doi:10.3969/j.issn.1000-436x.2014.07.002

基于信誉的快速无可信第三方匿名撤销方案

奚璨[1,2]，冯登国[1]

(1. 中国科学院 软件研究所可信计算与信息保障实验室，北京 100190；2. 中国科学院大学，北京 100049)

摘 要：首先指出 ESORICS 2012 中匿名撤销方案 PE(AR)^2 的安全问题，然后提出一个基于信誉的不依赖可信第三方的快速匿名撤销方案。该方案允许服务提供者赋予匿名会话正分或者负分并封禁信誉过低的用户。实验结果表明，当 K=80 时，本方案可以支持每分钟 820 次的登录请求，而此前最快速的方案 PERM 只能支持每分钟 21 次的登录请求。

关键词：匿名撤销；匿名信誉；无可信第三方；匿名认证

中图分类号：TP309 **文献标识码**：A **文章编号**：1000-436X(2014)07-0010-12

Efficient anonymous reputation-based revocation without TTP

XI Li[1,2], FENG Deng-guo[1]

(1. TCA, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China; 2. University of Chinese Academy Sciences, Beijing 100049, China)

Abstract: The security problems in PE(AR)^2 are pointed out then a new TTP-free scheme is proposed to supports anonymous reputation-based revocation. This scheme allows the SP(service provider) to assign positive or negative scores to anonymous sessions and block users whose reputations are not high enough. The benchmark shows that for proposed scheme the SP can handle 820 authentications per minute while PERM can only handle 21 authentications per minute.

Key words: anonymous revocation; anonymous reputation; TTP-free; anonymous authentication

1 引言

在当前的网络环境中，用户往往需要匿名性来保护自己的隐私。没有匿名性，用户可能不愿意发表和个人行为相关联的信息，如对一个专科医生做出评价。另一方面，匿名性往往会被滥用，如恶意的用户可能会利用匿名网络发布虚假消息。

为了封禁匿名的恶意用户，一个直观的方法就是引入可信第三方。当匿名的恶意行为发生时，服务提供者向可信第三方求助，可信第三方通过将该匿名行为去匿名化或者链接该用户的后续行为来封禁此恶意用户。但是可信第三方的存在意味着用户并不能确保自己行为的匿名性。因此基于可信第三方的匿名撤销方案[1~3]不能满足用户的强隐私需求。

为了解决上述问题，学术界提出了一些不基于可信第三方的匿名撤销方案，例如 BLAC、EPID、PEREA、FAUST、BLACR、PE(AR)、PERM^2[4~11]。这些方案使用零知识证明或者盲签名来代替可信第三方，从而保证了用户的匿名性。

BLAC[4]和 EPID[5]需要用户通过零知识证明其证书和黑名单中的所有票据均不关联。其主要缺点是计算复杂度和黑名单的长度线性相关，因此不适用于那些拥有较大黑名单的服务提供商，如 WikiPedia 和新浪微博。

为了降低匿名撤销方案在服务提供者端的计算复杂度，Tsang 等人提出了基于撤销窗口概念的 PEREA[6]。当用户登录以后，PEREA 要求服务提供者在该用户再登录 K 次以内对该用户的此次登录做出评判，这里 K 是撤销窗口的大小。在服务

收稿日期：2014-06-03；**修回日期**：2014-07-02

基金项目：国家自然科学基金资助项目（61202414, 91118006）；国家重点基础研究发展规划（“973”计划）基金资助项目（2013CB338003）

Foundation Items: The National Natural Science Foundation of China (61202414, 91118006); The National Basic Research Program of China (973 Program) (2013CB338003)

端，PEREA 的计算复杂度和 K 线性相关，并独立于黑名单的长度。如果服务提供者需要一天来评判用户的登录行为，那么撤销窗口的概念要求用户在一天内最多只能登录 K 次，这样才能确保恶意用户能够被封禁。因此 PEREA 需要和另外提供登录限速的方案进行绑定，如文献[7]。

在撤销窗口概念下 FAUST[8]通过使用白名单来降低计算复杂性，但是不能提抗合谋攻击。在 ESORICS 12 中 Yu 等人提出了 PE(AR)^2[9]。PE(AR)^2 通过使用用户的票据乘积来进行非成员证据，从而提高了服务提供者端的运算效率，不过 PE(AR)^2 只能支持极为有限的撤销策略，即只要用户犯一次错就会被封禁。而在现实中，不同恶意行为往往有着不同的严重程度，比如人身攻击严重程度可以是−1，散播谣言的严重程度可能是−5，而服务提供者可能希望封禁恶意行为总分在−10 以下的用户。

信誉系统在当前的网络中得到了广泛的应用。用户可以利用信誉系统来评估网络信息的可信性。如在著名问答网站 StackOverflow 中，提供正确答案的用户可以得到一定的正分，用户的信誉是其过往打分的总和。因此信誉较高的用户提供的信息一般被认为拥有较高的可信度。

信誉系统还可以和激励惩罚机制绑定。如拥有较高信誉的用户可以具有额外的特权，而信誉过低的用户则可能遭到封禁。

考虑到信誉系统的重要性，Au 等人在 NDSS12 中将 BLAC 拓展为 BLACR(BLAC with Reputation)[10]。BLACR 提出了基于信誉系统的匿名撤销。用户的良好行为如提供正确解答可以赢得正分，而恶意行为则会被评为负分。信誉过低的用户会遭到封禁。原始的 BLACR 和 BLAC 一样，其运算量和信誉列表的长度线性相关。为了提高效率，Au 等人提出了快速通道技术，但是 BLACR 依然只能支持数千项的信誉列表。

为了支持大型的信誉列表，Au 等人在 CCS2012 中将 PEREA 拓展为 PERM[11]。通过添加信誉的记忆值，PERM 可以提供基于信誉的匿名撤销。通过引入审查指针，PERM 提高了用户端的效率。在服务提供者端，PERM 的计算效率依然和撤销窗口的大小 K 线性相关，并且比 PEREA 慢。文献[11]表明一个八核的服务器只能支持每分钟 10 次的登录需求。这对于那些重负载的服务提供商如 YouTube 来说是远远不够的。和 PEREA 一样，PERM 也需要和另外的提供登录限速的方案进行绑定。

基于撤销窗口的体制如 PEREA 和 PERM 的不足是要求所有的行为都必须在一个确定的时间窗口中被评估。虽然大部分明显的恶意行为可以在较短时间内被发现，一些有争议的行为可能需要较长的时间来进行评价。由于用户在时间窗口中只能认证 K 次，因此简单的增加评估时间并不是一个令人满意的选择。

PE(AR)^2[9]中也提供了一个信誉系统，但是只能支持正分，因此不能实现基于信誉系统的匿名撤销。值得注意的是在 PE(AR)^2 中发现了严重的安全问题，该问题导致了体制的信誉系统不可用。

注意到目前的基于信誉系统的匿名撤销方案如 BLACR 和 PERM 在服务提供者端都只能支持分分钟数十次的登录需求。这使得这些系统不能适用于重负载的服务提供商如 YouTube 等。而另一方面，一些快速的匿名撤销方案如 FAUST[8]，PE(AR)^2[9]则存在着安全问题，并且撤销功能不够强大，无法提供基于信誉的匿名撤销。针对上述问题，本文提出了一种新型的基于信誉系统的匿名撤销方案 ARBRE(anonymous reputation-basedrevocation)。

本文主要贡献为：指出了 PE(AR)^2 中的安全问题，并提出了一种新的匿名撤销方案 ARBRE。该方案和目前方案比较有如下优势。

1) 服务提供方可以按照不同的类别对匿名会话进行评估并打分。例如 YouTube 可以具有 2 个类别：影音和评论。一个受欢迎的视频在类别影音中可能被评为+5 分，而一个恶意攻击的评论在类别评论中则被评为−2 分。服务提供方可以灵活制定认证策略来封禁恶意用户，如要求登录用户在类别影音中的信誉大于−5 分，并且在类别评论中的信誉大于−4 分。

2) 和 PEREA 和 PERM 不同，ARBRE 并不要求所有的会话都必须在一个确定的时间窗口内被评估。因此服务提供方可以拥有较长的时间对争议会话进行评价，降低了服务提供方的负担，提高了争议会话评估的正确性。

3) 因为用户只能移除已经被评价的会话，ARBRE 不需要和额外的提供登录限速的方案进行绑定。因此在必要时发布一些和隐私相关的信息

时，用户可以快速地匿名登录多次，以保证每次登录的行为不能被链接。

4) 在服务提供者端，ARBRE 的运算量和信誉列表的长度无关，并远高于现有的体制。实验结果表明，当 K=80 时，ARBRE 可以支持每分钟 820 次的登录需求，而 PERM 只能支持每分钟 21 次的登录需求。

