

抗凭证泄露攻击的图形口令认证方案*

陈杰^{1,2}, 许春香^{1,2}, 张源^{1,2}, 蒋昌松^{1,2}, 韩云霞^{1,2}, 曹辰辰^{1,2}

¹(电子科技大学 计算机科学与工程学院, 四川 成都 611731)

²(电子科技大学 长三角研究院(湖州), 浙江 湖州 313001)

通信作者: 许春香, E-mail: chxu@uestc.edu.cn



摘要: 图形口令既可以减轻用户记忆传统文本口令的负担, 又可以简化用户输入口令的步骤, 近年来, 广泛应用于移动设备的用户认证. 现有的图形口令认证方案面临严峻的安全问题. 首先, 图形口令容易遭受肩窥攻击: 用户的登录过程被攻击者通过眼睛或者摄像头等方式偷窥导致图形口令泄露. 更为严重的是, 这类认证方案不能抵抗凭证泄露攻击: 服务器存储与用户图形口令有关的认证凭证并利用其验证用户身份, 攻击者如果得到服务器保存的凭证就可以通过离线口令猜测攻击恢复用户图形口令. 为了解决上述问题, 提出了一个安全的图形口令认证方案 (GADL). GADL 方案通过将随机的挑战值嵌入到用户的图形口令来抵御肩窥攻击, 因此攻击者即使捕获了用户的登录信息也无法得到用户图形口令. 为了解决服务器凭证数据库泄露问题, GADL 方案采用了一种确定性的门限盲签名技术来保护用户图形口令. 该技术利用多个密钥服务器来协助用户生成凭证, 使得攻击者即使获得凭证也无法实施离线猜测攻击来获得用户口令. 给出的安全性分析证明了 GADL 方案可以抵抗上述攻击. 此外, 给出了全面的性能分析表明 GADL 方案在计算、存储和通信开销这 3 个方面性能较高, 且在移动设备上易于部署.

关键词: 图形口令认证; 肩窥攻击; 口令猜测攻击; 凭证泄露攻击

中图法分类号: TP309

中文引用格式: 陈杰, 许春香, 张源, 蒋昌松, 韩云霞, 曹辰辰. 抗凭证泄露攻击的图形口令认证方案. 软件学报, 2023, 34(12): 5787–5806. <http://www.jos.org.cn/1000-9825/6759.htm>

英文引用格式: Chen J, Xu CX, Zhang Y, Jiang CS, Han YX, Cao CC. Graphical Password Authentication Scheme Against Credential Leakage Attacks. Ruan Jian Xue Bao/Journal of Software, 2023, 34(12): 5787–5806 (in Chinese). <http://www.jos.org.cn/1000-9825/6759.htm>

Graphical Password Authentication Scheme Against Credential Leakage Attacks

CHEN Jie^{1,2}, XU Chun-Xiang^{1,2}, ZHANG Yuan^{1,2}, JIANG Chang-Song^{1,2}, HAN Yun-Xia^{1,2}, CAO Chen-Chen^{1,2}

¹(School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China)

²(Yangtze Delta Region Institute (Huzhou), University of Electronic Science and Technology of China, Huzhou 313001, China)

Abstract: The graphical password mitigates the burden of memorizing traditional textual passwords and simplifies the process of entering passwords, which has been widely applied to user authentication of mobile devices in recent years. Existing graphical password authentication schemes face critical threats. First, graphical passwords are vulnerable to shoulder-surfing attacks, namely that users' graphical passwords may be leaked if attackers capture their login information through eyes or cameras. More seriously, these schemes are subject to credential leakage attacks. In other words, as the server stores authentication credentials related to the graphical passwords of users to verify their identities, if attackers obtain these credentials, they can perform offline password guessing attacks to retrieve users' graphical passwords. To solve the above problems, this study proposes a secure graphical password authentication scheme, dubbed GADL. GADL embeds random challenge values into the graphical passwords of users to resist shoulder-surfing attacks, and thus attackers cannot

* 基金项目: 国家自然科学基金 (61872060, 62002500); 四川省重点研发项目 (2021YFG0158)

收稿时间: 2022-03-28; 修改时间: 2022-06-21; 采用时间: 2022-08-02; jos 在线出版时间: 2023-03-02

CNKI 网络首发时间: 2023-03-03

obtain users' passwords even if they capture their login information. To address credential database leakage of the server, GADL utilizes a deterministic threshold blind signature technique to protect users' graphical passwords. In this technique, multiple key servers are utilized to assist users in the credential generation, which ensures that attackers cannot perform offline guessing attacks to obtain any knowledge of users' passwords even if they obtain users' credentials. The security analysis given in this study proves that GADL is resistant to the aforementioned attacks. In addition, the comprehensive performance evaluation of GADL demonstrates its high performance in terms of computation, storage, and communication overhead and proves that it can be easily deployed on mobile devices.

Key words: graphical password authentication; shoulder-surfing attack; password guessing attack; credential leakage attack

智能手机等移动设备的快速发展,方便了用户的生活. 用户可以利用这些设备通过公开网络向远程的服务提供商请求服务,比如数据共享服务. 然而,用户在享受这些便利的同时也面临着安全威胁:攻击者试图冒充用户登录到服务器中以实施诈骗或者其他行为来获取利益. 因此服务提供商需要一个安全的身份认证方案确认用户身份^[1].

基于口令的身份认证方案是移动用户使用最为普遍的认证方案^[2-4]. 用户在注册时选择一个用户名和口令,服务器根据这些信息生成一个凭证保存在本地,在登录阶段,用户通过安全信道向服务器提供正确的用户名和口令,服务器按照同样的方式计算凭证并与本地保存的凭证进行对比来判断用户身份.

文本口令是用户使用最为广泛的口令认证形式. 文本口令一般由字母、数字和特殊字符组成. 用户在请求服务前,需要正确地输入文本口令以完成身份认证. 用户可以设置长且复杂的文本口令来降低攻击者猜测出用户口令的概率. 然而,这种方式虽然会增加安全性,但同时也会增加用户的记忆负担,特别是在用户需要记忆很多口令来访问不同服务的情况下. 因此用户倾向于设置短且低熵的口令来减轻记忆负担,但这容易遭受在线口令猜测攻击^[5,6]和离线口令猜测攻击^[7,8]. 此外,用户还倾向于在不同的服务器使用姊妹口令(具有一定联系的口令)^[9,10]来减轻记忆负担,但是这存在较高的安全风险. 例如,如果一个服务器的数据库遭到攻击导致用户的口令泄露,攻击者就可以推测出用户在其他服务器使用的口令并冒充其登录.

图形口令是面向移动用户最为便捷的认证方式. 这类口令允许用户选择一些图片或者图片中的一些像素点来作为其口令. 在登录的时候用户选择这些图片或者点击这些像素点来认证自己的身份. 与文本口令相比,图形口令有两个优势: (1) 便于记忆^[11,12]. 图形口令可以使用户联想到某一个场景或者事物,加深用户的记忆,从而减轻了记忆负担. (2) 便于输入. 用户可以采用滑动或者点击图片的方式来达到认证的目的,避免了文本口令容易输入错误的问题.

然而,图形口令认证方案存在严峻的安全性问题. 首先,图形口令容易遭受肩窥攻击,即攻击者通过眼睛或者摄像头等方式偷窥用户在登录时操作图形界面的过程,以试图通过此方式窃取用户的图形口令并冒充用户登录. 更为严重的是,图形口令认证方案继承了口令认证方案存在的安全性问题,即这类方案无法抵抗凭证泄露攻击:在得到用户的图形口令凭证后,由于用户的图形口令信息熵较低,攻击者可以穷举所有可能的图形口令,计算每一个图形口令对应的凭证,并与服务器保存的凭证进行匹配,直到找出用户的图形口令.

考虑到上述问题,本文提出了一个图形口令认证方案,称作 GADL. 为了抵抗肩窥攻击, GADL 方案引入数字挑战值并将其嵌入在用户的图形口令中,用以隐藏用户真实的图形口令. 攻击者即使偷窥用户操作图形界面的过程也无法得到数字挑战值,因此无法得到用户图形口令. 为了解决服务器凭证泄露带来的安全性问题, GADL 方案采用了确定性的门限盲签名技术来协助用户生成凭证. 具体来讲, GADL 方案将服务器分为认证服务器和密钥服务器,并采用哈希加盐法生成凭证,但是盐不是认证服务器在注册阶段为用户生成的随机字符串,而是用户在注册阶段产生一个随机密钥并利用其计算出的签名值. 并且,盐没有保存在认证服务器中,而是在每一次认证的时候由用户动态地恢复出来,为此,用户在注册阶段将产生的随机密钥以门限秘密共享的方式分布在密钥服务器群组中,在认证时可以利用自己的图形口令以门限盲签名方法恢复出盐值. 这种方式可以有效保证攻击者即使获取了服务器保存的用户凭证也不能恢复出用户的图形口令,并且分布式保存用户密钥的方式即使在攻击者窃取一些服务器密钥的情况下也可以为用户的密钥提供安全性保证. 本文给出了 GADL 方案的安全性分析,证明 GADL 方案可以抵抗多种攻击. 此外,本文基于安卓平台实现了 GADL 方案,并且给出了全面的性能评估表明 GADL 方案在计算、通信和存储 3 个方面是高效的.

本文第1节介绍了相关工作,第2节阐述了系统模型与攻击模型,第3节给出了预备知识和 GADL 方案的概述,第4节详细描述了 GADL 方案的设计思路,第5节给出了 GADL 方案的安全性证明,第6节给出了 GADL 方案的性能评估,第7节对全文进行了总结。

1 相关工作

随着人们安全意识的提高,大量用于保护用户账户安全的认证方案被提出来。认证方案主要分为3类:生物特征认证方案^[13]、文本口令认证方案^[14]和图形口令认证方案^[15,16]。

传感器和智能手机的快速发展使得生物特征认证技术在近年来得到广泛应用^[17,18]。准确可靠的生物特征识别技术,如指纹识别、人脸识别、虹膜识别,能够方便人们完成身份认证。然而生物特征识别技术面临一个问题,即生物特征无法更改^[19]。生物特征的泄露可能导致很多未授权的行为,比如用户生物特征数据被用于机器学习^[20],这会引起用户的担忧。

除了生物识别技术,传统的文本口令在近几十年里使用得最频繁^[21,22]。它们没有类似生物特征识别的问题,可以随意更改。包含大写字母、小写字母、数字和特殊字符的文本口令可能非常复杂,使得攻击者很难正确猜测。并且,它们可以定期更换。文本口令存在的最大问题是难以记忆。随着用户账户的增加,他们往往难以清楚地记忆每一个账号对应的口令。为了解决这个问题,人们倾向于使用姊妹口令^[9,10],然而这增加了口令泄露的风险,因为攻击者可以从用户的一个口令中推测出用户其他账号对应的口令。

图形口令相较于传统的文本口令而言,具有两个优点。首先,图形口令易于输入。用户仅需要点击或者滑动图片,简化了文本口令的输入步骤^[23]。其次,图形口令易于记忆^[11,12]。图形口令可以让人们联想到一些场景或者事物,因而加强了用户的记忆。

由于早期设备处理能力较弱,因此基于图形口令设计的认证方案较为简单,如1999年的 DAS 方案^[24]。该方案需要用户在给定的方格中绘制图案来验证用户身份。图案越复杂,安全性越高。然而用户绘制的图案一般不会很复杂,因而口令空间较小。

后来 Wiedenbeck 等人^[25]提出了一个称作 PassPoints 的方案。该方案允许用户选择图片中的某一些像素点来作为图形口令。用户在登录的时候需要点击这些点或者其附近的区域来验证身份。由于可点击的区域较大,使得该方案可以有效抵抗字典猜测攻击。然而大部分用户选择的图片中的像素总是热点位置^[26],这会使得攻击者更容易利用热点猜测用户口令。为了缓解热点猜测问题,Chiasson 等人^[27]提出了 CCP 方案,利用多个图片上的不同点来代替一张图片上的不同点,在一定程度上提高了安全性。然而 CCP 方案还是会受到热点猜测攻击,他们又提出了 PCCP 方案^[28]。在 PCCP 方案中,系统随机为用户指定图片中的某一块区域,用户只能从图片中的这一块区域中选择自己的口令,而不能从整个图片中选择,避免了所有用户都选择图片中热点区域的点。

上述的方案均不能抵御肩窥攻击。后面许多研究工作被提出来用于抵御肩窥攻击。能够抵御肩窥攻击的方法包括多轮交互^[29]、增加随机值来隐藏真实的口令^[30,31]、避免直接点击用户口令^[14,32]等。

Wiedenbeck 等人^[29]提出的 CHC 方案要求用户在注册阶段选择至少3个图案。在登录阶段,系统将所有图案随机放置在屏幕上的各个位置。用户需要找到自己所选择的图案的位置,然后确定出这些图案所围成的区域。用户仅需要点击该区域内的图案即可完成一次挑战-响应过程,并重复此步骤,直到所有的挑战-响应都通过。然而 CHC 方案只需要用户选择3个图案即可,更多的图案对安全性并没有明显提高,并且多次的挑战-响应过程增加了用户登录所需时间,不方便用户使用。

Yu 等人^[33]提出的 EvoPass 方案在注册阶段将用户上传的全部图片转化为只有黑白颜色的素描。认证过程中屏幕上会出现很多的素描,用户需要从这些素描中识别并选择自己注册阶段上传图片的素描。每一次登录过程,所有素描均随机出现在屏幕的各个位置,并且属于该用户素描的数量也是随机的,攻击者通过偷窥难以识别用户的素描,因而该方案可以在一定程度上缓解肩窥攻击。

CCGP 方案^[34]允许用户通过在屏幕上绘制汉字来代替文本输入。由于用户输入的汉字时间一般较长,因此该方案可以一定程度上抵御偷窥。然而登录时间过长会影响用户体验,登录时间较短就容易因偷窥导致用户口令泄露。

此外还有一类图形口令认证方案将图形口令与文本口令结合起来. Zhao 等人^[15]提出的 S3PAS 方案是一个较为早期的将图形和文本相结合的认证方案. 用户在注册阶段仍然采用文本口令. 在认证阶段每一个字符随机出现在屏幕上, 用户需要顺次点击这些字符围成的三角形内部区域. S3PAS 方案避开了用户直接输入或者点击口令这个过程, 使得攻击者即使偷窥了用户的登录过程也不能得到用户的图形口令. 然而这个方案中, 用户仍然需要记忆文本口令, 记忆难度较大.

