学校代码：10286
分类号：TN918.91
密　级：
U D C：621.3
学　号：

东南大學

SOUTHEAST UNIVERSITY

硕士学位论文

物联网设备共享中的隐私保护技术研究

研究生姓名：
导 师 姓 名：

申请学位类别 工学硕士　学位授予单位 东 南 大 学
一级学科名称 信息与通信工程　论文答辩日期 2017 年 月 日
二级学科名称 信息安全　学位授予日期 2017 年 月 日
答辩委员会主席　评　阅　人

2017 年 月 日

东南大学

硕士学位论文

物联网设备共享中的隐私保护技术研究

专 业 名 称：_____

研究生姓名：_____

导 师 姓 名：_____

RESEARCH ON PRIVACY PRESERVING TECHNOLOGY OF IOT FACILITY SHARING

A Dissertation（Thesis）Submitted to

Southeast University

For the Academic Degree of Master of Engineering

BY

Supervised by

School of Information Science and Engineering

Southeast University

May 2017

东南大学学位论文独创性声明

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得东南大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

研究生签名：_____日期：_____

东南大学学位论文使用授权声明

东南大学、中国科学技术信息研究所、国家图书馆有权保留本人所送交学位论文的复印件和电子文档，可以采用影印、缩印或其他复制手段保存论文。本人电子文档的内容和纸质论文的内容相一致。除在保密期内的保密论文外，允许论文被查阅和借阅，可以公布（包括以电子信息形式刊登）论文的全部内容或中、英文摘要等部分内容。论文的公布（包括以电子信息形式刊登）授权东南大学研究生院办理。

研究生签名：_____导师签名：_____日期：_____

摘要

共享经济鼓励设备所有者将闲置的设备以租赁的方式提供给其他用户使用，从而实现资源的有效利用。随着物联网技术的出现，设备所有者可以通过互联网中间代理远程的将物联网设备共享给其他用户，而无需与用户直接接触。以在线房屋共享租赁平台为例，房主将房屋委托给房屋共享租赁代理平台，由代理平台负责向租赁用户授权，租赁用户利用授权凭证入住，整个交互过程中，通过中间代理进行房屋权限的传递。在此场景中，物联网设备权限进行委托传递时的隐私保护主要体现在设备所有者权限的敏感度保护以及用户身份的匿名保护上。现有的研究主要集中在物联网设备的权限管理上，对于可委托授权的物联网设备共享场景中的隐私保护问题并没有有效的解决方案。本文针对上述问题，提出了一种可保护隐私的委托授权方案。该方案可以实现保护设备所有者权限敏感度的委托授权、可保护隐私的用户身份认证以及基于社交关系信任度的访问控制。本文的具体工作内容如下：

1. 针对物联网设备共享场景中设备所有者委托授权时的权限敏感度保护问题，在 Mambo-Usauda-Okamoto 代理签名方案的基础上进行了改进，提出了一种基于信任度的可控的部分权限委托授权机制。该机制中中间代理在设备所有者许可范围内将部分被委托的权限授权给用户，中间代理使用的代理签名私钥由其和设备所有者共同生成，不仅可以防止 Mambo-Usauda-Okamoto 代理签名方案中设备所有者和中间代理相互抵赖的问题，同时提供了设备所有者权限的敏感度保护。经安全分析证明，该机制可满足权限传递所需的可验证性、不可否认性、可区分性、可识别性和不可滥用性等安全属性。
2. 针对用户访问物联网设备时中间代理过度授权的问题，提出了一种基于信任度的访问控制机制。该机制采用基于角色和信任度的访问控制模型，物联网设备根据由用户动态社交关系所生成的信任度和设备所有者设定的信任度阈值确定是否向访问用户授权。该机制可以有效防止中间代理过度授权的问题。
3. 提出了一种基于动态社交关系的信任度生成方案，该方案根据用户间联系人、社交联系地址和使用的手机 APP 的相似性实时的生成用户当前的信任度。该信任度可根据用户间社交关系的变化进行自适应的调整。
4. 针对用户访问物联网设备进行身份认证时的隐私保护问题，提出了一种基于代理签名和知识签名技术的匿名身份认证机制。该机制通过代理签名技术实现了设备对用户提交的中间代理身份信息的合法性的验证，然后通过用户提交的知识签名在无需用户身份信息的前提下验证该用户是否是由合法的中间代理授权的用户。本文采用 Canetti-Krawczyk 模型对该机制的认证安全性进行了形式化分析和证明，分析表明该机制可以抵抗重放攻击和伪装攻击。
5. 在所提方案的基础上，本文设计并实现了一种可控委托授权的可保护隐私的房屋租赁共享原型系统。该系统由房屋所有者、中间代理、房屋租赁者以及房屋智能锁系

统构成，可实现房屋租赁的委托授权、用户匿名身份认证和基于社交关系信任度的房屋访问控制功能，所有流程交互均通过移动智能终端进行，以实现远程的房屋共享。移动智能终端程序基于 Android 平台开发，房屋智能锁基于 Arduino 平台开发。经测试表明，该系统可以有效的实现远程的房屋租赁共享服务，用户租赁房屋时打开智能锁的认证交互延时均值为 200ms 左右，具有较强的实用价值。