5) 在客户端，ARBRE 也具有良好的表现，当恶意行为列表有十万项的时候，客户端的登录时间只需要 1.36 s。

2 预备知识

2.1 零知识证明协议

在零知识证明(zero knowledge proof of knowledge) 协议中，证明者在不泄露额外信息的情况下证明一个命题成立。Σ-协议[12]是一种可以被转化为知识签名的零知识证明协议。本文使用 Camenisch 和 Stadler[13]的表示方式：$PK\{(x): y=g^x\}$代表证明证明方知道一个值 x 满足 $y=g^x$ 的零知识证明协议。$SPK\{(x): y=g^x\}(M)$ 代表对应的知识签名。

2.2 承诺体制

ARBRE 使用的承诺体制由 Fujishaki 和 Okamoto[14]提出，并由 Damgard 和 Fujishaki[15]改进。$\mathbb{C}(\{m_i\}_{i=1}^n)$ 表示关于消息$\{m_i\}_{i=1}^n$的承诺。该承诺体制具有同态性质，即 $\mathbb{C}(\{a_i\}_{i=1}^n)\mathbb{C}(\{b_i\}_{i=1}^n)=\mathbb{C}(\{a_i+b_i\}_{i=1}^n)$。

2.3 动态通用累加器

PEREA、PE(AR)^2 和 ARBRE 使用了动态累加器(DUA, dynamic universal accumulator)[1]。DUA 有一个安全问题[16]，不过可以被改进[9]。本节介绍改进后的正确 DUA，主要关注非成员证据(NMW, non-membership witness)。

DUA Setup：DUASetup 算法的输入为安全参数 K，该算法选择一个 K bit 安全素数乘积 N 和 $g\in_R QR_N$。累加器的公钥 $PK_{acc}=(N,g)$，私钥 $SK_{acc}=\phi(N)$，累加器的初始集合 $acc=\varnothing$，初始票据乘积 $U=1$，初始累加器值 $V=g$。

DUA 票据累加算法：票据累加算法 $\widetilde{V}=accu(V,S_t)$ 可以将一些票据 $S_t=\{t_i\}_{i=1}^n$ 累加到累加器中。算法的输入为累加器值 V 和 S_t，输出新的累加器值 $\widetilde{V}=V^{\prod_{t\in S_t}t}$。

NMW 生成算法(不使用私钥)：给定一个累加器 acc，NMW 生成算法 $w\leftarrow nmw_{gen}(T,acc)$ 为一个票据 T 生成 NMW w。该算法如下。

1) 计算累加器票据乘积：$U=\prod_{t\in acc}t$。

2) 使用欧几里得算法计算 a 和 b，满足 $aU+bT=1$，$0<a<T$。

3) NMW $w=(a,g^{-b} \bmod N)$。

NMW 更新算法：当累加器中新增了票据集合 S_t，NMW 更新算法 $\hat{w}\leftarrow nmw_{update}(T,w,S_t,V)$ 用来为票据 T 更新 NMW $w=(a,g^{-b})$。该算法如下。

1) 计算新增票据乘积 $T_{add}=\prod_{t\in S_t}t$。

2) 使用欧几里得算法计算 a_0、r_0、$a_0T_{add}+r_0T=1$，$0<a_0<T$，计算 $\hat{a}=a_0a \bmod T$。

3) 计算 r 满足 $\hat{a}T_{add}=a+rT$，升级后的 NMW 为 $\hat{w}=(\hat{a},dV^r \bmod N)$。

NMW 知识证明：给定一个元素 $h\in QR_n$，一个票据 T，一个随机值 r 和一个 NMW(a,d)，证明者可以使用零知识证明协议 $PK\{(T,r,a,d): C_1=g^Th^r \wedge V^a=d^Tg\}$ 来证明 $gcd(T,U)=1$。令 $nmw_{verif}(x,w,V)=1/0$ 表示 w 是一个正确/不正确的非成员证据。

注意 NMW 证明了 $gcd(T,U)=1$，因此如果 $t\in S_t$ 均为素数，可以使用票据乘积 $T=\prod_{t_i\in S_t}t_i$ 的 NMW 来证明所有的 $t\in S_t$ 均未被累加。该技术在 PE(AR)^2 中被提出，ARBRE 中也采用了该技术。

2.4 CL 签名

CL(camenisch lysyanskaya)签名[17]具有 2 个有用的协议。

1) 为绑定在承诺中的秘密消息值生成签名。用 $cl_{\text{sign}}(\mathbb{C}(m_1),\cdots,\mathbb{C}(m_n))$ 来表示该协议。该协议正确执行后，用户得到一个关于消息值 $(m_1,\cdots,m_n)$ 的 CL 签名。

2) 通过零知识证明拥有一个关于 m 的合法 CL 签名 σ，m 绑定在承诺 $\mathbb{C}(m)$ 中。用 $PK\{(m,\sigma): C=\mathbb{C}(m) \wedge cl_{verif}(\sigma,m)=1\}$ 来表示该协议，这里 $cl_{\text{verif}}(\sigma,m)=1/0$ 表示 σ 是个正确/不正确的签名。

2.5 基于签名的范围证明

使用 Camenisch 等人提出的基于签名的范围证明[18]。简要的讲，验证者对所有在给定范围内的元素生成签名，证明者通过零知识协议证明他拥有一个关于 x 的签名来证明 x 在给定范围内。

3　PE(AR)^2 的安全问题

在 PE(AR)^2 中，在注册阶段用户首先随机选择 K 个素数 $\{t_1,\cdots,t_K\}$ 作为其票据集合，一般 K=1 000，记 $T=\prod_{i=1}^{K}t_i$ 为此 K 个素数的乘积，由于票据均为素数，为简明起见本文中不会区分票据集合 $\{t_1,\cdots,t_K\}$ 和票据集合中所有票据的乘积 T。在注册阶段，用户可以从服务提供者得到一个关于 T 的 CL 签名，同时不泄露 T。当用户匿名登录时，他从这 K 个素数中选择一个未被使用过的票据 t 作为此次会话的会话 ID，为简明起见下文中称会话 ID 为票据。由于这 K 个素数均为随机选择，并且 T 不会泄露，因此用户不同的登录会话不能被链接，保证了匿名性。服务提供者对匿名会话 t 进行审计并打分，然后生成一个 CL 签名将票据 t 和相应的打分 S 进行绑定。服务提供者同时以累加器的形式维护一个黑名单 BL，如果一个会话被评为恶意行为，则该会话对应的票据会被加入到累加器中。

在 PE(AR)^2 的 auth 协议中，如 2.3 节所述，用户使用其票据集合中所有票据的乘积 T 来生成非成员证据(NMW)，从而使用一次非成员证据来证明所有的票据都不在黑名单中，提高了服务提供者端的验证效率。

PE(AR)^2 也具有一个支持正分的信誉系统，其 Redeem 协议允许用户收回其登录会话得到的正分，即证明一个登录会话的票据 t 确实属于该用户的票据集合 T 并且将对应的打分 S 加入该用户的信誉 R 中。在 Redeem 协议中，当一个票据 t 的打分被用户回收，t 将会被移除出用户的票据集合 T：$T\leftarrow T/t$。Redeem 协议也允许用户生成新的票据集合 T_{new}，并将这些新的票据更新到票据集合 T 中：$T\leftarrow T\cdot T_{\text{new}}$。PE(AR)^2 的细节可以参考文献[9]。遗憾的是发现在 PE(AR)^2 存在着以下安全问题，导致该方案不可用。

3.1　信誉伪造

在 PE(AR)^2 中，用户只可以通过 Redeem 协议来回收其会话的正分。令 T_{old} 为用户要回收会话的票据的乘积，为了满足不可链接性，T_{old} 不能被泄露。因此用户需要通过零知识证明要回收的票据 T_{old} 确实由该用户使用。如文献[9]的 4.2 节中，在 Redeem 协议中用户生成以下零知识证明：$PK_3\{(t_{i+1},T,T_{\text{old}},T_{\text{new}},\sigma,s,s',\mathbb{S}):T_{\text{old}}\,|\,T\wedge s'=\sum_{s_j\in\mathbb{S}}s_j\ \wedge\ 1=Verify_{\text{CL}}((t_j,s_j),\sigma_j,pk_{\text{CL}})\wedge 1=Verify_{\text{CL}}((T,\ t_{i+1},s),\sigma,\ pk_{\text{acc}})\}_{(t_j,s_j,\sigma_j)\in\mathbb{S}}$。该零知识证明协议证明了 $T_{\text{old}}\,|\,T$，即要回收的票据 T_{old} 属于该用户的票据集合 T。遗憾的是，因为证明了 $T_{\text{old}}\,|\,T$ 并不意味着 T_{old} 中的票据确实曾经由该用户使用。