Chakraborty 等人^[31]提出的 MobSecure 方案由 36 个不同颜色表示的节点, 且每一个节点表示一个大写字母或数字. 在用户登录的时候, 系统通过其他硬件 (如耳机) 产生一系列随机值. 用户需要结合这些随机值和自己注册时选择的真实口令生成临时口令, 并在屏幕上依次点击这些临时口令所包含的字符或者数字. 然而, 用户每次登录都需要借助耳机等硬件获取随机值, 并结合真实口令生成新的临时口令, 使得用户在登录的时候需要花费较长的时间, 并且在某些时候对用户而言并不方便.

Nyang 等人^[35]提出的 two-thumbs-up 方案去除了 MobSecure 方案需要硬件设备的限制. 该方案利用用户的肢体去抵御肩窥攻击. 用户在登录时需要双手同时放在屏幕上指定位置才会在屏幕上显示挑战值, 并根据此挑战值确定出自己的临时口令. 这种动作会遮挡屏幕的大部分区域, 使得攻击者很难通过偷窥得到该挑战值. 然而该方案需要双手同时放在屏幕指定位置, 有时候用户不愿意这样做或者不能这样做.

Sun 等人^[30]提出了一个方案称作 PassMatric. 该方案将每个图片划分为多个方块. 在注册阶段, 用户需要从图片中选择某一个方块作为自己的口令. 在验证阶段, 系统创建一个包含随机字母和数字的指示器. 用户需要根据该指示器滑动坐标栏使得所选择的方块的坐标与指示器所显示的数字相对应. 然而该方案不能抵御录屏攻击. Fang 等人^[32]在 PassMatric 方案的基础上提出了 THP 方案. 该方案不再使用随机生成的指示器, 而是将这一部分整合到用户口令中, 因而用户的口令由两个图形口令因子组成. 当用户进行身份验证时, 他需要先定位自己的第一个因子所在的坐标, 然后根据此坐标去滑动第 2 因子所在的图形界面上的坐标, 使得第 2 因子的坐标与之对应. 由于每一次登录过程中, 用户的第 1 因子的位置随机出现, 且 THP 方案避免了用户直接点击口令, 使得该方案能够抵御偷窥和录屏攻击.

然而我们注意到现有的图形口令认证方案关注点主要集中在抵御肩窥攻击和易用性方面, 却缺少对服务器数据库凭证安全性的关注. 在这类方案中, 服务器存储一个与用户口令有关的凭证并用该凭证来校验用户身份. 一旦攻击者窃取服务器端的凭证数据后, 可以通过口令猜测等方式恢复出用户的图形口令, 然后冒充用户访问服务器, 给用户的隐私和财产安全带来极大威胁.

2 问题描述

2.1 系统模型

图 1 是移动环境下用户身份认证的系统模型. 此模型包含 3 类实体, 分别为用户、密钥服务器和认证服务器.

- 用户. 系统中包含许多用户, 且每一个用户均相互独立. 每一个用户拥有一台移动设备, 用户使用该设备完成图形口令的输入, 然后再进行身份认证.

- 密钥服务器. 系统中包含 n 个相互独立的密钥服务器. 密钥服务器不提供身份认证功能, 其作用是保存在注册阶段用户发来的信息, 并在认证阶段利用保存的信息协助用户恢复凭证. 每一个密钥服务器保存一份用户的子密钥 $s_i, 1 \leq i \leq n$, 并且在认证阶段根据用户发送来的请求利用该子密钥计算一个响应结果.

- 认证服务器. 认证服务器的功能是认证用户身份 $cred$. 在注册阶段, 认证服务器保存用户计算得出的凭证 $cred$. 在登录阶段, 用户将自己的凭证 $cred$ 发送给认证服务器, 认证服务器校验该 $cred$ 是否正确来确认用户身份. 如果校验成功, 则用户通过身份认证.

整个系统模型分为两部分: 注册部分和登录部分. 在注册部分, 用户使用安全信道 (如 TLS^[36]安全信道) 与服务器 (包括密钥服务器和认证服务器) 通信. 在登录部分, 用户与密钥服务器的通信消息在公开的网络上传输, 与认证服务器通信则使用安全信道.

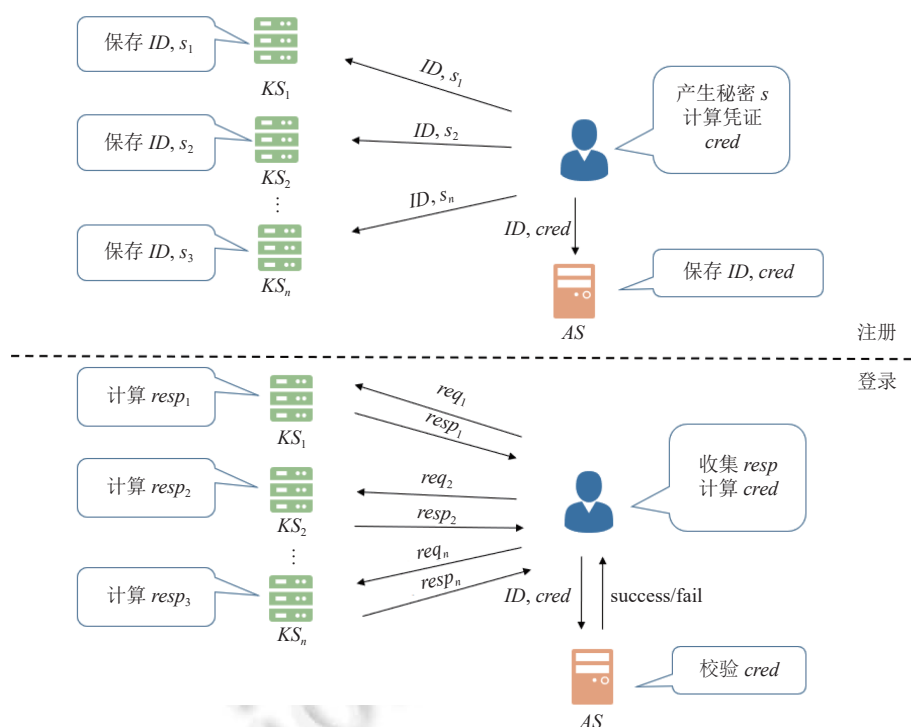


图1 系统模型

注册时, 用户选择一个唯一的身份凭证 ID 和对应的图形口令 gpw . 选定后, U 生成一个随机密钥 s , 然后利用该密钥与图形口令计算凭证 $cred$, 并将 $cred$ 发送给认证服务器进行注册. 之后 U 将密钥 s 拆分为 n 个子密钥, 并通过安全信道将这些子密钥分别发送给 n 个密钥服务器, 密钥服务器收到后保存用户名及对应的子密钥.

登录时, 用户首先根据图形口令 gpw 生成请求 req , 然后将该请求分别发送给 n 个密钥服务器. 每一个密钥服务器收到后利用本地保存的子密钥信息计算响应结果 $resp$, 并将其发送给用户. 用户收到后将它们聚合成一个完成的结果并恢复出凭证, 然后通过安全信道将凭证发送给认证服务器. 认证服务器收到后校验该凭证的正确性. 如果校验成功, 则用户登录成功.

2.2 威胁模型

现实中, 用户经常在公共场合登录到服务器以请求服务. 然而, 用户在公共场合操作图形界面的过程容易被偷窥, 导致图形口令泄露. 攻击者也可能执行在线字典猜测攻击获得用户口令, 或者监听用户在公开网络中发送的消息以获得有用信息. 此外, 保存在服务器内部的用户凭证如果被敌手获得, 可能导致攻击者实施离线口令猜测攻击恢复出用户口令. 具体而言, 包含以下攻击方式.

(1) 肩窥攻击. 图形口令认证方案操作简单, 比如点击或者滑动某一个区域, 因此用户在图形界面的操作过程容易被偷窥. 攻击者试图通过各种手段偷窥正常用户的登录过程以试图推测出用户的图形口令, 然后利用用户的 ID 和得到的图形口令冒充用户登录到服务器中.

(2) 在线字典猜测攻击. 攻击者可以通过在线字典猜测攻击去试探正常用户的图形口令. 比如, 攻击者事先准备好一个 ID 集合 Set_{ID} , 记为 $\{ID_1, ID_2, \dots\}$ 和一个图形口令集合 Set_{gpw} , 记为 $\{gpw_1, gpw_2, \dots\}$. 对于 Set_{gpw} 中的每一个 gpw , 攻击者依次取出 Set_{ID} 中的 ID , 然后计算凭证并发送给服务器进行验证. 如果验证成功, 攻击者就猜测成功, 否则猜测失败, 攻击者从 Set_{ID} 中取出下一个 ID , 重新执行上述步骤. 当 Set_{ID} 中所有的 ID 均测试完毕后, 攻击者重新在 Set_{gpw} 中选择一个新的 gpw , 重复上述步骤.

(3) 监听攻击. 攻击者可以监听用户与密钥服务器之间的通信信息, 试图从中恢复用户图形口令, 或产生可通

过验证的凭证.

(4) 凭证泄露攻击: 攻击者在获取数据库的凭证信息后, 可以不断地猜测所有可能的图形口令, 直到找到用户的图形口令. 此外, 攻击者还可以根据得到的凭证信息直接伪造能够通过服务器验证的凭证数据.

2.3 设计目标

(1) 功能性: GADL 方案能够正常满足用户需求.

(a) 可用性: 拥有 ID 和正确图形口令 gpw 的用户可以登录到服务器中, 而没有正确图形口令的攻击者不能登录到服务器中.

(b) 可迁移性: 用户仅仅要记住 ID 和 gpw , 且用户设备中不需要保存任何与用户身份有关的数据. 这样用户可以在任何一个设备上通过 ID 和 gpw 实现身份认证.

(c) 高效率: GADL 方案包含用户、认证服务器和密钥服务器 3 类实体. 这 3 类实体在注册和登录两个阶段的计算、通信以及存储开销都尽量小.

(2) 安全性: GADL 方案可以有效抵御前述威胁模型中提及的各类攻击.

3 方案介绍

3.1 符号表示

本文使用 G 表示一个加法群, 其生成元为 Q , 阶为素数 p . H_1 和 H_2 代表安全的哈希函数, 其中 $H_1: \{0, 1\}^* \rightarrow G$, $H_2: \{0, 1\}^* \rightarrow Z_p$. e 表示双线性映射 $G \times G \rightarrow G_T$, 其中 G_T 是 p 阶乘法群. 用户的图形界面包含图案界面和方块界面. 一个图案界面包含 $u \times v$ 个图案. 一个方块界面包含 $u \times v$ 个方块, 每一个方块的位置记为 (i, j) , $1 \leq i \leq u, 1 \leq j \leq v$, 其中 i, j 分别表示第 i 行和第 j 列. 用户的图形口令包含图案和方块两部分, Set_g 表示系统中的图案集合, P_g 表示用户所选择的图形口令里的图案, S_q 表示用户选择的图形口令里的方块, gpw 表示用户的图形口令 $gpw = (P_g, S_q)$. $[1, n]$ 表示整数 1 到 n 的元素集合, 它也可以记为 $\{1, 2, \dots, n\}$. $x||y$ 表示将两个字符串 x 和 y 拼接. Z_p 表示模 p 的剩余类环, 由于 p 是一个素数, 因此 Z_p 可以视为一个集合 $\{0, 1, \dots, p-1\}$, Z_p^* 是模 p 的乘法群, 它可视为集合 $\{1, 2, \dots, p-1\}$.

