAKOBSSON M, JUELS A. Millimix: Mixing in Small Batches[R]. DIMACS Technical Report, 1999. 99-133.

[18] WAKSMAN A. A permutation network[J]. Journal of the Association for Computing Machinery, 1968, 15(1):159-163.

[19] FURUKAWA J, SAKO K. An efficient scheme for proving a shuffle[A]. Cryptology- Crypto'01[C]. Santa Barbara, California, USA, 2001. 368-387.

[20] NEFF A. A verifiable secret shuffle and its applicatio to E-voting[A]. Proceedings of ACM CCS '01[C]. Philadelphia, Pennsylva ia, USA, 2001. 116-125.

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

[22] NGUYEN L, SAFAVI R, KUROSAWA K. Verifiable shuffles: a formal model and a paillier-based efficient construction with provable security[A]. Proceedings of ACNS '04[C]. Yellow Mountain, China, 2004. 61-75.

[23] PENG K, BOYD C, DAWSON E. Simple and efficient shuffli g with provable correctness and ZK privacy[A]. Cryptology- CRYPTO 2005[C]. Santa Barbara, California, USA, 2005. 188-204.

[24] WIKSTROM D. A sender verifiable mix-net and a new proof of a shuffle[A]. Cryptology-Asiacrypt '05[C]. Chennai, India, 2005. 273-292.

[25] PENG K. Failure of a mix network[J]. International Journal of Network Security & Its Applications, 2011, 3(1): 81-97.

[26] JAKOBSSON M. A practical mix[A]. Cryptology-Eurocrypt '98[C]. Espoo, Finland, 1998. 448-461.

[27] JAKOBSSON M. Flash mixing[A]. Proceedings of PODC '99[C]. Atlanta, Georgia, USA, 1999. 83-89.

[28] DESMEDT Y, KUROSAWA K. How to break a practical mix and design a new one [A]. Cryptology-Eurocrypt'00[C]. Bruges, Belgium, 2000. 557-572.

[29] MITOMO M, KUROSAWA K. Attack for flash mix[A]. Proceedings of Asiacrypt 2000[C]. Kyoto Japan, 2000. 192-204.

[30] JAKOBSSON M, JUELS A, RIVEST R. Making Mix-net robust for electronic voting by randomized partial checking[A]. Proceedings of USENIX'02[C]. San Francisco USA, 2002. 339-353.

[31] GOLLE P, BONEH D. Almost entirely correct mixing with lications to voting[A]. Proceedings of ACM CCS'02[C]. Washington DC, USA, 2002. 68-77.

[32] BONEH D, FRANKLIN M. Identity based encryption from the Weil pairing[J]. SIAM J of Computing, 2003, 32(3):586-615.

[33] PEDERSEN P. Non-interactive and information theoretic secure verifiable secret sharing[A]. Cryptology-Crypto'91[C]. Santa Barbara, California, USA, 1991. 129-140.

[34] BELLARE M, ROGAWAY P. Random oracles are practical: a paradigm for designing efficient protocols[A]. Proceedings of ACM CCS' 93[C]. Fairfax, Virginia, USA, 1993. 62-73.

[35] CRAMER R, DAMGAARD I, SCKOENMAKERS B. Proofs of partial knowledge and simplified design of witness hid ng protocols[A]. Cryptology – Crypto '94[C]. Santa Barbara, California, USA, 1994. 174-187.

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

[37] KATE A, ZAVERUCHA G, GOLDBERG I. Pairing based onion outing[A]. The 7th Workshop on Privacy Enhancing Technologies[C]. Ottawa, Canada, 2007. 95-112.

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

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

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。

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。

苏锐丹（1978-），男，河南灵宝人，西安电子科技大学副教授，主要研究方向为计算机与网络安全。