首先，用户 A 和 B 可以在注册(Reg)协议或者赎回协议中进行合谋，使得一个票据 t 既在 A 的票据集合 T_{A} 又在 B 的票据集合 T_{B} 中，即 $t\,|\,T_{\text{A}}$ 并且 $t\,|\,T_{\text{B}}$。注意到这种合谋是容易做到的，因为在 Reg 协议或者 Redeem 协议中，票据集合如 T_{A} 和 T_{B} 不能被泄露以保证不可链接性，无法检测票据集合中是否存在碰撞，即是否有一个票据 t 既在 T_{A} 又在 T_{B} 中。如果用户 A 使用 t 进行了一次良好行为并且得到了正分，那么 A 和 B 均可以回收此票据以得到更高的信誉，虽然事实上 B 并没有使用过票据 t。其次在 Redeem 协议中 A 可以在其生成的新的票据集 T_{new} 中加入 t，从而可以任意次回收赋予 t 的正分以提高自己的信誉。因为新的票据集合 T_{new} 也不能被泄露，所以该攻击也是成立的。

为了抵抗这种信誉伪造攻击，本文指出协议应该满足的 2 个目标。

1) 用户只能回收那些确实是经由自己使用的票据。该性质可以用来抵抗合谋攻击。

2) 每个票据只能被回收一次。该性质保证用户不能多次回收自己使用的票据以提高信誉。

3.2　CL 签名的误用

PE(AR)^2 的主要想法是使用票据集合 $\{t_1,\cdots,t_K\}$ 中所有票据的乘积值 $T=\prod_{i=1}^{K}t_i$ 的非成员证据进行非成员证明，即证明 $gcd(X,T)=1$，这里 X 是黑名单 BL 中的所有票据的乘积：$X=\prod_{t\in BL}t$。由于票据 $\{t_1,\cdots,t_K\}$ 都是素数，通过一次非成员证明即可证明票据集合中所有的票据都不在黑名单 BL 中。PE(AR)^2 需要服务提供者为 T 生成一个 CL 签名，如文献[9]的 4.2 节中，为了防止票据碰撞，每个票据 t_i 均为 $l_t=166$ bit 的素数，当 K=1 000 时，乘积 T 为 166 000 bit。这意味着服务提供者在为 T 生成 CL 签名时，需要生成一个 166 002 bit 的素数(为了保证安全性，CL 签名中的素数 e 需要比被签署的消息大 2 bit[18])，这显然是不实际的(大素数生成极

为耗时)。在 PE(AR)^2 中，这个问题被忽视了，因为在文献[9]的 4.2 节中只要求 CL 签名中素数 e 的长度满足 $l_e > l_t + 2 = 168$，这里 $l_t = 166$ 是单个票据的长度，但是需要签署的消息并不是单个票据。

4 ARBRE 概述

ARBRE 同样利用票据的乘积来生成非成员证据，但使用了和 PE(AR)^2 不同的构造方法。该构造不仅避免了在上一节指出的安全问题，并且显著地增强了功能和提高了效率。PE(AR)^2 只能支持正分，而 ARBRE 可以支持正负分，从而可以实现基于信誉的匿名撤销。而通过缩短票据长度和提出新的非成员证据生成算法以及使用票据池，ARBRE 大幅度提高了用户端和服务提供者端的效率。

在 ARBRE 中，用户向服务提供者注册并获得证书。成功注册以后，用户可以进行匿名登录。每次成功登录后，用户会发送一个素数票据作为本次匿名会话的会话 ID。服务提供者(SP, service provider)对匿名会话按照类别进行打分，如 YouTube 可以维持 2 个类别：影音类别和评论类别。而一个受欢迎的视频打分可以是(+5,0)，这意味着在类别影音中被评为+5 分，而在类别评论中被评为 0 分。令 m 是类别的个数，SP 维护一个信誉列表，信誉列表的每一项包含一个匿名会话的票据 t，相应的评分 $\{S_j\}_{j=1}^m$，这里 S_j 是 t 在类别 j 中的打分，以及一个关于该票据和评分的 CL 签名 σ_t。服务提供者同时以动态累加器的形式维护一个恶意行为名单，当一个会话被评价为恶意行为(即被打负分)，该会话的票据会被加入该累加器。SP 制定形式为 $\vee_{k=1}^{J}(\wedge_{j=1}^{m} P_{kj})$ 的登录策略 Pol，即一些子句 P_{kj} 的合取析取范式，子句 P_{kj} 为不小于 η_{kj} 的形式，这意味着在第 j 个类别中，用户的信誉应该不小于 η_{kj}。登录的用户必须满足登录策略 Pol。

在展示 ARBRE 的构造细节前，首先介绍 ARBRE 构造中使用的主要技巧，主要目的是为了让读者从直观上理解 ARBRE。

4.1 票据集合

在 PE(AR)^2 中，用户的票据集合 T 在注册阶段时预先生成，并只能通过 Redeem 协议进行更新。在上一节展示了这样的构造不能检查票据碰撞，因而会导致信誉伪造攻击。在 ARBRE 中，用户的票据集合 T_U 会随着用户的登录进行更新：票据集合 T_U 由该用户使用过但尚未回收评分的票据构成。注册时用户的票据集合 T_U 并不包含预先生成的有效票据。用户每次登录后，服务提供者会检查本次登录使用的票据 t 的新鲜性，然后 t 会被更新到用户的票据集合中：$T_U \leftarrow tT_U$。用户使用赎回协议来回收属于自己的票据的评分，令 t_r 为用户需要赎回的票据，用户需要证明 t_r 属于 T_U，即 $t_r | T_U$。当 t_r 的评分被回收后，t_r 会被移除出票据集合：$T_U \leftarrow T_U / t_r$。

和 PE(AR)^2 中预先存储的不能泄露的票据集合 T 不同，在 ARBRE 中只有用户使用过的票据 t 才能被添加到该用户的票据集合 T_U 中。注意到服务提供者会检查每次会话使用的票据 t 的新鲜性，以保证 t 未被其他用户使用过。因此不同用户的票据集合之间不会发生碰撞。而每个评分被回收的票据都会被移除出票据集合，从而保证每个票据只能被回收一次。因此本文构造满足 3.1 节提出的 2 个目标。

PEREA[6]和 PERM[9]中使用的票据集合是一个票据队列 $\{t_1,\cdots,t_K\}$，由于票据队列有着先进先出的性质，即使用的票据必须首先被移除出队列，PEREA 和 PERM 要求所有的会话都必须在一个确定的时间窗口内被评估(注意到票据在被移除之前必须要被评估)。而 ARBRE 中使用的是这些票据的乘积，避免了先进先出的性质，因此那些较难评估的票据可以保留更长时间。ARBRE 允许 SP 使用较长的时间对争议会话进行评价，提高了评价的正确性，降低了 SP 的评价压力。

4.2 票据长度

PEREA[6]和 PE(AR)^2[9]中随机生成的素数票据的长度要足够长以防止碰撞，为达到 2^{-112} 的碰撞概率，票据至少是 230 bit。对于 PEREA 和 PE(AR)^2 来说，防止票据之间的碰撞是必须的，因为票据会被用来证明证书的新鲜性。在 ARBRE 中，证书的新鲜性是由绑定在证书中的随机数 q 证明。当新鲜性被证明以后，用户可以选择一个较短的票据作为会话 ID，如果该票据已经被其他用户使用(以很小的概率发生，如 2^{-20})，则用户可以重新选择一个票据。不让服务提供者分配票据的原因是用户事先维护一个票据池，票据池中包含该用户在一个时间段内将会使用的票据。票据池的概念会在下文进行解释。

ARBRE 使用的素数票据的长度 l_t 比 PEREA 和 PE(AR)^2 要短，l_t=56 就足够，因为长度小于 56 bit 的素数至少有 2^{50} 个[19]。如果整个系统使用过的票据有 2^{30}，则随机选择的长度小于 56 bit 的素数票据

被使用过的概率只有 2^{-20}。

虽然 ARBRE 中的票据长度 l_t 较小，但是当票据集合中票据的个数 K 较大时，票据乘积依然会比较大。如 3.2 节所示，这会导致 CL 签名中的素数 e 过大。为了支持较大的 K，本文的另一个技术就是将票据乘积 T_U 分割为 n 个 l_m bit 的消息块 $\{T_i\}_{i=0}^{n-1}$，这些消息块满足：$T_U=\sum_{i=0}^{n-1}2^{i\cdot l_m}T_i$，一般 l_m=224 即可。