3.2 预备知识

(1) 双线性映射

给定两个 p 阶群 G, G_T , 其中 p 为素数, G 为加法群, G_T 为乘法群. 存在一个双线性映射 $e: G \times G \rightarrow G_T$ 当且仅当同时满足下列 3 条性质.

(a) 双线性性: 对于任意 $P, Q \in G, a, b \in Z_p^*$, 满足 $e(aP, bQ) = e(P, Q)^{ab}$.

(b) 非退化性: 任意 $P, Q \in G$, 且 $P \neq Q$, 使得 $e(P, Q) \neq 1_{G_T}$, 其中 1_{G_T} 为 G_T 的单位元.

(c) 可计算性: 对于任意 $P, Q \in G$, 存在有效的计算方法计算 $e(P, Q)$.

(2) 秘密共享与门限签名

(t, n) -门限秘密共享技术^[37]用于在 n 个参与者中共享一个秘密 s . 秘密 s 被拆分为 n 个子秘密, 其中每一个参与者获得一个子秘密, 且任何一个参与者均没有关于秘密 s 的信息. 在这 n 个参与者中, 任意诚实的 t 个参与者合作可以恢复出共享的秘密, 但是如果少于 t 个参与者参与将不能得到任何有用信息. 该方案通过构建一个 $t-1$ 次多项式来生成秘密碎片, 并通过拉格朗日插值法和 t 个子秘密来恢复出共享的秘密 s .

(t, n) -门限签名^[38-41]指 n 个签名者各含有一对私钥和公钥, 且共同组成一个签名群体, 该群体拥有一对公私钥, 其中任意诚实的 t 个签名者的签名都可以恢复出群体的签名, 但是如果少于 t 个签名者对消息的签名则不能恢复出群体对消息的签名.

(3) 哈希函数

哈希函数 H 是指将任意长的输入映射为固定长度且长度较短的值作为消息摘要. 在本文中, 一个安全的哈希函数 H 是指该哈希函数具有抗碰撞性.

定义 1. 一个安全的哈希函数 H 具有抗碰撞性, 是指对于任何一个概率多项式敌手 $\mathcal{A}$, 存在一个可忽略函数 $negl$, 使得:

$$\Pr_{\mathcal{A}, \lambda}^{\text{FindCollision}} \leq negl(\lambda) \quad (1)$$

其中, FindCollision 是一个查找碰撞的游戏, 该式表示 $\mathcal{A}$ 赢得游戏的概率是可忽略的. 游戏定义如下.

- (a) $\mathcal{A}$ 选择 x 和 x' ; 当且仅当 $x' \neq x$ 且 $H(x') = H(x)$ 时, 返回 1. 否则, 返回 0.
- (b) $\mathcal{A}$ 至多尝试 $poly(\lambda)$ (λ 的多项式) 次得到 1 时赢得游戏, 否则 $\mathcal{A}$ 失败.

(4) BLS 签名

BLS 签名算法由 Boneh 等人^[42]提出. 给定一个生成元为 Q 的加法群 G 和一个安全的哈希函数 $H: \{0, 1\}^* \rightarrow G$, 用户私钥为 sk , 公钥 $pk = sk \cdot Q$, 则用户对消息 msg 产生的 BLS 签名为 $\sigma = sk \cdot H(msg)$, 验证签名的方法为: $e(H(msg), pk) = e(\sigma, Q)$.

3.3 GADL 方案概述

GADL 方案采用确定性的门限盲签名技术保护用户的图形口令. 具体来说, 该方案采用哈希加盐法的方式生成凭证, 即 $cred = H(gpwl||salt)$, 但是盐 $salt$ 不是服务器端为用户生成的随机字符串, 而是用户在注册阶段利用选择的密钥 s 对图形口令产生的 BLS 签名. 盐值没有暴露给服务器, 也没有保存在服务器端, 而是由用户在认证阶段动态地恢复出来. 为此, 用户在注册阶段需要将该密钥以门限秘密共享技术分布在密钥服务器群组中, 在认证阶段用户以门限盲签名技术同密钥服务器群组交互恢复出密钥 s 对图形口令的签名值, 并将该值作为盐计算凭证. 同时, 盲签名技术在用户与密钥服务器群组交互中可以保证用户图形口令的安全性.

GADL 方案包含 6 个算法: Setup、Register、Share、Check、Recovery 和 Authenticate.

(1) Setup: 此算法的作用是生成整个系统所需要的公共参数 PP , 这些参数在所有用户、认证服务器和密钥服务器中共享.

(2) Register: 用户执行此算法可以将自己的信息注册到认证服务器中. 此算法随机生成一个私钥 s , 利用 BLS 签名算法计算签名, 将签名结果作为盐生成凭证, 并向认证服务器进行注册.

(3) Share: 此算法允许用户将 Register 算法产生的密钥 s 以 (t, n) -门限秘密共享的方式分布在 n 个密钥服务器中.

(4) Check: 此算法允许密钥服务器在认证阶段协助用户恢复 BLS 签名时校验该用户的请求频率是否超过限制, 如果超过限制, 则密钥服务器拒绝该用户的请求.

(5) Recovery: 此算法允许用户和密钥服务器群组交互将密钥 s 对用户图形口令的签名恢复出来.

(6) Authenticate: 此算法允许用户利用恢复出的签名结果计算认证凭证并向认证服务器认证自己的身份.

GADL 方案包含初始化、注册和认证 3 个部分. 整个系统仅执行一次初始化步骤. 一个用户需要注册一次, 可以认证多次.

在初始化部分, 系统执行 Setup 算法, 生成公共参数 PP , PP 在所有实体中共享.

在注册阶段, 用户需要选择自己的用户名和图形口令, 然后在服务器端注册. 本文假定每一个用户的注册环境是安全的 (例如, 任何人都不会偷窥他的注册过程, 并且用户使用安全信道和密钥服务器以及认证服务器进行通信, 如 TLS^[36]信道). 一个新用户注册时需要提供用户名和图形口令, 其中图形口令包含图案和方块两个部分. 用户在注册阶段的主要步骤如下.

- (1) 选择一个唯一的用户名 ID 和对应的图形口令 gpw (图形口令包含图案和方块).
- (2) 执行 Register 算法将自己的信息注册到认证服务器, 执行过程会产生一个私钥 s .
- (3) 执行 Share 算法将上述私钥 s 以秘密共享的方式分布在 n 个密钥服务器中.

在注册阶段用户使用了 BLS 签名计算凭证, 且签名对应的私钥共享在 n 个密钥服务器中, 因此在登录阶段, 用户 $\mathcal{U}$ 需要和密钥服务器群组交互安全地将 BLS 签名恢复出来, 然后再恢复认证凭证. $\mathcal{U}$ 需要执行以下步骤来完成身份认证.

- (1) 输入自己的用户名 ID , 然后正确地操作图形界面完成图形口令的输入.

- (2) 执行 Recovery 算法恢复 BLS 签名.
- (3) 执行 Authenticate 算法利用恢复出的 BLS 签名进行认证.

3.4 方案对比

GADL 方案与现有的图形口令认证方案对比, 具有以下几个优势: 首先, GADL 方案密钥空间较大, 可以有效抵抗攻击者进行在线字典猜测攻击. 其次, 为了抵抗肩窥攻击, GADL 方案将随机的数字挑战融入到用户口令中, 避免了使用耳机等设备来接收挑战值, 也不需要使用手或者身体来遮挡屏幕上的挑战值, 因此在抵抗偷窥攻击的同时使得方案简单易用. 更为重要的是, GADL 方案考虑到了服务器凭证数据库泄露带来的安全问题, 即攻击者在得到凭证后可能通过离线穷举可能的图形口令的方式恢复出用户的图形口令. 为了解决这个问题, GADL 方案引入了确定性的门限盲签名技术来协助用户生成认证凭证. 因此相较于其他图形口令认证方案来说, GADL 方案可以抵抗凭证泄露攻击, 这也是 GADL 方案的最大优势. 表 1 给出了 GADL 方案与其他方案的对比情况, 其中 ω 表示用户选择的图形口令 (组) 数. 按照本文实验设定的情况下, 即 $u = 11, v = 7, |Set_g| = 143$, 一对图形口令 (包含图案和方块) 的可能组合有 $11 \times 7 \times 143 = 11011$, GADL 方案允许用户一次完成一组 (两对) 图形口令的输入, 一组图形口令的可能组合有 $11011 \times 11011 = 121242121$, 达到了 10^8 , 因此, GADL 的密钥空间为 $10^{8\omega}$.

表 1 方案对比

方案	抵抗肩窥攻击	抵抗录屏攻击	易于记忆	密钥空间	易用性	抵抗服务器凭证泄露
PassPoints ^[25]	×	×	×	$>10^8$	√	×
S3PAS ^[14]	×	×	×	$>10^8$	√	×
MobSecure ^[31]	√	√	×	$>10^8$	×	×
two-thumbs-up ^[35]	√	×	√	$>10^8$	×	×
PassMatric ^[30]	√	×	√	$10^{2\omega}$	√	×
THP ^[32]	√	√	√	$10^{4\omega}$	√	×
GADL (本文)	√	√	√	$10^{8\omega}$	√	√

4 GADL 方案设计

GADL 方案分为初始化、注册、认证 3 个阶段.

4.1 初始化阶段

初始化阶段整个系统只执行一次. 初始化时, 执行 Setup 算法确定出系统参数 PP , PP 在所有用户、密钥服务器和认证服务器中共享.

算法 Setup: 给定一个安全参数 λ , 系统进行初始化, 确定出系统参数 PP :

$$PP = \{G, G_T, e, Q, p, H_1, H_2, Set_g, Set_p, Max_{req}\} \tag{2}$$

其中, G 是阶为素数 p , 生成元为 Q 的加法群, G_T 是 p 阶乘法群, $e: G \times G \rightarrow G_T$ 代表双线性映射, $H_1: \{0, 1\}^* \rightarrow G$, $H_2: \{0, 1\}^* \rightarrow Z_p$ 代表两个安全的哈希函数, Set_g 代表用户可选择的图案集合, 用户图形口令的第 1 部分只能在 Set_g 中选择, Set_p 表示图片集合, 图片可以协助用户记忆方块的位置, 不作为图形口令的组成部分, 用户可以从 Set_p 中选择图片, 也可以从本机选择图片, Max_{req} 表示认证服务器允许一个用户的最大连续错误认证次数.

4.2 注册阶段

用户使用移动设备选择图形口令, 移动设备上保存了系统中的整个图案集合 Set_g 和图片集合 Set_p . 用户需要在移动设备的图形界面上选择图形口令, 记为 gpw . gpw 包含用户图案 P_g 和方块 S_q 两部分, 用户需要分别选出 P_g 和 S_q . 用户操作图形界面的具体步骤如下.

(1) 选择 P_g . 客户端从图案集合 Set_g 中随机选择 h 个图案, 并将其显示在屏幕上, 这些图案记为 $S_{r,P} = \{P_{r,1}, P_{r,2}, \dots, P_{r,h}\}$. 用户需要在 $S_{r,P}$ 中选择任意两个图案作为图形口令的第 1 部分, 记为 P_g .

(2) 选择 S_q . 系统在屏幕上展示 $u \times v$ 个方块, 其中每个方块的坐标对应为 (i, j) , $1 \leq i \leq u, 1 \leq j \leq v$. 用户选择任意两个方块, 其坐标分别为 (x_1, y_1) 和 (x_2, y_2) , 记为 $S_q = ((x_1, y_1), (x_2, y_2))$.

为了更好地解释注册阶段的具体过程, 本文使用一个实例来描述用户的操作流程. 实例中取 $u = 11, v = 7$, 如图 2 所示. 用户 $\mathcal{U}$ 操作客户端图形界面的详细步骤如下.



图 2 注册阶段

(1) 选择用户名 ID . ID 是用户的身份标识, 必须具有唯一性. 用户选择 ID 后, 认证服务器从数据库中查找是否存在相同的 ID . 如果存在, 则 $\mathcal{U}$ 需要重新选择 ID (此步骤未在图 2 中展示).

(2) 选择两个图案作为 $\mathcal{U}$ 的图形口令的第 1 部分, 记为 P_g . 如图 2(a) 所示, 客户端从图案集合 Set_g 中随机选取 77 张不同的图案, 并将其随机打乱, 排列为 11 行 7 列, 图案包括狮子、熊猫、足球等. $\mathcal{U}$ 从中选择任意两张图案. 如图 2 中显示, $\mathcal{U}$ 选择了“apple”和“bag” (见红方框).

(3) 从本地数据库中选择一张图片. 客户端将 Set_p 中所有的图片呈现在屏幕上, 如图 2(b) 所示. $\mathcal{U}$ 通过滑动来浏览所有的图片, 并且选择其中一张图片. 图 2(c) 是用户选择的图片 (值得注意的是, 这张图片不是图形口令的组成部分. 它的作用是为了帮助用户记忆后续步骤中选择的方块的位置, 用户也可以选择本机图片).

(4) 选择两个方块, 记为 S_q . 用户选择的图片被分割成 11 行 7 列 ($u = 11, v = 7$), 共 77 个小方块. 每一个方块的位置记为 (i, j) , 表示第 i 行第 j 列. $\mathcal{U}$ 需要从中选取两个方块作为他的图形口令第 2 部分. 如图 2(d) 所示, $\mathcal{U}$ 首先选择了第 7 行 7 列的红色方块, 然后选择了第 5 行 2 列的红色方块. 所以 $S_q = ((7, 7), (5, 2))$.

用户选择完成之后, 客户端将用户图形口令的第 1 部分 (本例中为“apple”和“bag”) 存储在本地, 方便用户下次登录. 如果客户端不存储, 用户也可以在后续登录过程中指定.

用户按照上述步骤操作完毕后, 得到图形口令 $gpw = ((apple, bag), ((7, 7)(5, 2)))$, 然后执行算法 Register 将自己的信息注册到认证服务器中.

算法 Register: 用户 $\mathcal{U}$ 使用自己的用户名 ID 和图形口令 gpw 向认证服务器进行注册. 具体步骤如下所示.

- (1) $\mathcal{U}$ 产生一个随机密钥 s , 满足 $s \in Z_p^*$.
- (2) $\mathcal{U}$ 利用 s 计算 BLS 签名 $\sigma = s \cdot H_1(gpw)$.
- (3) $\mathcal{U}$ 计算凭证 $cred^* = H_2(gpw \parallel \sigma)$.
- (4) $\mathcal{U}$ 和认证服务器建立一个安全的信道, 并利用该信道安全地将 ID 和 $cred^*$ 发送至认证服务器 $\mathcal{AS}$.
- (5) $\mathcal{AS}$ 校验用户信息的正确性后, 计算 $cred = H_2(cred^*)$, 然后为 $\mathcal{U}$ 生成一个计数变量 $count = 0$, $\mathcal{AS}$ 将 $count$ 和 $cred$ 保存在本地, 并给 $\mathcal{U}$ 发送“success”.

用户在执行算法 Register 阶段生成了一个密钥 s 用于生成 BLS 签名. 为了能够在登录阶段恢复出 BLS 签名,

用户还需要将该密钥 s 安全地共享在 n 个密钥服务器中, 因此用户需要执行算法 Share.

算法 Share: 用户 $\mathcal{U}$ 利用门限秘密共享技术将密钥 s 共享在 n 个密钥服务器. 具体步骤如下所示.

(1) $\mathcal{U}$ 随机生成一个 Z_p^* 上的 $t-1$ 阶多项式 $f(x) = s + a_1x + a_2x^2 + a_3x^3 + \dots + a_{t-1}x^{t-1}$, 其中 $\{a_i\}_{1 \leq i \leq n} \in Z_p^*$.

(2) $\mathcal{U}$ 选择 n 个随机数, 分别记为 $x_1, x_2, \dots, x_n$, 且满足 $\{x_i\}_{1 \leq i \leq n} \in Z_p^*$; 计算 $y_1 = f(x_1), y_2 = f(x_2), \dots, y_n = f(x_n)$.

(3) $\mathcal{U}$ 与 n 个密钥服务器建立安全信道, 并将 $(ID, x_1, y_1), (ID, x_2, y_2), \dots, (ID, x_n, y_n)$ 分别安全地发送给 n 个密钥服务器.

(4) 密钥服务器 $\mathcal{KS}_i$ 保存该用户对应的私钥 $sk_i = y_i$, 公钥 $pk_i = y_i \cdot Q$; 同时, 为该用户生成并保存一个 *requests* 变量, 用于记录该用户的最近访问记录.

(5) $\mathcal{U}$ 将 s 安全地分布在 n 个密钥服务器后删除 $s, f(x), sk_i, 1 \leq i \leq n$, n 个密钥服务器共享的私钥 $sk = s$, 公钥 $pk = s \cdot Q$.

用户成功执行算法 Share 后, 注册阶段完成.

4.3 认证阶段

在认证阶段, 用户向认证服务器进行认证之前, 首先需要操作客户端输入图形口令. 具体步骤如下所示.

(1) 定位 P_g 的坐标. 在认证阶段, 图案界面上仍然有一个图案集合 $S_{a,p} = \{P_{a,1}, P_{a,2}, \dots, P_{a,h}\}$, h 为界面上图案的总个数. $S_{a,p}$ 排列为 u 行 v 列, 因此满足 $h = u \cdot v$, 且每一个图案对应一个坐标. 用户需要定位出图形口令 P_g 在 $S_{a,p}$ 中的位置, 然后确定出其坐标 (x_1, y_1) 和 (x_2, y_2) .

(2) 滑动坐标栏. 在定位出 P_g 后, 用户需要滑动 S_q 所对应的坐标栏, 并使 S_q 中两个方块对应的坐标分为 (x_1, y_1) 和 (x_2, y_2) .

同样地, 本文使用一个实例来描述认证阶段用户 $\mathcal{U}$ 在客户端的操作过程, 如图 3 所示. 实例中取 $u = 11, v = 7$.

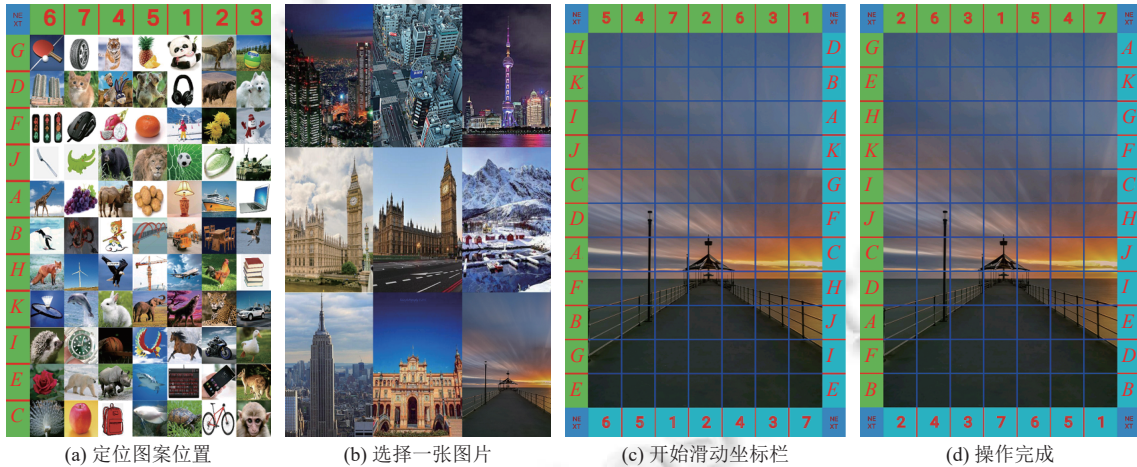


图 3 登录阶段

(1) $\mathcal{U}$ 准确地输入自己的 ID , 然后点击“NEXT” (此步骤未在图 3 中展示).

(2) 客户端随机在图案集合 Set_g 中选择 77 张图案, 并将这些图案以 11 行 7 列的形式显示在屏幕上, 如图 3(a) 所示. 如果 $\mathcal{U}$ 没有在屏幕上同时找到自己所选的两个图案, 可以返回重试, 直到屏幕上出现了自己所选择的两个图案为止. 在这个界面中, 除了随机选择的图案, 屏幕的上方和左侧还各有一个不可滑动的坐标栏, 用以表示每一个图案的坐标, 其中左侧坐标范围为 (A-J), 上方坐标范围为 (1-7). 每一次生成该界面的时候, 两个坐标栏的坐标顺序均随机打乱. $\mathcal{U}$ 需要在这个界面找到自己图形口令中 P_g 所对应的坐标, 并记住它. $\mathcal{U}$ 注册时选择的图案为 “apple” 和 “bag”, 此时它们的坐标为 (C,7) 和 (C,4), 因此他需要记住 (C,7) 和 (C,4). 此步骤不需要 $\mathcal{U}$ 额外的操作. $\mathcal{U}$ 点击“NEXT”按钮.

(3) 客户端在屏幕上展示一系列的图片, $\mathcal{U}$ 可以选择任意一张图片, 如图 3(b) 所示 (此处 $\mathcal{U}$ 可以选择自己注册时所选择的图片, 这样有利于用户回忆自己注册所选择方块的位置). $\mathcal{U}$ 点击该图片进入到下一界面.

(4) 客户端将 $\mathcal{U}$ 选择的图片展开, 并将其拆分为 11×7 个方块. 这个界面存在两对坐标栏, 分别是左上和右下 (图 3 中分别以绿色和蓝色表示). 它们和图 3(a) 中的坐标栏有一个不同之处在于它们可以滑动. $\mathcal{U}$ 需要滑动两对坐标栏使得自己图形口令中的两个方块的坐标正好分别对应 P_g 中两个图案的坐标. 由于第 1 个方块为 (7,7), 而此时该方块对应的坐标为 (A,1) (第 1 个方块对应左上方绿色左边栏, 第 2 个方块对应右下方蓝色坐标栏, 此时第 1 个方块的坐标为 (A,1)), 因此 $\mathcal{U}$ 需要将该方块的坐标从 (A,1) 滑动至 (C,7). 同样地, $\mathcal{U}$ 需要将第 2 个方块的坐标从 (G,5) 滑动至 (C,4) (对应图 3 中右下坐标栏). 图 3(d) 为滑动后的结果.

接下来客户端根据 $\mathcal{U}$ 的操作确定出用户的图形口令. 如果客户端没有保存 P_g ($\mathcal{U}$ 的图形口令的第 1 部分), 那么用户需要先向客户端指定 P_g . 之后, 客户端根据图 3(a) 定位出 P_g 第 1 个图案的坐标为 (C,7), 然后根据用户的最终滑动结果定位出 (C,7) 的位置. 如图 3(d) 所示, (C,7) 为第 7 行第 7 列, 因此第 1 个方块对应为 (7,7). 同样地, 根据 P_g 第 2 个图案的坐标 (C,4) 可以定位出第 2 个方块的位置为 (5,2). 因此 $S_q = ((7,7)(5,2))$, 进而得出 $gpw = ((apple, bag), ((7,7)(5,2)))$.

在图形界面操作完成之后, 为了完成身份认证, $\mathcal{U}$ 首先需要利用算法 Recovery 恢复出 BLS 签名. 在这之前, 先介绍算法 Check.

算法 Check: 密钥服务器利用本地维护的用户请求记录 *requests* 校验用户的请求频率是否超过限定. 如果超过限定, 则拒绝该用户的此次请求, 否则允许用户的此次请求, 并更新 *requests*. 该算法可以防止攻击者利用密钥服务器不停地猜测用户的图形口令. 校验请求频率的算法有很多, 例如滑动窗口算法. 本文将该算法当作一个黑盒算法, 不对该算法做过多描述.

算法 Recovery: 用户以门限盲签名方式恢复出密钥 s 对图形口令的 BLS 签名 σ . 具体步骤如下.