本文的改进有 2 个优势：首先更小的消息块长度 l_m 意味着 CL 签名中的素数 e 的长度 l_e 也比较小，因为 $l_e>l_m+2$。由于生成大的素数是比较耗时，本文的改进大大提高了服务提供者端的效率。其次，在用户端，主要耗时的操作是非成员证据的升级，其计算复杂度为大小为 $\Delta_{acc}\cdot l_t$ 的指数运算，这里 Δ_{acc} 是新被加入到累加器的票据的个数，l_t 是票据的长度。因此缩短票据的长度可以显著地提高用户端的运算效率，当 K=40 时，ARBRE 的非成员证据升级比 PEREA 要快 160 倍(PEREA 需要升级 K 个非成员证据，而每个非成员证据的升级时间是 ARBRE 的 4 倍，因为 ARBRE 票据的长度只有 PEREA 的 1/4)。

4.3　正分和负分

为了支持基于信誉的匿名封禁，ABBRE 需要允许服务提供商给匿名会话打负分，而 PE(AR)^2 只能支持正分。注意到用户不会主动地回收负分，因此支持负分的难点在于强制用户回收所有属于他的负分。这也是如 EPID[5]、BLAC[4]、BLACR[9]等体制效率不高的原因：用户不仅要收回恶意行为名单中属于自己的项，还需要证明恶意行为名单中剩下的所有其他项都不属于自己。

在 ARBRE 中，用户的票据集合为 T_U，首先通过零知识证明回收恶意行为名单中属于自己的票据 $\{t_i\}_{i=1}^{n}$，令 $T_r=\prod_{i=1}^{n}t_i$。然后使用剩余票据乘积 T_U/T_r 的非成员证据来证明所有剩余的票据都不在恶意行为名单中，从而证明自己已经回收了所有的负分项。

4.4　票据池和非成员证据的快速算法

非成员证据 NMW 在服务提供者端的验证效率较高并且和恶意行为名单的长度无关。但是在客户端由于没有累加器私钥，非成员证据的生成是相当耗时的，特别是当恶意行为名单较长时，因为其需要和名单长度成正比的指数运算。

PE(AR)^2 的票据集合在登录时保持不变，只能通过 Redeem 协议进行更新，而 ARBRE 为了避免安全问题和支持负分，使用了动态的票据集合 T_U。动态的票据集合带来了 2 个新的问题。第一，在回收票据 T_r 之前，用户拥有票据集合 T_U 对应于旧恶意行为名单 BL 的非成员证据 w_U。当用户回收票据 T_r 后，需要为新的票据集合 T_U/T_r 计算对应于新恶意行为名单 BL' 的非成员证据 w。第二，在每次登录后，用户的票据集都会更新：$T_U\leftarrow tT_U$，这里 t 是本次登录使用的票据。每次登录后用户都需要为新的票据集 tT_U 计算非成员证据。因为用户没有累加器私钥，当恶意行为名单较长时，直接计算该非成员证据会非常耗时。

为了解决这 2 个新问题，本文提出了一种新的计算非成员证据的算法。该算法的输入为 2 个数 T_1,T_2，$T_1\,|\,T_2$，以及 T_2 对应于累加器 BL 的非成员证据 w_2，输出为 T_1 对应该累加器 BL 的非成员证据 w_1。该算法表示为 $w_1\leftarrow nmw_{\text{pool}}(T_1,T_2,w_2,BL)$。有了该算法后，对于第一个问题，用户首先计算 T_U/T_r 对应于旧恶意行为名单 BL 的非成员证据：$w'=nmw_{\text{pool}}(T_U/T_r,T_U,w_U,BL)$，然后使用 BL'/BL 将 w' 升级为 w。

对于第二个问题，在一个较长的时间段内，如一年，用户生成一个票据池 P_{tk}，其中包含了 n_P 个票据，一般 n_P=3 000，活跃的用户可以生成更多的票据。令 $T_P=\prod_{t\in P_{tk}}t$，在该时间段内，用户登录使用的票据将会从该票据池中取出，因此用户的票据集合满足 $T_U\,|\,T_P$。用户维护票据池的非成员证据 w_P (在下一节可以看到，维护票据池的非成员证据和维护单个票据的非成员证据的效率基本一致)。于是在每一次登录后，用户可以根据 T_P 的非成员证据 w_P 快速地计算出票据集合 T_U 的非成员证据 w_U。

5　ARBRE 构造

本节首先提出新的算法 $nmw_{\text{pool}}(T_1,T_2,w_2,acc)$ 并展示该算法如何降低用户端的运算量，然后介绍 ARBRE 的具体构造细节。

5.1　算法 $w_1\leftarrow nmw_{\text{pool}}(T_1,T_2,w_2,V)$

给定 2 个数 T_1,T_2，$T_1\,|\,T_2$，令 $\tilde{T}=T_2/T_1$。acc 是一个累加器，$U=\prod_{t\in acc}t$ 是累加器中所有票据的乘积，

累加器值 $V=g^U$，这里 g 是累加器所在群中的一个生成元。T_2 对应该累加器的非成员证据是 $w_2=(a,d)$，$a<T_2$。因此存在 b 满足 $aU+bT_2=1$ 和 $g^{-b}=d$。

本文希望计算 T_1 的非成员证据 w_1。直观上，需要计算 $(a',g^{-b'})$，其中 $a'U+b'T_1=1$，$0<a'<T_1$。因为 $aU+b\tilde{T}T_1=1$，计算 $a'=a \bmod T_1$，找到 r，满足 $a=a'+rT_1$。注意到 $r<\tilde{T}$，因为 $a<T_2$，因此 $b'=b\tilde{T}+rU$，$d'=g^{-b'}=g^{-rU}d^{\tilde{T}}=V^{-r}d^{\tilde{T}}$。$T_1$ 的非成员证据 w_1 是 (a',d')。从 w_2 计算出 w_1 需要一个大小为 r 的指数运算以及一个大小为 $\tilde{T}$ 的指数运算，$r<\tilde{T}<T_2$。

现在展示算法 $w_1\leftarrow nmw_{\text{pool}}(T_1,T_2,w_2,V)$ 如何显著地降低用户的计算量。用户生成一个票据池 P_{tk}，池中包含了 n_P 个随机素数票据。恶意行为列表 BL 对应的累加器值为 V_{BL}。令 $T_P=\prod_{t\in P_{tk}}t$，用户维护着票据池对应 BL 的非成员证据 w_P。因为用户登录使用的票据均从票据池 P_{tk} 中取出，每次登录后新的票据集合 T_U 满足 $T_U\mid T_P$，T_U 的非成员证据 w_U 可以有效地由 T_P 的非成员证据 w_P 生成：$w_U\leftarrow nmw_{\text{pool}}(T_U,T_P,w_P,V_{BL})$。

令 l_t 为票据的长度，Δ_{BL} 是累加器每天新增的票据的个数，N_{BL} 是累加器中所有票据的个数。每天更新票据池 T_P 的非成员证据需要大小为 $\Delta_{BL}l_t$ 的指数运算，从 T_P 的非成员证据 w_P 计算出票据集合的非成员证据 w_U 需要大小为 $2n_Pl_t$ 的指数运算。而无累加器私钥时，直接计算票据集合的非成员证据 w_U（不使用算法 nmw_{pool}）需要大小为 $N_{BL}l_t$ 的指数运算。当 $n_P=3\,000$，$\Delta_{BL}=3\,000$，$l_t=56$ 时，本文的实验结果表明每天升级 w_P 并从 w_P 计算出 w_U 需要 4.5 s。作为对比，当每天新增票据个数 Δ_{BL} 为 3 000 时，可以假设累加器里的票据总数 N_{BL} 约为 1 000 000 个，直接计算非成员证据 w_U 需要约 700 s。

除了应用在 ARBRE 中，算法 nmw_{pool} 在需要不使用私钥计算多个票据的非成员证据的时候也非常有效。令 N 是累加器中票据的个数，当需要为 $k\approx N^{1/2}$ 个票据 $\{t_i\}_{i=1}^k$ 分别计算非成员证据(不使用私钥)时，首先计算 $T=\prod_{i=1}^k t_i$，为 T 计算非成员证据需要 Nl_t 大小的指数运算，从 T 的非成员证据使用 nmw_{pool} 计算出 t_i 的非成员证据需要 kl_t 大小的指数运算。因此，为每个 t_i 计算非成员证据平均需要 $1/kNl_t+kl_t\approx 2N^{1/2}l_t$ 大小的指数运算。而直接为 t_i 计算非成员证据需要 Nl_t 大小的指数运算。当 $N=1\,000\,000$ 时，使用 nmw_{pool} 可以提高 500 倍的效率。