- (1) $\mathcal{U}$ 选择一个随机数 $r \in \mathbb{Z}_p^*$, 计算 $\mu = r \cdot H_1(gpw)$, $\mathcal{U}$ 将 (ID, μ) 发送给 n 个密钥服务器 $\mathcal{KS}_i$, 其中 $1 \leq i \leq n$.
- (2) 对于每一个密钥服务器 $\mathcal{KS}_i$, 它根据收到的 ID 查询本地数据库是否存在对应的 (x_i, y_i) . 如果没有查询到, 则给 $\mathcal{U}$ 发送“fail”消息并退出. 反之, $\mathcal{KS}_i$ 取出该用户的 *requests* 变量, 并利用 Check 算法校验用户的频度是否超过限定. 如果超过, 则 $\mathcal{KS}_i$ 给 $\mathcal{U}$ 发送“fail”消息并退出, 否则, $\mathcal{KS}_i$ 继续执行.
- (3) $\mathcal{KS}_i$ 计算 $\sigma_{r_i} = y_i \cdot \mu$, 并将 $(x_i, \sigma_{r_i}, pk_i)$ 发送给 $\mathcal{U}$, 其中 $pk_i = y_i \cdot Q$ 是该密钥服务器的公钥.
- (4) $\mathcal{U}$ 收到 σ_{r_i} 后验证 $e(\mu, pk_i) = e(\sigma_{r_i}, Q)$, 如果验证不成立, 则拒绝 σ_{r_i} , 否则保存 σ_{r_i} .
- (5) $\mathcal{U}$ 收集到任意 t 个正确的签名后 (不失一般性, 分别用 $\sigma_{rk_1}, \sigma_{rk_2}, \dots, \sigma_{rk_t}$ 表示, 其中 $rk_i \in \{1, 2, \dots, n\}, 1 \leq i \leq t$), 按照公式 (3) 和公式 (4) 计算:

$$\eta_i = \prod_{\substack{j=1 \\ j \neq i}}^t \frac{x_{k_j}}{x_{k_j} - x_{k_i}} \quad (3)$$

$$\sigma = r^{-1} \cdot \sum_{i=1}^t (\eta_i \cdot \sigma_{rk_i}) \quad (4)$$

并通过 $e(\sigma, Q) = e(H_1(gpw), pk)$ 验证 σ 的正确性, 其中 $pk = s \cdot Q$ 为 n 个密钥服务器共享的公钥.

$\mathcal{U}$ 得到 BLS 签名后可以利用算法 Authenticate 向服务器认证自己的身份.

算法 Authenticate: 用户利用得到的 BLS 签名生成凭证, 并向认证服务器认证身份. 具体步骤如下.

- (1) $\mathcal{U}$ 计算 $cred^*$:

$$cred^* = H_2(gpw || \sigma) \quad (5)$$

并通过安全信道将 $(ID, cred^*)$ 安全地发送给 $\mathcal{AS}$.

(2) $\mathcal{AS}$ 从安全信道中提取 ID 和 $cred^*$, 然后从数据库中查询该用户对应的 $cred$, 如果不存在, 则拒绝用户登录, 反之, 取出该用户对应的计数器 *count*, 并校验 $count \leq Max_{req}$ 是否成立, 如果不成立, 则拒绝用户登录, 否则, 继续执行.

(3) $\mathcal{AS}$ 校验 $H_2(cred^*) = cred$ 是否成立, 如果等式不成立, 则 $\mathcal{AS}$ 拒绝其登录, 执行 $count = count + 1$, 否则 $\mathcal{AS}$ 允许 $\mathcal{U}$ 登录, 执行 $count = 0$.

4.4 正确性证明

在 Recovery 算法的第 5 步中, 假定 $\mathcal{U}$ 得到 t 个正确的签名 $\{\sigma_{rk_1}, \sigma_{rk_2}, \dots, \sigma_{rk_t}\}$, 其中 $\sigma_{rk_i} = y_{k_i} \cdot H_1(gpw)$, $1 \leq i \leq t$, 签名私钥 s 在注册阶段共享在 n 个密钥服务器中, 由拉格朗日插值公式可知密钥服务器组共享的私钥可以按照如下方式恢复:

$$\eta_i = \prod_{\substack{j=1 \\ j \neq i}}^t \frac{x_{k_j}}{x_{k_j} - x_{k_i}} \quad (6)$$

$$s = \sum_{i=1}^t (\eta_i \cdot y_{k_i}) \quad (7)$$

客户端不恢复私钥 s , 但可以恢复私钥对应的 BLS 签名:

$$\sigma = r^{-1} \cdot \sum_{i=1}^t \eta_i \cdot \sigma_{rk_i} = r^{-1} \cdot \sum_{i=1}^t \eta_i \cdot r \cdot y_{k_i} \cdot H_1(gpw) = \sum_{i=1}^t \eta_i \cdot y_{k_i} \cdot H_1(gpw) = s \cdot H_1(gpw) \quad (8)$$

因此可以通过

$$\sigma = r^{-1} \cdot \sum_{i=1}^t \eta_i \cdot \sigma_{rk_i} \quad (9)$$

恢复出 BLS 签名 $s \cdot H_1(gpw)$, 且可通过

$$e(\sigma, Q) = e(H_1(gpw), pk) \quad (10)$$

验证 BLS 签名的正确性.

5 安全性分析

本文假定用户在注册阶段是安全的, 即用户操作图形界面的过程不会被攻击者偷窥, 并且使用安全信道进行通信. 在用户登录阶段, 用户操作客户端的过程可能被攻击者偷窥. 并且用户与密钥服务器的通信消息在公开网络上传输, 因此这些信息可能会被攻击者获取. 此外, 服务器凭证数据库也可能泄露. 在第 2.2 节本文考虑了 4 种攻击, 接下来本节逐一分析这些攻击.

• 抗肩窥攻击. GADL 方案可以抵抗肩窥攻击, 原因如下.

用户在图形界面的操作仅有一个, 即滑动图形界面上的坐标栏. 无论用户的图形口令是哪一个, 或者哪一些, 用户仅滑动了坐标栏, 而没有直接点击口令, 而滑动的操作都是类似的. 很显然, 如果用户直接点击口令, 那么攻击者通过偷窥可以得到用户图形口令. 通过滑动侧边坐标栏的方法代替直接点击图形口令可以有效抵抗偷窥攻击. 站在攻击者的角度来看, 用户滑动坐标栏的操作类似于随机滑动.

假定用户的图形口令一共包含 $2 \times \omega$ 对 (每一对口令包含一个图案和方块, 用户一次可以使用两对口令, 可以重复执行 ω 次操作以选择 ω 组图形口令). 一般情况下, 一对口令的口令空间包含 $z \times 77$ 个口令, 其中 z 表示图案集合中图案的总个数, 那么 2ω 对的口令空间集合的个数为 $(z \times 77)^{2\omega}$. 如果攻击者 $\mathcal{A}$ 偷窥了用户的一次登录过程, 那么 $\mathcal{A}$ 可以将用户的一对口令空间集合缩小为 77 个, 则 2ω 对图形口令的口令空间集合缩小至 $77^{2\omega}$ 个. 因此, $\mathcal{A}$ 即使偷窥了用户的一次登录过程, 依然无法从中得到用户的图形口令, 而只能缩小用户的图形口令空间.

此外, $\mathcal{A}$ 也不能模仿用户上一次的操作过程来冒充用户登录. 在每一次登录时, 图形界面上所有图案的位置、顺序以及坐标栏中的坐标顺序等均随机生成, 因此两次登录过程也完全不同.

• 抗字典猜测攻击. GADL 方案可以抵抗在线字典猜测攻击, 原因如下.

由于图形口令的空间较小, 攻击者可能会通过字典猜测攻击获取用户的口令. 具体来说, 假设攻击者拥有一个图形口令集合, 记为 $gpws = \{gpw_1, gpw_2, gpw_3, \dots\}$, 同时有一个用户 ID 集合, 记为 $IDS = \{ID_1, ID_2, ID_3, \dots\}$. 攻击者

首先从 $gpws$ 中选择一个图形口令, 然后从 IDs 中选择一个用户 ID , 使用该 ID 和 gpw 去尝试登录服务器. 攻击者如果登录成功, 就成功地猜测出该用户 ID 对应的口令. 如果登录失败, 攻击者从 IDs 中选择下一个用户 ID , 继续猜测. 当所有的 ID 都猜测完毕, 攻击者从 $gpws$ 中选择下一个 gpw , 重复上述步骤.

为了方便解释上述攻击, 本文首先给出如下定义.

定义 2. 如果称 GADL 可以抵抗在线口令猜测攻击, 那么对于一个 PPT 敌手 $\mathcal{A}$, 口令空间为 P , 存在一个可忽略函数 $negl$, 满足:

$$\Pr \leq \frac{\text{Max}(\text{Max}_{\mathcal{A}_t}(\mathcal{KS}_1, \mathcal{KS}_2, \dots, \mathcal{KS}_n), \text{Max}_{req})}{|P|} + \text{negl}(\lambda) \quad (11)$$

其中, $\Pr$ 表示攻击者 $\mathcal{A}$ 猜中用户口令的概率, $\text{Max}()$ 表示求多个输入的最大值, $\text{Max}_{\mathcal{A}_t}(\mathcal{KS}_1, \mathcal{KS}_2, \dots, \mathcal{KS}_n)$ 表示在密钥服务器群组限定次数内攻击者猜测次数的上界, Max_{req} 表示认证服务器允许该用户名的连续错误认证次数. 为了更加直观, 考虑如下攻击.

(1) 攻击者针对某一个目标用户随机选择一个用户图形口令 gpw .

(2) 攻击者以门限的方式同 n 个密钥服务器进行交互, 然后按照正常步骤计算出 $cred^*$ 并发送给认证服务器验证, 如果验证成功, 则攻击者猜测成功, 否则攻击者猜测失败.

(3) 攻击者在服务器允许的次数内限定的次数内重复执行步骤 (1) 和 (2), 直到猜测成功.

攻击者通过上述攻击成功猜测出用户图形口令的概率不会大于 $\frac{\text{Max}(\text{Max}_{\mathcal{A}_t}(\mathcal{KS}_1, \mathcal{KS}_2, \dots, \mathcal{KS}_n), \text{Max}_{req})}{|P|} + \text{negl}(\lambda)$. 在字典猜测攻击中, 攻击者针对一系列账号进行不断地尝试, 但实际上, 对每一个目标账号, 攻击者成功猜测的概率不会大于 $\frac{\text{Max}(\text{Max}_{\mathcal{A}_t}(\mathcal{KS}_1, \mathcal{KS}_2, \dots, \mathcal{KS}_n), \text{Max}_{req})}{|P|} + \text{negl}(\lambda)$. 因此, 本方案可以抵抗在线字典猜测攻击.

• 抗监听攻击. GADL 方案可以抵抗监听攻击, 原因如下.

由于用户和认证服务器之间采用安全信道进行通信, 因此攻击者只能监听用户和密钥服务器之间的通信消息. 攻击者可得到的消息包含 $\mu = r \cdot H_1(gpw)$ 和 $\sigma_{rk_i} = r \cdot y_{k_i} \cdot H_1(gpw)$, $1 \leq i \leq t$, 它们均和随机数 r 绑定在一起. 比如 $\mu = r \cdot H_1(gpw)$, 由于离散对数的难解性, 攻击者在不知道 r 的情况下无法通过暴力穷举口令空间的方式来得到任何有关 gpw 的有用信息.

• 抗凭证泄露攻击. 凭证泄露攻击有两种具体攻击方法, 首先是凭证伪造攻击, 其次是图形口令猜测攻击.

GADL 方案可以抵御凭证伪造攻击, 原因如下.

回顾算法 Authenticate, 如果 $\mathcal{A}$ 意图冒充 $\mathcal{U}$ 登录到系统中, 那么 $\mathcal{A}$ 需要提交给认证服务器的信息为 $cred^*$, 使得 $H_2(cred^*) = cred$, 其中 $cred$ 为 $\mathcal{U}$ 保存在认证服务器内部的凭证. 假设 $\mathcal{A}$ 已经成功攻击了服务器的数据库, 得到了 $\mathcal{U}$ 保存在认证服务器内部的登录凭证 $cred$, $\mathcal{A}$ 的目的是找到 $cred^*$.

为了证明 $\mathcal{A}$ 不能从中恢复出 $cred$ 的原像, 本文给出引理 1 和引理 2 (此处参照了文献 [5] 中证明方法).

引理 1. 给定一个用户 $\mathcal{U}$ 的凭证 $cred$ 及其图形口令 gpw , PPT 敌手 $\mathcal{A}$ 即使与 t' ($t' < t$) 个密钥服务器合谋也不能伪造出正确的签名, 即存在可忽略函数 $negl$, 满足:

$$\Pr_{\mathcal{A}, \lambda}^{\text{Predict}} \leq \text{negl}(\lambda) \quad (12)$$

其中, Predict 是本文定义的一个游戏, $\Pr_{\mathcal{A}, \lambda}^{\text{Predict}}$ 表示 $\mathcal{A}$ 赢得游戏的概率.

证明: Predict 的游戏定义如下.

- (1) 执行环境 $\mathcal{S}$ 进行初始化, 确定公开参数 PP , $\mathcal{S}$ 将 PP 发送给 $\mathcal{A}$.
- (2) $\mathcal{S}$ 随机选取口令 gpw 和密钥 s , 计算 $\sigma = s \cdot H_1(gpw)$, $cred^* = H_2(gpw || \sigma)$, 然后将 gpw 和 $H(cred^*)$ 发送给 $\mathcal{A}$; $\mathcal{S}$ 将 s 以 (t, n) -秘密共享的方式拆分为 n 份, 记为 $\{s_1, s_2, \dots, s_n\}$; $\mathcal{S}$ 将 $H(cred^*)$ 和 $\{s_1, s_2, \dots, s_n\}$ 保存在本地.
- (3) $\mathcal{A}$ 从 $[1, n]$ 中选取 t' 个索引, 分别记为 $\{idx_1, idx_2, \dots, idx_{t'}\}$, 然后将其发送给 $\mathcal{S}$.
- (4) $\mathcal{S}$ 取出对应索引的子密钥, 将其发送 $\mathcal{A}$.
- (5) $\mathcal{S}$ 选定一个随机数 x , 将其发送给 $\mathcal{A}$.
- (6) $\mathcal{A}$ 产生 x 的 BLS 签名 σ_x , 然后将其发送给 $\mathcal{S}$.

(7) $\mathcal{S}$ 校验 σ_x 的正确性, 如果正确, 则输出 1, $\mathcal{A}$ 赢得游戏, 否则输出 0, $\mathcal{A}$ 游戏失败.

上述 $\mathcal{A}$ 赢得游戏的概率为 $\Pr_{\mathcal{A}, \lambda}^{\text{Predict}}$. $\mathcal{A}$ 赢得游戏的关键挑战是伪造 BLS 签名 σ_x . 如果 $\mathcal{A}$ 能够以 $\Pr_{\mathcal{A}, \lambda}^{\text{Predict}}$ 的概率获取胜利, $\mathcal{A}$ 就能够以相同的概率伪造 BLS 签名. 而 BLS 签名具有不可伪造性, 因此 $\mathcal{A}$ 游戏胜利的概率是可忽略的, 因而有 $\Pr_{\mathcal{A}, \lambda}^{\text{Predict}} \leq \text{negl}(\lambda)$.

证毕.

引理 2. 给定一个用户 $\mathcal{U}$ 的凭证 cred , PPT 敌手 $\mathcal{A}$ 与 t' ($t' < t$) 个密钥服务器合谋伪造出一个可以通过验证的凭证 cred^* 的概率满足:

$$\Pr_{\mathcal{A}, \lambda}^{\text{Forge}} \leq \frac{\text{Max}(\text{Max}_{\mathcal{A}, t'}(\mathcal{KS}_1, \mathcal{KS}_2, \dots, \mathcal{KS}_n), \text{Max}_{\text{req}})}{|P|} + \text{negl}(\lambda) \quad (13)$$

其中, negl 代表一个可忽略的函数, $|P|$ 表示用户的图形口令空间大小, $\text{Max}_{\mathcal{A}, t'}(\mathcal{KS}_1, \mathcal{KS}_2, \dots, \mathcal{KS}_n)$ 表示 $\mathcal{A}$ 在密钥服务器限制次数内能够猜测的口令数的上界, Max_{req} 表示认证服务器允许该用户名的连续错误认证次数, $\Pr_{\mathcal{A}, \lambda}^{\text{Forge}}$ 是攻击者找出 cred^* 的概率.

证明: 综合分析, $\mathcal{A}$ 找出 cred^* 的方法有如下几种.

- (1) 穷举哈希原像.
- (2) 在线猜测用户图形口令.
- (3) 获取服务器端密钥, 进一步暴力破解用户图形口令, 然后再离线计算 cred^* .

方法 (1) 是最简单的方法. 由于 $H_2()$ 是一个安全的哈希函数 (预备知识中已经定义), 因此通过这种方法找到满足条件的原像的概率是可忽略的.

方法 (2) 的证明与前文抵抗在线字典猜测攻击的证明类似. 攻击者针对目标用户随机选择口令, 然后与密钥服务器交互恢复出 cred^* , 然而在密钥服务器和认证服务器限定次数内攻击者猜中用户图形口令的概率始终不会大于 $\frac{\text{Max}(\text{Max}_{\mathcal{A}, t'}(\mathcal{KS}_1, \mathcal{KS}_2, \dots, \mathcal{KS}_n), \text{Max}_{\text{req}})}{|P|} + \text{negl}(\lambda)$.

对于方法 (3), 如果攻击者能够以不可忽略的概率获取密钥服务器群组共享的密钥, 那么攻击者可以以相同的概率伪造对 g_{pw} 的签名. 根据引理 1 可知, 攻击者通过 cred 及对应的图形口令 g_{pw} 伪造出正确的签名的概率是可忽略的, 因此攻击者获取密钥服务器群组共享的密钥的概率是可忽略的, 进而通过暴力猜测的方式恢复用户的图形口令是可忽略的.

综上所述, 攻击者伪造出 cred^* 使得 $H(\text{cred}^*) = \text{cred}$ 的概率始终不大于 $\frac{\text{Max}(\text{Max}_{\mathcal{A}, t'}(\mathcal{KS}_1, \mathcal{KS}_2, \dots, \mathcal{KS}_n), \text{Max}_{\text{req}})}{|P|} + \text{negl}(\lambda)$.

证毕.

由引理 1 和引理 2 可知攻击者不能以大于 $\frac{\text{Max}(\text{Max}_{\mathcal{A}, t'}(\mathcal{KS}_1, \mathcal{KS}_2, \dots, \mathcal{KS}_n), \text{Max}_{\text{req}})}{|P|} + \text{negl}(\lambda)$ 的概率伪造出可通过服务器验证的 cred^* , 因此 GADL 方案可以抵抗凭证伪造攻击.

GADL 方案可以抵抗图形口令猜测攻击, 原因已经在上述引理 2 中说明, 攻击者不能以大于 $\frac{\text{Max}(\text{Max}_{\mathcal{A}, t'}(\mathcal{KS}_1, \mathcal{KS}_2, \dots, \mathcal{KS}_n), \text{Max}_{\text{req}})}{|P|} + \text{negl}(\lambda)$ 的概率通过在线口令猜测和离线口令猜测两种方式恢复用户的图形口令.

6 性能评估

我们基于安卓平台实现了 GADL 方案, 并从计算开销、通信开销和存储开销 3 个指标评估 GADL 方案的性能表现.

6.1 实 现

本文使用 Java 语言在安卓平台上实现了 GADL 方案, 使用的密码学库为 JPBC. 认证服务器和密钥服务器的硬件参数均为 Intel(R) Core(TM) i7-9700 CPU、32 GB RAM, 操作系统均为 Ubuntu 16.0.4. 为了测试移动端性能,

实验选择的是安卓手机, 其型号为 OPPO R5, CPU 频率为 1.5 GHz, RAM 为 2 GB, 操作系统为 Android 6.0. 实验采用的双线性映射曲线为 MNT 曲线, 安全参数为 128 位. 数据库一共包含 143 张可以供用户选择的图案, 这些图案包括动物、数码设备、水果等. 客户端同时保存了 9 张图片供用户选择, 也允许用户上传自己的图片到本地数据库中.

图 4 是 GADL 方案主要实现的函数. 黑色方框是用户执行步骤, 黄色方框是密钥服务器执行的步骤, 蓝色方框是认证服务器执行的步骤. 由于整个系统的初始化过程只执行了一次, 且在系统第 1 次运行执行初始化函数, 确定出系统参数 PP , 因此此处没有将该过程展示在图 4 中. 为了更好地解释图中函数, 本文简要地描述用户、密钥服务器和认证服务器 3 个实体在注册和登录阶段的执行过程.

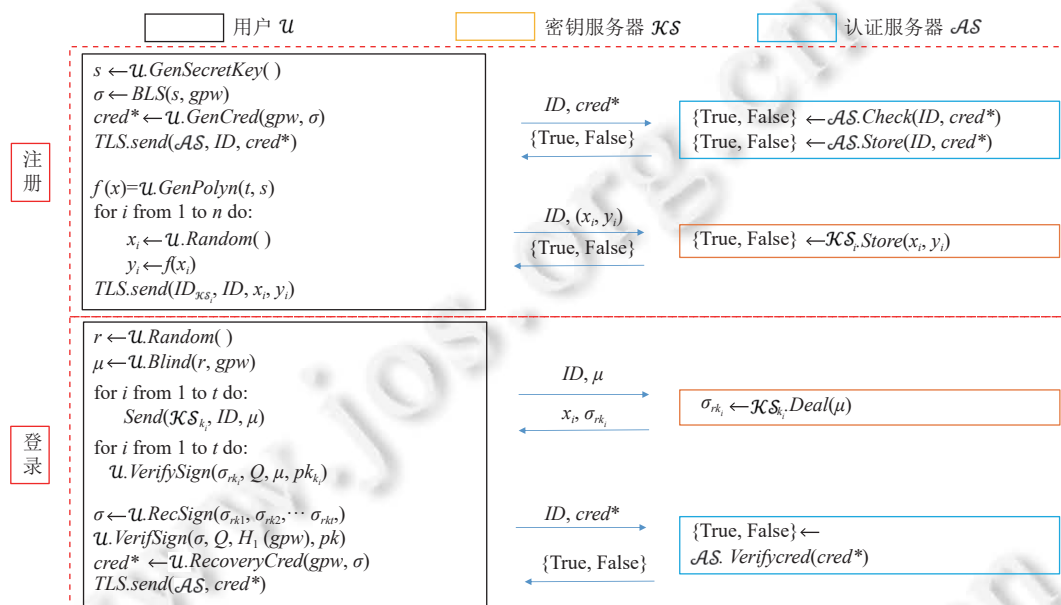


图 4 方案实现

注册阶段, 用户 $\mathcal{U}$ 选择的身份为 ID , 选择的图形口令为 gpw . $\mathcal{U}$ 选择完毕后, 调用 $GenSecretKey()$ 生成私钥 s , 并利用 s 通过 $BLS()$ 函数计算 BLS 签名 σ . $\mathcal{U}$ 利用 $GenCred()$ 函数生成认证凭证 $cred^*$. $\mathcal{U}$ 与 AS 建立 TLS 安全信道, 并将 ID 和 $cred^*$ 等信息发送给 AS . AS 收到消息后调用 $Check()$ 函数验证用户的信息是否正确合法. 验证无误后, AS 调用 $Store()$ 函数将凭证的哈希值存储在本地. 之后 $\mathcal{U}$ 使用 $GenPolyn()$ 函数生成一个随机多项式 $f(x)$, 使用 $Random()$ 函数生成 n 个随机数 x_i ($1 \leq i \leq n$), 并计算 $y_i = f(x_i)$, 然后与 n 个密钥服务器 KS_i ($1 \leq i \leq n$) 分别建立安全信道, 将 $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ 分别发送给 n 个密钥服务器保存.

认证阶段, 用户 $\mathcal{U}$ 的身份为 ID , 图形口令为 gpw . $\mathcal{U}$ 在图形界面完成操作之后, 调用 $Random()$ 函数生成一个随机数 r , 并使用 $Blind()$ 函数将 r 和 gpw 绑定得到 μ . $\mathcal{U}$ 将 μ 发送给任意 t 个密钥服务器 $KS_{k_i}, k_i \in \{1, 2, \dots, n\}, 1 \leq i \leq t$. KS_{k_i} 收到消息后调用 $Deal()$ 函数对 μ 处理得到 σ_{rk_i} , 并将其发给 $\mathcal{U}$. $\mathcal{U}$ 收到消息后通过 $VerifySign()$ 验证 σ_{rk_i} 的正确性. 在收到 t 个正确的签名后, $\mathcal{U}$ 利用 $RecSign()$ 恢复出 BLS 签名 σ , 并通过 $VerifySign()$ 验证 σ 的正确性. 在确认 σ 正确的情况下, $\mathcal{U}$ 调用 $RecoveryCred()$ 计算凭证 $cred^*$, 然后通过 TLS 安全信道将其发送给服务器 AS . AS 得到 $cred^*$ 后调用 $VerifyCred()$ 函数验证凭证是否正确. 如果正确, 则 $\mathcal{U}$ 登录成功, 否则登录失败.

本文在实现的时候做了优化, 即在恢复签名时, 用户不需要向所有密钥服务器请求签名, 而是向其中 t 个密钥服务器发起请求, 如果发现签名错误或者密钥服务器无法连接等问题, 才向其他密钥服务器发起请求.

6.2 计算开销

表 2 展示了 GADL 中 3 类实体执行的重要操作.