5.2 具体构造

本节介绍 ARBRE 的具体构造。

1) Setup

系统参数为 $param=(l_t,K,l_r,l_m,l_n,l,l_s,l_e,l_\phi,l_H)$，这里 l_t (56)是票据长度，K 是票据集合中票据个数的最大值，l_r (224)是用来证明证书新鲜性的随机数的长度，根据生日悖论 l_r=224 可以保证随机数发生碰撞的概率为 2^{-112}。l_m (224)是 CL 签名中消息长度(票据集合中所有票据的乘积为 Kl_t，将其分为 $K/4$ 个长度为 l_m 的消息块以减少 CL 签名中 e 的长度)，l_n 是 RSA 模的大小，l 是 CL 签名中安全参数的大小，l_s 和 l_e 是 CL 签名中元素 s 和 e 的大小，为了 CL 签名的安全性，要求 $l_s>l_n+l_m+l$，$l_e>l_m+2$。l_ϕ 是控制统计零知识证明属性的参数，l_H 是杂凑函数的输出长度。

给定参数 $param$，服务提供者 SP 生成 l_n bit 的安全素数乘积 $N=pq$，p 和 q 是随机安全素数，私钥是 $\phi(N)$。SP 生成 $g_{acc}\in_R \mathcal{QR}_N$ 作为累加器中使用群的生成元。SP 选择一个 l_t bit 的随机素数 $\hat{t}$ 用来填充用户注册时的票据集合。令 m 是分类的个数，J 是子策略的个数，SP 选择一个形式为 $\vee_{k=1}^{J}(\wedge_{j=1}^{m}P_{kj})$ 的认证策略 Pol。

SP 将信誉列表 RL 和恶意行为名单 BL 初始化为空集，相应累加器的值 V 初始化为 g_{acc}，公共参数为 $(N,g_{acc},\hat{t},Pol,RL,BL,V)$，私钥为 $\phi(N)$。

2) Registration

① 用户 Alice 生成一个包含 $\hat{t}$ 和 $\tilde{n}$ 个随机选择的随机素数的票据池 P_{tk}，一般 $\tilde{n}=3\,000$。令 $T_P=\prod_{t\in P_{tk}}t$。Alice 选择 l_r bit 随机数 q。令 $Q=(\{T_i\}_{i=0}^{K/4-1},q,\{R_j=0\}_{j=1}^{m})$，这里 $\sum_{i=0}^{K/4-1}2^{i\cdot l_m}T_i=\hat{t}$（$T_U=\hat{t}$ 是用户的初始票据集合，将长度 Kl_t 的票据集合 T_U 分割或 $K/4$ 个长度为 l_m 的消息块)，$\{R_j\}_{j=1}^{m}$ 代表 m 个类别中的初始信誉值。Alice 生成关于 Q 的承诺 $C_Q=\mathcal{C}(Q)$，将该承诺和零知识证明 $PK\{(q):C_Q=$

$\mathbb{C}(\{T_i\}_{i=0}^{K/4-1}, q, \{0\}_m)\}$ 发送给 SP。

② 如果该零知识证明正确，SP 和 Alice 运行协议 cl_{sign}，cl_{sign} 协议运行成功后，Alice 得到关于 Q 的 CL 签名 σ。SP 同时会将当前的恶意行为名单 BL 和相应的累加器值 V 发送给 Alice。

③ Alice 计算 T_P 的非成员证据 $w_P = nmw_{\text{gen}}(T_P, BL)$，票据集合的非成员证据 $w_U = nmw_{\text{pool}}(\hat{t}, T_P, w_P, V)$。

Alice 保存 $(T_U = \hat{t}, T_P, \sigma, w_U, w_P, q, \{R_j\}_{j=1}^m, BL, V)$。

3) Redeem

① Alice 拥有 $(T_U, T_P, \sigma, w_U, w_P, q, \{R_j\}_{j=1}^m, BL, V)$，她下载最新的信誉列表 $\overline{RL}$ 恶意行为列表 $\overline{BL}$ 和相应的累加器值 $\overline{V}$，令 $\Delta_{BL} = \overline{BL} \setminus BL$，即为恶意列表中新增加的项。如果其票据集合中票据 t 被打分并被加入信誉列表 $\overline{RL}$，则 Alice 可以使用 Redeem 协议来回收赋予该票据的分数 $\{S_j\}_{j=1}^m$。该票据在信誉列表 $\overline{RL}$ 中对应的项为 $(t, \{S_j\}_{j=1}^m, \sigma_t)$，其中，$\sigma_t$ 是关于 $t, \{S_j\}_{j=1}^m$ 的 CL 签名。

Alice 计算承诺 $C_R = \mathbb{C}(\{R_j\}_{j=1}^m)$，$C_S = \mathbb{C}(\{S_j\}_{j=1}^m)$，$C_{U'} = \mathbb{C}(T_U / t)$，将承诺随机数 q 及以下零知识证明发送给 SP。

$$\Pi = SPK\left\{\begin{array}{l}(\{t, \sigma, \sigma_t, T_U, \{T_i\}_{i=0}^{n-1}, \{R_j\}_{j=1}^m, \{S_j\}_{j=1}^m):\\ cl_{\text{verif}}(\sigma, \{T_i\}_{i=0}^{n-1}, q, \{R_j\}_{j=1}^m) = 1 \wedge\\ T_U = \sum_{i=0}^{n-1} 2^{i \cdot l_m} T_i \wedge C_R = \mathbb{C}(\{R_j\}_{j=1}^m) \wedge\\ cl_{\text{verif}}(\sigma_t, t, \{S_j\}_{j=1}^m) = 1 \wedge\\ C_S = \mathbb{C}(\{S_j\}_{j=1}^m) \wedge C_{U'} = \mathbb{C}(T_U / t)\end{array}\right\}(M)$$

该零知识协议证明：当前票据集合是 T_U，当前信誉是 $\{R_j\}_{j=1}^m$，C_R 是关于信誉 $\{R_j\}_{j=1}^m$ 的承诺，票据 t 的分数是 $\{S_j\}_{j=1}^m$，C_S 是关于 $\{S_j\}_{j=1}^m$ 的承诺，票据 t 属于票据集合 T_U，即 $t \mid T_U$，并且 $C_{U'}$ 是关于 $T_U' = T_U / t$ 的承诺。

② SP 验证随机数 q 的新鲜性和零知识证明的正确性。SP 计算 $C_R' = C_S C_R$，根据承诺的同态性质，SP 确信 C_R' 是关于 $\{R_j + S_j\}_{j=1}^m$ 的承诺。

③ Alice 生成一个新的随机数 q' 和相应的承诺 $C_{q'} = \mathbb{C}(q')$，她将 T_U / t 分为 $\{T_i'\}_{i=0}^{n-1}$，生成零知识证明

$$PK\{(\{T_i'\}_{i=0}^{n-1}, T_U / t) : T_U / t = \sum_{i=0}^{n-1} 2^{i \cdot l_m} T_i' \wedge C_T = \mathbb{C}(\{T_i'\}_{i=0}^{n-1}\}$$

和承诺 $C_T = \mathbb{C}(\{T_i'\}_{i=0}^{n-1})$，并将零知识证明和承诺发送给 SP。

④ SP 验证零知识证明的正确性，和 Alice 运行协议 cl_{sign}。Alice 得到关于 $Q' = (\{T_i'\}_{i=0}^{K/4-1}, q', \{R_j + S_j\}_{j=1}^m)$ 的 CL 签名 σ'。

Alice 从票据池中删除在恶意行为列表 $\overline{BL}$ 中的票据 T_B：$T_P' \leftarrow T_P / T_B$，并计算 $T_P' = T_P / T_B$ 对应 $\overline{BL}$ 的非成员证据 w_P'：$\overline{w}_P = nmw_{\text{pool}}(T_P', T_P, w_P, V)$，$w_P' = nmw_{\text{update}}(T_P', \overline{w}_P, \Delta_{BL}, V)$。

若 Alice 已经回收了所有负分票据，她计算 $T_U' = T_U / t$ 对应 $\overline{BL}$ 的非成员证据 $w_U' = nmw_{\text{pool}}(T_U', T_P', w_P', \overline{V})$。

Alice 保存 $(T_U', T_P', \sigma', w_U', w_P', q', \{R_j + S_j\}_{j=1}^m, \overline{BL}, \overline{V})$。

4) Authentication

Alice 从 SP 那里得到最新的信誉列表 $\overline{RL}$，恶意行为列表 $\overline{BL}$ 以及登录策略 Pol，在 Alice 试图登录之前，她需要使用上面描述的 Redeem 协议来回收所有被加入到恶意行为列表 $\overline{BL}$ 中的票据。

假设 Alice 已经使用 Redeem 协议回收了自己被打分的票据，并拥有 $(T_U, T_P, \sigma, w_U, w_P, q, \{R_j\}_{j=1}^m, \overline{BL}, \overline{V})$，Alice 检查其信誉 $\{R_j\}_{j=1}^m$ 是否符合登录策略 Pol，若不符合则放弃登录，否则 Alice 使用如下步骤登录。

① 证明拥有一个合法的证书 σ。

Alice 生成承诺 $C_R = \mathbb{C}(\{R_j\}_{j=1}^m)$，$C_U = \mathbb{C}(T_U)$ 和

$$\Pi_1 : \Pi_1 = SPK\left\{\begin{array}{l}(T_U, \sigma, \{T_i\}_{i=0}^{n-1}, \{R_j\}_{j=1}^m):\\ cl_{verif}(\sigma, \{T_i\}_{i=0}^{n-1}, q, \{R_j\}_{j=1}^m) = 1 \wedge\\ T_U = \sum_{i=0}^{n-1} 2^{i \cdot l_m} T_i \wedge C_U = \mathbb{C}(T_U) \wedge\\ C_R = \mathbb{C}(\{R_j\}_{j=1}^m)\end{array}\right\}(M)$$

Alice 将承诺，证明 Π_1 和随机数 q 发送给 SP。SP 检查 Π_1 的正确性和 q 的新鲜性。

② 使用非成员证据证明票据集合中的所有票据都不在恶意行为列表中。

Alice 将以下零知识证明 Π_2 发送给 SP：

$$PK\{(T_U, w_U) : C_T = \mathbb{C}(T_U) \wedge 1 = nmw_{\text{verif}}(T_U, w_U, \overline{V})\}$$

Π_2 证明 T_U 中所有的票据都不在恶意行为列表中。

③ 证明满足登录策略。

Alice 生成以下零知识证明 Π_3：

$$PK\{(\{R_j\}_{j=1}^m): \vee_{k=1}^J (C_R = \mathbb{C}(\{R_j\}_{j=1}^m) \wedge_{j=1}^m (R_j \geqslant \eta_{kj}))\}$$

Π_3 证明信誉 $\{R_j\}_{j=1}^m$ 满足登录策略。

④ 提交票据作为登录会话 ID。

Alice 从票据池 T_P 随机选择一个素数 t 作为本次登录的票据，如果 t 已经被其他用户使用过，则从票据池中删除 t：$T_P' \leftarrow T_P / t$，并从中选择另一素数作为票据。

⑤ 获得新的证书并为新的票据集合生成非成员证据。

Alice 生成一个新的随机数和相应的承诺 $C_{q'} = \mathbb{C}(q')$，她将 $T_U' = tT_U$ 分为 $\{T_i'\}_{i=0}^{n-1}$，生成零知识证明 $PK\{(\{T_i'\}_{i=0}^{n-1}, tT_U): tT_U = \sum_{i=0}^{n-1} 2^{i \cdot l_m} T_i' \wedge C_T = \mathbb{C}(\{T_i'\}_{i=0}^{n-1}\}$ 和承诺 $C_T = \mathbb{C}(\{T_i'\}_{i=0}^{n-1})$，将零知识证明和承诺发送给 SP。

SP 验证零知识证明的正确性，和 Alice 运行协议 cl_{sign}，从而 Alice 得到关于 $Q' = (\{T_i'\}_{i=0}^{K/4-1}, q', \{R_j\}_{j=1}^m)$ 的 CL 签名 σ'。

⑥ 更新非成员证据。

登录成功以后，Alice 从票据池中删除在恶意行为列表 $\overline{BL}$ 中的票据 T_B：$T_P' \leftarrow T_P / T_B$（$T_B$ 可能是由其他用户使用，若票据池中的票据均不在 $\overline{BL}$ 中，则 T_B =1），并计算 T_P / T_B 对应 $\overline{BL}$ 的非成员证据 w_P'：$\overline{w}_P = nmw_{\text{pool}}(T_P', T_P, w_P, V)$，$w_P' = nmw_{\text{update}}(T_P', \overline{w}_P, \Delta_{BL}, V)$。

然后计算 $T_U' = tT_U$ 对应 $\overline{BL}$ 的非成员证据 $w_U' = nmw_{\text{pool}}(T_U', T_P', w_P', \overline{V})$。

Alice 保存 $(T_U', T_P', \sigma', w_U', w_P', q', \{R_j\}_{j=1}^m, \overline{BL}, \overline{V})$。

5) List Management

对于每个匿名会话 t，SP 对其进行评价并打分，令 $\{S_j\}_{j=1}^m$ 为相应的分数，SP 生成关于 $t, \{S_j\}_{j=1}^m$ 的 CL 签名 σ_t。若 t 是恶意行为，即被判为负分，则 SP 亦将 t 加入恶意行为名单 BL，并更新相应的累加值 $V \leftarrow V'$。

6 ARBRE 安全性分析

6.1 安全属性

1) 认证性

只有满足认证策略的合法注册用户才能够登录成功。

2) 匿名性

服务提供者只能确定登录的用户满足认证策略，除此之外不能得到有关用户的任何其他信息。

3) 不可陷害性

如果服务提供者是诚实的，那么任何其他第三方都不能陷害一个诚实用户，使得该满足认证策略的诚实用户不能登录。

6.2 形式化定义和证明

和 PERM、PEREA 一致，本文使用真实世界/理想世界模型来刻画和证明 ARBRE 的安全性。在真实世界中，参与方之间运行着密码协议，而在理想世界中存在着相同的参与方，不同的是参与方将其输入发送给一个理想可信方 T，并从该理想可信方得到输出。该理想可信方 T 实现了 ARBRE 所应该实现的功能。注意到和理想可信方的通信并不是匿名的。在 2 个世界中，不诚实的参与方由敌手 $\mathcal{A}$ 控制。环境 $\mathcal{E}$ 向参与方提供输入并和敌手任意的交互，称 ARBRE 是安全的，如果对于任意的真实环境中的敌手 $\mathcal{A}$ 和环境 $\mathcal{E}$，在理想世界中存在一个模拟器 $\mathcal{S}$，该模拟器控制着参与方和敌手 $\mathcal{A}$ 控制的一致，使得环境 $\mathcal{E}$ 不能区分他是在真实环境中和敌手 $\mathcal{A}$ 交互还是在理想世界中和模拟器 $\mathcal{S}$ 交互。使用静态模型，即敌手控制的参与方在系统设置之前已经确定。ARBRE 支持以下功能。

1) Setup

当诚实和非诚实的参与方确定以后，系统启动。

真实世界：服务提供者 SP 运行真实 Setup 协议，生成公开系统参数私钥。所有参与方都可以得到公开系统。

理想世界：理想可信方初始化一个集合 *RS*，该集合用来存储用户的注册和登录信息。

2) Registration

环境要求用户 i 向 SP 进行注册。

真实世界：用户 i 向 SP 发送注册请求，和 SP 运行真实 Registration 协议，并独立地将协议运行结果返回给环境。如果 SP 是诚实的，他将会拒绝一个已注册用户的注册请求。

理想世界：用户 i 向 T 发送注册请求，T 告知 SP 用户 i 需要注册以及 i 是否曾注册过。SP 向 T 返回接受/拒绝，T 将该回复返回给用户 i。如果这是用户 i 的第一次成功注册，T 将注册信息储存在

RS 中。用户 *i* 和 SP 独立地将结果返回给环境。

3) Redeem

环境要求用户 *i* 回收票据。

真实世界：用户 *i* 向 SP 发送 Redeem 请求，和 SP 运行真实 Redeem 协议，并独立地将协议运行结果返回给环境。

理想世界：用户 *i* 向 *T* 发送 Redeem 请求。*T* 告知 SP 一个匿名用户需要回收票据的打分。SP 返回最新的信誉名单 RL 和恶意行为名单，*i* 检查其是否可以回收票据，并选择继续或者放弃。如果用户 *i* 选择继续，他选择想回收的票据并发送给 *T*，*T* 检查票据是否属于用户 *i* 以及是否在 RL 中，然后告知 SP 该匿名用户是否可以回收票据。SP 返回接受或者拒绝。如果 SP 接受，SP 将 *RS* 存储的用户信息进行更新。用户 *i* 和 SP 独立地将结果返回给环境。