表 2 密码学操作符号

符号	操作	符号	操作
$Pair_G$	双线性映射 e	Mul_G	群 G 的乘法运算
Add_G	群 G 的加法运算	Mul_{Z_p}	Z_p 上的乘法运算
$Hash_G$	将给定值映射到群 G 中	$Hash_{Z_p}$	将给定值映射到 Z_p 中

注册阶段,用户在选定用户名和图形口令后,需要随机生成一个私钥,并利用此私钥计算 BLS 签名.随后用户利用签名计算 $cred^*$,将私钥拆分为 n 个子秘密并分别发送给 n 个密钥服务器,因此总的执行操作为: $1 \times Hash_G + 1 \times Mul_G + 1 \times Hash_{Z_p} + n \times Mul_{Z_p}$,其中 n 为密钥服务器的总数.

认证阶段,在优化后,用户首先需要和 t 个密钥服务器进行交互,恢复正确的签名,然后利用该签名计算登录凭证,总共需要执行的操作为: $1 \times Hash_G + (1+t) \times Mul_G + (1+t) \times Pair_G + 1 \times Hash_{Z_p} + 1 \times Mul_{Z_p}$,其中 t 为门限.值得注意的是,增大 t ,会提升用户账户的安全性,但也会同时增加客户端的计算量.

n 个密钥服务器在注册阶段没有额外的计算开销,在认证阶段则只需要对客户消息进行一次计算,因此 n 个密钥服务器总的操作次数为: $t \times Mul_G$ (正常情况下,只有其中 t 个密钥服务器做了计算).

认证服务器在注册阶段需要计算用户凭证,需要一次哈希运算: $1 \times Hash_{Z_p}$,在认证阶段需要根据本地保存的用户凭证验证用户发来的凭证,因此需要执行一次哈希运算: $1 \times Hash_{Z_p}$.

分析发现 GADL 方案给客户端带来的计算量稍大,这是因为大多数的计算和校验步骤都在客户端进行,而没有在服务器端进行.这些开销是可以接受的,因为计算量增加带来的是更高的安全性,将重要的步骤放在客户端中计算,避免服务器获得有关用户图形口令的重要信息,可以更好地保护用户图形口令.并且,考虑到现有的智能手机等移动设备的算力,这些计算量并不会给这些设备带来较大的负担.综上所述,引入密钥服务器协助用户在每一次登录的时候恢复认证凭证所带来的计算开销是可以接受的.

实验结果如图 5 所示,图中描述了客户端注册和登录所需要的时间,其中横轴表示密钥服务器的数量,且满足 $t = n$,纵轴表示时延.从图中可以看出,客户端可以在较短时间内完成注册和登录两个步骤.

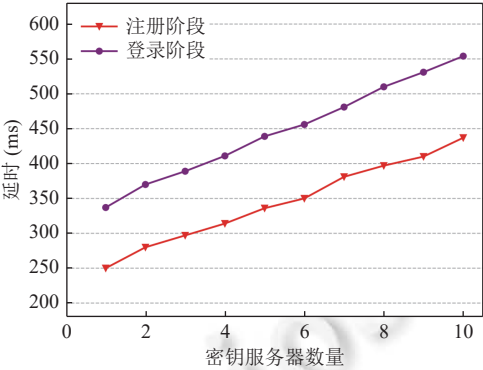


图 5 客户端计算时延

6.3 通信开销

实验中计算凭证使用的哈希函数为 SHA-256,凭证的长度为 32 B,此外,用户的 ID 长度为 20 B.在注册阶段,用户不仅需要给认证服务器发送凭证,还需要给所有的密钥服务器发送子密钥,随着密钥服务器的数量增加,通信开销增加.同理,每一个密钥服务器需要接收来自用户的子密钥,因此总的通信开销也在增加.认证服务器仅需要接收用户发送的登录凭证,与密钥服务器的数量无关,因此通信开销保持稳定.图 6(a) 表示在注册阶段 3 类实体通信开销,其中横轴表示密钥服务器的总数.

在注册阶段,用户需要利用自己的口令和密钥服务器交互恢复出 $cred^*$,然后将 $cred^*$ 发送给认证服务器.每一

个密钥服务器接收用户的信息并发送一个数字签名结果. 认证服务器仅需要接收用户发来的凭证. 图 6(b) 表示在密钥服务器总数为 10 的情况下登录阶段 3 类实体的通信开销, 其中横轴表示门限 t . 此阶段的结果同图 6(a) 相似. 用户需要以 (t, n) -门限的方式恢复 BLS 签名. t 越大, 用户的通信开销越大, 密钥服务器总的通信开销也越大. 认证服务器仅和用户进行交互, 与密钥服务器的数量无关, 因此认证服务器的通信开销稳定.

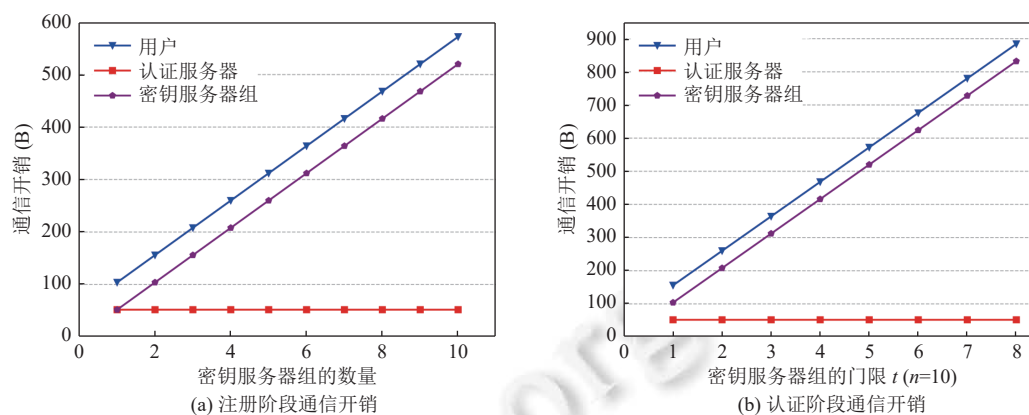


图 6 注册阶段和认证阶段通信开销

图 7 描述了 GADL 方案与 PROTECT 方案^[5]和哈希加盐法的对比情况. 该图横坐标是密钥服务器的门限值, 其中密钥服务器总数为 20, 纵坐标表示用户在认证阶段的通信开销. 从图中可以看出, 哈希加盐法的通信数据量与密钥服务器数量无关, 而 GADL 方案比 PROTECT 方案的通信开销小, 这是由于 PROTECT 方案需要用户与密钥服务器群组交互 4 轮, 而 GADL 方案仅需要和密钥服务器交互 2 轮. 整体来说, GADL 方案给用户端带来的开销较小, 对用户端而言是可以接受的.

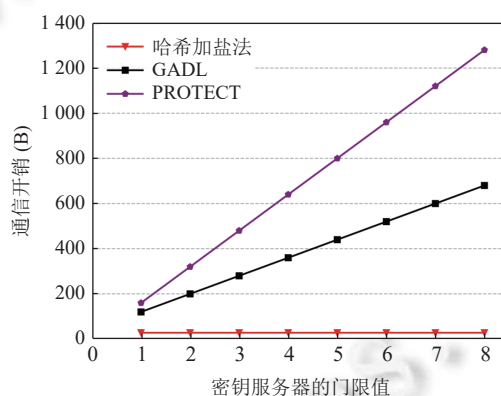


图 7 通信开销对比

6.4 存储开销

用户端需要保存一些公共的图片资源, 以供注册和登录时选择. 为了方便使用, 客户端可以保存用户图形口令的第 1 部分, 如果客户端不保存, 则需要用户登录的时候手动指定. 用户仅需要记住自己的账户 ID 和图形口令 gpw , 而不需要记住额外的信息. 也正是因为客户端可以不保存与用户口令有关的信息, 使得用户可以在任何一个设备上完成身份认证.

为了在用户登录时恢复凭证, 每一个密钥服务器均需要保存用户在注册阶段发送来的用户 ID 和其对应的私钥. 认证服务器保存了用户的 ID 、凭证等信息以便验证用户身份.

图 8(a) 表示认证服务器和密钥服务器对应的存储开销, 其中密钥服务器的存储开销分为 3 种情况, 即 n 分别

为 5、10、15. 每一个密钥服务器均需要保存用户对应的子秘密, 因此密钥服务器的数量越多, 总的存储开销越大. 另外随着用户数量增加, 密钥服务器和认证服务器的存储开销均不断增大.

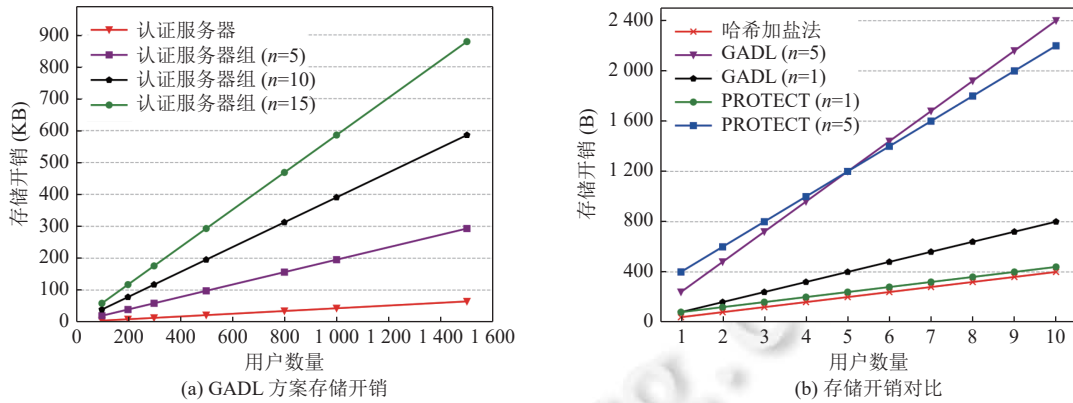


图 8 存储开销及对比

图 8(b) 是 GADL 方案与哈希加法以及 PROTECT 方案^[5]的存储开销对比, 可以看出哈希加法需要双方交互的信息量最少. 在密钥服务器总数为 1 的情况下, 哈希加法法和 PROTECT 所需存储量相近, GADL 方案所需存储量较大, 在密钥数量为 5 的时候, 随着用户数量增加, PROTECT 方案所需的存储开销大于 GADL 方案. 总体来说, GADL 方案所需的存储开销较小, 相比于哈希加法, 增加存储开销的同时一定程度上提高了安全性, 是可以接受的.

7 总结

本文提出的图形口令认证方案, 称作 GADL, 具有两个特点: 1) 可以解决服务器数据库泄露导致图形口令泄露的问题, 增强了图形口令认证方案的安全性; 2) 可以抵抗肩窥攻击, 使得攻击者在用户登录时偷窥用户操作过程也不能得到用户真正的图形口令. GADL 方案不需要额外的硬件或者辅助操作, 方便用户使用. 本文不仅给出了安全性分析来证明 GADL 方案可以抵抗肩窥攻击和服务端凭证泄露攻击, 而且通过实验部分表明 GADL 方案在通信、计算、存储方面的效率较高.

为了解决服务器数据库泄露导致的口令泄露问题, GADL 方案引入了多密钥服务器协助用户生成凭证. 密钥服务器在一定程度上带来了密钥管理的困难. 在今后的工作中, 我们将致力于找到一种方法在去掉密钥服务器的基础上依然可以达到我们所需要的安全性目标.