4) Authentication

环境要求用户 *i* 向 SP 进行登录认证。

真实世界：用户 *i* 向 SP 发送 Authentication 请求，和 SP 运行真实 Authentication 协议，并独立地将协议运行结果返回给环境。

理想世界：用户 *i* 向 *T* 发送 Authentication 请求。*T* 告知 SP 一个匿名用户需要登录。SP 返回最新的信誉名单 RL 和恶意行为名单 BL 以及登录策略 pol，*i* 检查其是否满足登录策略，并选择继续或者放弃。如果用户 *i* 选择继续，*T* 检查用户 *i* 是否已经回收他的所有负分票据以及其是否满足登录策略，然后告知 SP 该匿名用户是否可以登录。SP 返回接受或者拒绝。如果 SP 接受，用户 *i* 选择一个新的票据 *t* 作为本次登录会话的 ID。*T* 将 *RS* 存储的用户信息进行更新。用户 *i* 和 SP 独立地将结果返回给环境。

理想世界中的 ARBRE 实现了想要的功能以及需要的安全属性。在 Redeem 协议中，理想可信方只告诉 SP 一个匿名用户希望回收票据，在 Authentication 协议中，*T* 只告诉 SP 一个匿名用户希望登录以及该用户是否满足登录策略。因此满足匿名性。在 Redeem 协议中，理想可信方检查用户是否可以回收该票据，在 Authentication 协议中，理想可信方检查用户是否满足登录策略，因此满足认证性和抗胁迫性。下面给出 ARBRE 安全的形式化定义。

定义 1　令 $Real_{\mathcal{E},\mathcal{A}}(\lambda)$（相应地，$Ideal_{\mathcal{E},\mathcal{S}}(\lambda)$）为环境 $\mathcal{E}$ 在真实世界（相应地，理想世界）和真实敌手 $\mathcal{A}$（相应地，模拟器 $\mathcal{S}$）交互后输出 1 的概率。称 ARBRE 是安全的。如果对于任意的多项式有界算法 $\mathcal{E}$ 和 $\mathcal{A}$，都有 $|Real_{\mathcal{E},\mathcal{A}}(\lambda)-Ideal_{\mathcal{E},\mathcal{S}}(\lambda)|=negl(\lambda)$。

证明　构造一个模拟器使得环境 $\mathcal{E}$ 不能区分理想世界中的 $\mathcal{S}$ 和真实世界中的 $\mathcal{A}$。分 2 种情况进行讨论：第一种情况下 SP 是诚实的，这种情况用来刻画认证性和不可陷害性；第二种情况下 SP 被敌手控制，这种情况用来刻画匿名性。

第一种情况：*SP* 是诚实的。

1) Setup。$\mathcal{S}$ 代表一个诚实的 SP 和敌手 $\mathcal{A}$ 交互。$\mathcal{S}$ 生成公开参数和私钥，将公开参数发送给敌手 $\mathcal{A}$。

2) Registration。$\mathcal{S}$ 代表非诚实的用户 *i* 和 *T* 交互/代表诚实 SP 和敌手 $\mathcal{A}$ 交互。$\mathcal{A}$ 作为非诚实的用户 *i* 向 $\mathcal{S}$ 进行注册，$\mathcal{S}$ 模拟诚实 SP。$\mathcal{S}$ 使用 rewinding 技术，利用零知识证明协议的知识抽取器抽取出注册时使用的随机数 q，$\mathcal{S}$ 将 (i, q) 存储到集合 U_q 中，该随机数 q 将会用在登录时候确定登录用户。

3) Redeem。$\mathcal{S}$ 代表非诚实的用户 *i* 和 *T* 交互/代表诚实 SP 和敌手 $\mathcal{A}$ 交互。$\mathcal{A}$ 作为非诚实的用户向 $\mathcal{S}$ 进行票据回收，$\mathcal{S}$ 模拟诚实 SP。注意到 $\mathcal{A}$ 可以使用任意的非诚实敌手的证书，因此 $\mathcal{S}$ 需要首先使用利用零知识证明协议的知识抽取器来抽取出 $\mathcal{A}$ 使用的随机数 q，然后使用 U_q 中存储的 (i, q) 项来确定执行 Redeem 协议用户的具体身份 *i*。随后 $\mathcal{S}$ 表非诚实的用户 *i* 和 *T* 交互。当 Redeem 协议执行成功后，*i* 会得到关于新的随机数 q' 的证书，$\mathcal{S}$ 抽取出 q'，将 U_q 中的项 (i, q) 更新为 (i, q')。

4) Authentication。$\mathcal{S}$ 代表非诚实的用户 *i* 和 *T* 交互/代表诚实 SP 和敌手 $\mathcal{A}$ 交互。$\mathcal{A}$ 作为非诚实的用户向 $\mathcal{S}$ 进行登录，$\mathcal{S}$ 模拟诚实 SP。注意到 $\mathcal{A}$ 可以使用任意的非诚实敌手的证书，因此 $\mathcal{S}$ 需要抽取出 $\mathcal{A}$ 使用的随机数 q 确定执行 authentication 协议的用户的具体身份 *i*。随后 $\mathcal{S}$ 表非诚实的用户 *i* 和 *T* 交互。当 Authentication 协议执行成功后，*i* 会得到关于新的随机数 q' 的证书，$\mathcal{S}$ 抽取出 q'，将 U_q 中的项 (i, q) 更新为 (i, q')。

如果 $\mathcal{E}$ 可以区分出 $\mathcal{S}$ 和 $\mathcal{A}$，则说明 $\mathcal{A}$ 可为伪造零知识证明，或者伪造 CL 签名。根据零知识证明的正确性和 CL 签名的不可伪造性，上述 $\mathcal{S}$ 的模拟过程失败的概率是可以忽略的。

第二种情况：SP 被敌手控制。

1) Setup。$\mathcal{S}$ 代表一个诚实的用户和敌手 $\mathcal{A}$ 交互。$\mathcal{S}$ 从敌手 $\mathcal{A}$ 那里得到公开参数。

2) Registration。$\mathcal{S}$ 代表非诚实的 SP 和 T 交互/代表诚实用户和敌手 $\mathcal{A}$ 交互。T 告知 $\mathcal{S}$ 一个用户 i 需要注册，$\mathcal{S}$ 代表诚实用户 i 使用正确的注册协议和 $\mathcal{A}$ 进行注册。

3) Redeem。$\mathcal{S}$ 代表非诚实的 SP 和 T 交互/代表诚实用户和敌手 $\mathcal{A}$ 交互。T 告知 $\mathcal{S}$ 一个匿名用户需要回收票据，$\mathcal{S}$ 使用零知识证明模拟器来模拟零知识证明和 $\mathcal{A}$ 票据回收。

4) Authentication。$\mathcal{S}$ 代表非诚实的 SP 和 T 交互/代表诚实用户和敌手 $\mathcal{A}$ 交互。T 告知 $\mathcal{S}$ 一个匿名用户需要登录，$\mathcal{S}$ 使用零知识证明模拟器来模拟零知识证明以进行登录。

在这种情况下，由于零知识证明的零知识特性，$\mathcal{S}$ 的模拟过程是完美的。

综合以上 2 种情况，ARBRE 是安全的，即满足认证性、不可陷害性和匿名性。

7 ARBRE 性能分析

对 ARBRE 的性能进行分析，并和 PEREA、BLACR 以及 PERM 进行对比，因为后三者是目前和 ARBRE 最接近的基于信誉的匿名撤销系统。

7.1 复杂度分析

本文将 ARBRE 与 PEREA、BLACR 以及 PERM 从复杂度角度进行对比。

对于 ARBRE，注意到每次登录对应着一次票据赎回操作。因此在分析 ARBRE 每次登录的计算复杂度时，考虑一次 Authentication 协议的执行加上一次 Redeem 协议的执行。在服务提供者端，ARBRE 的计算量和信誉列表的长度无关，并且利用了票据集合中所有 K 个票据的乘积 T_U 的非成员证据来证明 K 个票据都不在恶意行为名单中，因此非成员证据验证的计算复杂度和 K 无关(注意到计算指数运算 $g^a \bmod N$ 的时候，当 a 较大时，由于 SP 知道 $\phi(N)$，因此可以先计算 $b = a \bmod \phi(N)$，再计算 $g^b \bmod N$)。整个登录认证中唯一和 K 相关的证明过程是将 Kl_t bit 的 T_U 分割为 $K/4$ 个 l_m bit 消息块 $\{T_i\}_{i=0}^{K/4-1}$，从而需要通过零知识证明 $T_U = \sum_{i=0}^{n-1} 2^{i \cdot l_m} T_i$，因此总的计算复杂度依然是 $O(K)$。在服务提供者端，PEREA 和 PERM 需要分别对 K 个票据分别做签名的零知识证明，因此计算复杂度也是 $O(K)$。但是 ARBRE 关于 K 的线性系数要远远小于 PEREA 和 PERM。这在下一节的实验结果中可以看出。

在客户端，主要的运算量来源于升级非成员证据，注意到 ARBRE 的票据长度只有 PEREA 的 1/4，并且只需要为 T_U 升级非成员证据，因此大幅度降低了客户端的运算量。表 1 给出了 ARBRE、PEREA，BLACR 以及 PERM 的复杂度分析，其中 ΔL 是自从上次登录后恶意行为列表中新增项的个数。

表 1 复杂度分析

方案	客户端	服务提供者端
ARBRE	$O(\Delta L)$	$O(K)$
PEREA	$O(K\Delta L)$	$O(K)$
BLACR	$O(\Delta L)$	$O(\Delta L)$
PERM	$O(K)$	$O(K)$

7.2 实验仿真

实验环境搭建于 Lenovo T4990d PC 上, 配有 Intel i7-3770M CPU 和 8 GB RAM。实验系统采用 C 语言实现，ARBRE 和 PEREA 使用 RSA 模上的指数运算，其实验代码使用 Miracl 库实现，BLACR 和 PERM 使用椭圆曲线上的对运算，其实验代码使用 PBC 库实现。

由于服务提供者 SP 需要同时处理多个用户的登录需求，容易成为整个系统的瓶颈，因此 SP 端的运行效率是评价方案最为重要的指标。

在 SP 端，ARBRE、PEREA 以及 PERM 的运行时间都和信誉列表长度无关。图 1 给出服务提供者端认证时间开销。当 K=80 时，PEREA 可以支持每分钟 59 次登录请求，PERM 可以支持每分钟 21 次登录请求。而 AEBRE 可以支持每分钟 820 次登录请求，这比 PEREA 和 PERM 要快一个数量级。BLACR 的登录认证时间和恶意行为列表长度线性相关，当恶意行为列表中有 10 万项的时候，BLACR 只能支持每分钟 5 次的登录请求。

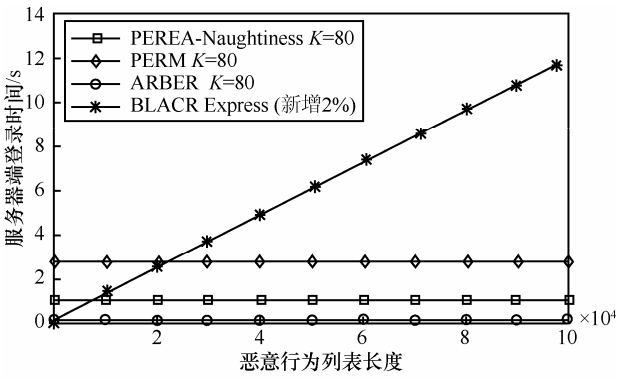

图 1 服务提供者端认证时间对比

图 2 给出了随着 K 的增长，ARBRE、PEREA 以及 PERM 在服务提供者端认证时间的增长对比。注意到对于 PEREA 和 PERM，用户在撤销窗口之内只能认证 K 次。因此如果服务提供者需要较长时间来评价会话，那么 K 必须增长以避免用户可用性上的限制。当 K 变大时，ARBRE 的认证时间增长得比 PEREA 以及 PERM 要慢得多。当 K=500 时，PEREA 可以支持每分钟 10 次登录请求，PERM 可以支持每分钟 4 次登录请求。而 AEBRE 可以支持每分钟 535 次登录请求，远超过 PEREA 和 PERM。

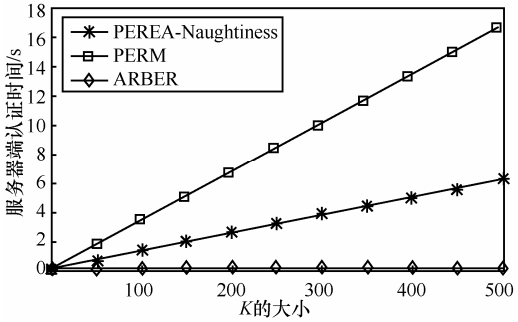

图 2　随着 K 的增长，服务提供者端认证时间对比

图 3 给出这些方案在用户端认证时间开销对比。PEREA 的主要缺点是每次认证前需要为 K 个票据升级相应的非成员证据，因此在用户端登录时间过长，ARBRE 只需要升级一个非成员证据，并且票据长度只有 PEREA 的 1/4，因此大大缩短了用户端的登录时间。当恶意行为列表中有 10 万项，并且假设上一次认证后新增了 2%的票据时，PEREA(K=80)每次登录需要 7 min，而 ARBRE 只需要 1.3 s。使用预计算，对于 BLACR 每次登录需要 5.6 s，而 PERM 需要 1 s。

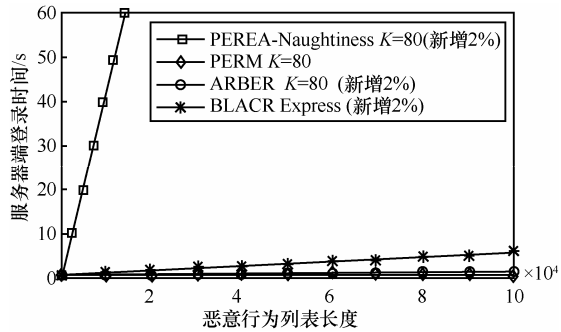

图 3 用户端登录时间对比

可以看出，ARBRE 在服务提供者端的表现远远超过其他体制，而在客户端表现和最快的 PERM 基本一致。

8　结束语

不基于可信第三方的匿名撤销方案可以在保护用户隐私的同时封禁恶意匿名用户，因此吸引了广泛的关注。最近的方案如 BLACR 和 PERM 实现了基于信誉的匿名撤销，大大提高了匿名撤销的灵活性，但是其在服务器端的认证效率并不能令人满意。另一方面，现有的快速匿名撤销方案如 FAUST 和 PE(AR)^2 则或功能不足或存在着安全问题。本文首先指出 PE(AR)^2 的安全问题，然后提出了一种新的基于信誉的匿名撤销方案 ARBRE。ARBRE 不要求所有的会话都必须在一个确定的时间窗口内被评估，也不需要和额外的提供登录限速的方案绑定。实验结果显示，该方案在服务器端的认证效率比现有方案快一个数量级，而在客户端的登录效率和当前最快的方案基本一致。

参考文献：

[1] LI J, LI N, XUE R.Universal accumulators with efficient nonmembership proofs[A]. Proceedings of ACNS 2007[C]. Springer, 2007.253-269.

[2] BONEH D, SHACHAM H. Group signatures with verifier-local revocation[A]. Proceedings of ACM CCS 2004[C]. ACM, 2004. 168-177.

[3] TSANG P P, KAPADIA A, CORNELIUS C, *et al*. Nymble: blockingmisbehaving users in anonymizing networks[J]. IEEE Transactions on Dependable and Secure Computing, 2011, 8(2): 256- 269.

[4] TSANG P P, AU M H, KAPADIA A, *et al*. BLAC: Revoking repeatedlymisbehaving anonymous users without relying on TTPs[J]. ACM Transactions on Information and System Security (TISSEC), 2010,13(4):39-46.

[5] BRICKELL E, LI J. Enhanced privacy ID: a direct anonymous attestation schemewith enhanced revocation capabilities[A]. Proceedings of WPES2007[C]. ACM, 2007.21-30.

[6] AU M H, TSANG P P, KAPADIA A. PEREA: practical TTP-free revocation of repeatedly misbehaving anonymous users[J]. ACM Transactions on Information and System Security (TISSEC), 2011, 14(4):29.

[7] CAMENISCH J, HOHENBERGER S, *et al*. How to win theclonewars: efficient periodic *n*-times anonymousauthentication[A]. Proceedings of ACM CCS 2006[C]. ACM, 2006.201-210.

[8] LOFGREN P, HOPPER N. Faust: efficient, TTP-free abuse prevention by anonymouswhitelisting[A].Proceedings of WPES 2011[C]. ACM, 2011.125-130.

[9] YU K Y, YUEN T H, CHOW S S, *et al*. PE (AR)^2: Privacy-enhanced anonymous authentication with reputation and revocation[A]. Proceedings of ESORICS 2012[C]. Springer,2012.679-696.

[10] AU M H, KAPADIA A, SUSILO W. BlACR: TTP-free blacklistable anonymous credentials with reputation[A]. Proceedings of NDSS 2012[C].San Diego, CA, USA.

（下转第 32 页）