References:

- [1] Zhao P, Bian KG, Zhao T, Song XT, Park JM, Li XM, Ye F, Yan W. Understanding smartphone sensor and APP data for enhancing the security of secret questions. *IEEE Trans. on Mobile Computing*, 2017, 16(2): 552–565. [doi: 10.1109/TMC.2016.2546245]
- [2] Herley C, Van Oorschot P. A research agenda acknowledging the persistence of passwords. *IEEE Security & Privacy*, 2012, 10(1): 28–36. [doi: 10.1109/MSP.2011.150]
- [3] Wang D, Zou YK, Dong QY, Song YM, Huang XY. How to attack and generate honeywords. In: *Proc. of the 2022 IEEE Symp. on Security and Privacy*. San Francisco: IEEE, 2022. 966–983. [doi: 10.1109/SP46214.2022.9833598]
- [4] Wang D, Cheng HB, Wang P, Yan J, Huang XY. A security analysis of honeywords. In: *Proc. of the 2018 Network and Distributed Systems Security Symp.* San Diego: NDSS, 2018. 1–16. [doi: 10.14722/ndss.2018.23142]
- [5] Zhang Y, Xu CX, Li HW, Yang K, Cheng N, Shen XM. PROTECT: Efficient password-based threshold single-sign-on authentication for mobile users against perpetual leakage. *IEEE Trans. on Mobile Computing*, 2021, 20(6): 2297–2312. [doi: 10.1109/TMC.2020.2975792]
- [6] Wang D, Zhang ZJ, Wang P, Yan J, Huang XY. Targeted online password guessing: An underestimated threat. In: *Proc. of the 2016 ACM SIGSAC Conf. on Computer and Communications Security*. Vienna: ACM, 2016. 1242–1254. [doi: 10.1145/2976749.2978339]
- [7] Zhang Y, Xu CX, Cheng N, Shen XM. Secure password-protected encryption key for deduplicated cloud storage systems. *IEEE Trans. on Dependable and Secure Computing*, 2022, 19(4): 2789–2806. [doi: 10.1109/TDSC.2021.3074146]

- [8] Wang D, Wang P. Offline dictionary attack on password authentication schemes using smart cards. In: Desmedt Y, ed. Information Security. Cham: Springer, 2015. 221–237. [doi: [10.1007/978-3-319-27659-5_16](https://doi.org/10.1007/978-3-319-27659-5_16)]
- [9] Wash R, Rader E, Berman R, Wellmer Z. Understanding password choices: How frequently entered passwords are re-used across websites. In: Proc. of the 12th USENIX Conf. on Usable Privacy and Security. Denver: USENIX Association, 2016. 175–188.
- [10] Das A, Bonneau J, Caesar M, Borisov N, Wang XF. The tangled Web of password reuse. In: Proc. of the 2014 Network and Distributed System Security Symp. San Diego: Internet Society, 2014. 23–26. [doi: [10.14722/ndss.2014.23357](https://doi.org/10.14722/ndss.2014.23357)]
- [11] Paivio A, Rogers TB, Smythe PC. Why are pictures easier to recall than words? Psychonomic Science, 1968, 11(4): 137–138. [doi: [10.3758/BF03331011](https://doi.org/10.3758/BF03331011)]
- [12] Suo XY, Zhu Y and Owen GS. Graphical passwords: A survey. In: Proc. of the 21st Annual Computer Security Applications Conf. Tucson: IEEE, 2005. 10–472. [doi: [10.1109/CSAC.2005.27](https://doi.org/10.1109/CSAC.2005.27)]
- [13] Jain AK, Ross A, Prabhakar S. An introduction to biometric recognition. IEEE Trans. on Circuits and Systems for Video Technology, 2004, 14(1): 4–20. [doi: [10.1109/TCSVT.2003.818349](https://doi.org/10.1109/TCSVT.2003.818349)]
- [14] Tan DS, Keyani P, Czerwinski M. Spy-resistant keyboard: More secure password entry on public touch screen displays. In: Proc. of the 17th Australia Conf. on Computer-human Interaction: Citizens Online: Considerations for Today and the Future. Canberra: ACM, 2005. 1–10.
- [15] Zhao HY, Li XL. S3PAS: A scalable shoulder-surfing resistant textual-graphical password authentication scheme. In: Proc. of the 21st Int'l Conf. on Advanced Information Networking and Applications Workshops. Niagara Falls: IEEE, 2007. 467–472 [doi: [10.1109/AINAW.2007.317](https://doi.org/10.1109/AINAW.2007.317)]
- [16] MacRae B, Salehi-Abari A, Thorpe J. An exploration of geographic authentication schemes. IEEE Trans. on Information Forensics and Security, 2016, 11(9): 1997–2012. [doi: [10.1109/TIFS.2016.2570681](https://doi.org/10.1109/TIFS.2016.2570681)]
- [17] Meng WZ, Wong DS, Furnell S, Zhou JY. Surveying the development of biometric user authentication on mobile phones. IEEE Communications Surveys & Tutorials, 2015, 17(3): 1268–1293. [doi: [10.1109/COMST.2014.2386915](https://doi.org/10.1109/COMST.2014.2386915)]
- [18] Mahfouz A, Mahmoud TM, Eldin AS. A survey on behavioral biometric authentication on smartphones. Journal of Information Security and Applications, 2017, 37: 28–37. [doi: [10.1016/j.jisa.2017.10.002](https://doi.org/10.1016/j.jisa.2017.10.002)]
- [19] Mai GC, Cao K, Lan XY, Yuen PC. SecureFace: Face template protection. IEEE Trans. on Information Forensics and Security, 2021, 16: 262–277. [doi: [10.1109/TIFS.2020.3009590](https://doi.org/10.1109/TIFS.2020.3009590)]
- [20] Jiang CS, Xu CX, Zhang Y. PFLM: Privacy-preserving federated learning with membership proof. Information Sciences, 2021, 576: 288–311. [doi: [10.1016/j.ins.2021.05.077](https://doi.org/10.1016/j.ins.2021.05.077)]
- [21] De Luca A, Harbach M, von Zezschwitz E, Maurer ME, Slawilk BE, Hussmann H, Smith M. Now you see me, now you don't: Protecting smartphone authentication from shoulder surfers. In: Proc. of the 2014 SIGCHI Conf. on Human Factors in Computing Systems. Toronto: ACM, 2014. 2937–2946. [doi: [10.1145/2556288.2557097](https://doi.org/10.1145/2556288.2557097)]
- [22] Bianchi A, Oakley I, Kostakos V, Kwon DS. The phone lock: Audio and haptic shoulder-surfing resistant pin entry methods for mobile devices. In: Proc. of the 5th Int'l Conf. on Tangible, Embedded, and Embodied Interaction. Funchal: ACM, 2011. 197–200. [doi: [10.1145/1935701.1935740](https://doi.org/10.1145/1935701.1935740)]
- [23] von Zezschwitz E, Dunphy P, De Luca D. Patterns in the wild: A field study of the usability of pattern and pin-based authentication on mobile devices. In: Proc. of the 15th Int'l Conf. on Human-computer Interaction with Mobile Devices and Services. Munich: ACM, 2013. 261–270. [doi: [10.1145/2493190.2493231](https://doi.org/10.1145/2493190.2493231)]
- [24] Jermyn I, Mayer A, Monroe F, Reiter MK, Rubin AD. The design and analysis of graphical passwords. In: Proc. of the 8th Conf. on USENIX Security Symp. Washington: USENIX Association, 1999. 1.
- [25] Wiedenbeck S, Waters J, Birget JC, Brodskiy A, Memon N. PassPoints: Design and longitudinal evaluation of a graphical password system. Int'l Journal of Human-computer Studies, 2005, 63(1–2): 102–127. [doi: [10.1016/j.ijhcs.2005.04.010](https://doi.org/10.1016/j.ijhcs.2005.04.010)]
- [26] Golofit K. Click passwords under investigation. In: Proc. of the 12th European Symp. on Research in Computer Security. Dresden: Springer, 2007. 343–358. [doi: [10.1007/978-3-540-74835-9_23](https://doi.org/10.1007/978-3-540-74835-9_23)]
- [27] Chiasson S, van Oorschot PC, Biddle R. Graphical password authentication using cued click points. In: Proc. of the 12th European Symp. on Research in Computer Security. Dresden: Springer, 2007. 359–374. [doi: [10.1007/978-3-540-74835-9_24](https://doi.org/10.1007/978-3-540-74835-9_24)]
- [28] Chiasson S, Stobert E, Forget A, Biddle R, Van Oorschot PC. Persuasive cued click-points: Design, implementation, and evaluation of a knowledge-based authentication mechanism. IEEE Trans. on Dependable and Secure Computing, 2012, 9(2): 222–235. [doi: [10.1109/TDSC.2011.55](https://doi.org/10.1109/TDSC.2011.55)]
- [29] Wiedenbeck S, Waters J, Sobrado L, Birget JC. Design and evaluation of a shoulder-surfing resistant graphical password scheme. In: Proc. of the 2006 Working Conf. on Advanced Visual Interfaces. Venezia: ACM, 2006. 177–184. [doi: [10.1145/1133265.1133303](https://doi.org/10.1145/1133265.1133303)]

- [30] Sun HM, Chen ST, Yeh JH, Cheng CY. A shoulder surfing resistant graphical authentication system. *IEEE Trans. on Dependable and Secure Computing*, 2018, 15(2): 180–193. [doi: [10.1109/TDSC.2016.2539942](https://doi.org/10.1109/TDSC.2016.2539942)]
- [31] Chakraborty N, Randhawa GS, Das K, Mondal S. MobSecure: A shoulder surfing safe login approach implemented on mobile device. *Procedia Computer Science*, 2016, 93: 854–861. [doi: [10.1016/j.procs.2016.07.256](https://doi.org/10.1016/j.procs.2016.07.256)]
- [32] Fang LM, Li Y, Yun XY, Wen ZY, Ji SL, Meng WZ, Cao ZH, Tanveer M. THP: A novel authentication scheme to prevent multiple attacks in SDN-based IoT network. *IEEE Internet of Things Journal*, 2020, 7(7): 5745–5759. [doi: [10.1109/JIOT.2019.2944301](https://doi.org/10.1109/JIOT.2019.2944301)]
- [33] Yu XJ, Wang Z, Li YJ, Li L, Zhu WT, Song L. EvoPass: Evolvable graphical password against shoulder-surfing attacks. *Computers & Security*, 2017, 70: 179–198. [doi: [10.1016/j.cose.2017.05.006](https://doi.org/10.1016/j.cose.2017.05.006)]
- [34] Lin YW. Design and implementation of Chinese characters graphical passwords based on redrawing [MS. Thesis]. Guangzhou: South China University of Technology, 2016 (in Chinese with English abstract).
- [35] Nyang D, Kim H, Lee W, Kang SB, Cho G, Lee MK, Mohaisen A. Two-Thumbs-Up: Physical protection for pin entry secure against recording attacks. *Computers & Security*, 2018, 78: 1–15. [doi: [10.1016/j.cose.2018.05.012](https://doi.org/10.1016/j.cose.2018.05.012)]
- [36] Dierks T, Rescorla E. The transport layer security (TLS) protocol version 1.2. 2008. <https://www.rfc-editor.org/rfc/rfc5246> [doi: [10.17487/RFC5246](https://doi.org/10.17487/RFC5246)]
- [37] Shamir A. How to share a secret. *Communications of the ACM*, 1979, 22(11): 612–613. [doi: [10.1145/359168.359176](https://doi.org/10.1145/359168.359176)]
- [38] Abdalla M, Miner S, Namprempre C. Forward-secure threshold signature schemes. In: *Proc. of the 2001 Cryptographers' Track at the RSA Conf. San Francisco: Springer*, 2001. 441–456. [doi: [10.1007/3-540-45353-9_32](https://doi.org/10.1007/3-540-45353-9_32)]
- [39] Boldyreva A. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In: *Proc. of the 6th Int'l Workshop on Public Key Cryptography. Miami: Springer*, 2003. 31–46. [doi: [10.1007/3-540-36288-6_3](https://doi.org/10.1007/3-540-36288-6_3)]
- [40] Gennaro R, Goldfeder S, Narayanan A. Threshold-optimal DSA/ECDSA signatures and an application to Bitcoin wallet security. In: *Proc. of the 14th Int'l Conf. on Applied Cryptography and Network Security. Guildford: Springer*, 2016. 156–174. [doi: [10.1007/978-3-319-39555-5_9](https://doi.org/10.1007/978-3-319-39555-5_9)]
- [41] Agrawal S, Miao P, Mohassel P, Mohassel P. PASTA: Password-based threshold authentication. In: *Proc. of the 2018 ACM SIGSAC Conf. on Computer and Communications Security. Toronto: ACM*, 2018. 2042–2059. [doi: [10.1145/3243734.3243839](https://doi.org/10.1145/3243734.3243839)]
- [42] Boneh D, Lynn B, Shacham H. Short signatures from the Weil pairing. *Journal of Cryptology*, 2004, 17(4): 297–319. [doi: [10.1007/s00145-004-0314-9](https://doi.org/10.1007/s00145-004-0314-9)]

附中文参考文献:

- [34] 林燕文. 基于重绘的汉字图形口令的设计与实现 [硕士学位论文]. 广州: 华南理工大学, 2016.



陈杰(1994—), 男, 硕士生, 主要研究领域为信息安全, 区块链.



蒋昌松(1997—), 男, 博士生, 主要研究领域为隐私保护, 密码学.



许春香(1964—), 女, 博士, 教授, 博士生导师, 主要研究领域为信息安全, 云计算安全, 密码学, 区块链.



韩云霞(1995—), 女, 博士生, 主要研究领域为物联网安全.



张源(1991—), 男, 博士, 研究员, 主要研究领域为密码学, 信息安全, 区块链, 云计算安全.



曹辰辰(1997—), 男, 硕士生, 主要研究领域为密码学, 区块链.