关键词：设备共享，隐私保护，委托授权，匿名认证，访问控制

Abstract

Shared economy encourages the owner to rent his unoccupied facilities to other users so that the personal resources can be used efficiently. Nowadays, the owner can share his IoT (Internet of Technology) facilities to other users remotely online through the middle agent. So the owner doesn't need to interact with the user directly. For example, in a housing rental environment, the owner can entrust his house to a housing rental platform. Then housing rental platform will authorize the house to other users on behalf of the owner. The user can enter into the house with the authorization warrant. In the whole procedure, the transfer of housing permissions is carried out through a middle agent. In this scenario, the privacy protection is reflected in the sensitivity protection of the owner's permissions and the anonymous protection of the user's identity. However, the current research mainly focuses on the authority management of IoT facilities and there are no effective solutions to the privacy protection in the IoT facility sharing environment where the authority of the owner is delegated to a middle agent. Based on the above problems, a novel privacy-protection partial delegation authoriaztion scheme for IoT facility sharing is proposed in this paper. This scheme can realize the controllable transfer of owner's permissions, anonymous authentication of the user's identity and access control based on the social relationship. The main work of this paper is as follows:

1. A controllable partial delegation authorization mechanism based on the trust is proposed for the protection of the sensitive permissions. This mechanism is improved on the basis of the Mambo-Usauda-Okamoto proxy signature scheme. The middle agent will authorize the IoT facility to other users under the owner's approval. The private key of the proxy signature used in this mechanism is generated jointly by the owner and the middle agent. It can prevent the repudiations between the owner and middle agent existed in the Mambo-Usauda-Okamoto proxy signature. It also can protect the sensitive permissions of the owner. According to the security analysis, this mechanism satisfies the security attributes: verifiability, non-repudiation, identifiability, distinguishability and prevention of misuse.
2. A social relationships based access control mechanism is proposed for the excessive authorization of the middle agent. This mechanism uses the role-based and trust-based access control model. It determines whether to authorize the IoT facilities to the user based on the dynamic social relationships of users and the trust threshold set by the owner. This mechanism can prevent the excessive authorization of middle agents effectively.
3. A trust-value generation mechanism based on the dynamical social relationship is

proposed. This mechanism can calculate the trust value timely based on the similarity of contact persons, social cntact addresses and mobile application softwares between users. The trust value can adjust adaptively with the change of social relationships.

4. An anonymous authentication mechanism based on the proxy signature and knowledge signature technology is proposed for the protection of user's identity. In this mechanism, firstly, the IoT facility authenticates the identity of the middle agent through the proxy signature and then verifies whether the user is authorized by the middle agent through the knowledge signature submitted by the user. And the user doesn't need to offer his identity information. The Canetti-Krawczyk model is used to prove the authentication security of the mechanism. The result of analysis shows that the mechanism could resisit replay attacks and camouflage attacks.
5. Based on the scheme mentioned in this paper, a housing sharing rental prototype system is designed. It can realize the controllable authority transfer and privacy protection. This system consists of the housing owner, middle agent, renter and smart lock of the house. The main function includes the delegation authorization of the owner, anonymous authentication of the user and the access control based on the social relationships. All the interactive processes are carried out through the mobile smart terminals to achieve the remote housing sharing. The program of mobile smart terminal is developed based on the Android platform, and the development of the smart lock is based on the Arduino platform. The test shows that this system can realize the remote housing rental sharing service. The average value of the authentication delay when the user open the smart lock is about 200ms. This system is very practical.

Keywords: facility sharing, privacy-protection, delegation authorization, anonymous authentication, access control

目录

插图目录

表格目录

第一章 绪论

1.1 研究背景

当前，全球人口增长和城市化进程的加快对世界各国的社会和经济发展都提出了极大的挑战。据统计[1]，2050 年全世界城市居住人口将增长 29 亿人次，城市总人口将达到 63 亿人次，占世界总人口的 70%。随着大量的人口进入到城市中，世界各大城市都面临着气候变暖、环境污染、交通拥挤等问题。城市中现有的已经发展成熟、完善的公共基础设施已经难以负荷日益增多的城市人口，政府和相关机构组织已经无法向更多的城市居民提供便捷有效的公共服务，因此，如何在现有条件下缓解城市压力，更好的向人们提供公共服务是当前需要着重解决的问题。在这一背景下，各国政府和组织相继提出了“智慧城市”的概念。智慧城市[2][3]是以物联网、互联网、云计算技术等为基础来管理城市公共基础设施和向城市居民提供公共服务，它可以更加精准、有效的实现城市政务、房屋、交通、医疗、旅游等公共事业的管理。智慧城市通过信息资源的整合和共享，可以更加方便的为人们提供各种智能化的服务。

物联网技术的发展、移动智能终端的普及[4]和智慧城市建设的完善为共享经济的发展奠定了基础。据统计[5]，过度消费导致人们购买的商品约 40%左右都处于闲置状态，而资源的闲置也就意味着城市中个人资源的过剩。共享经济可以通过互联网整合线下闲散的资源和服务者向用户提供产品或服务，资源提供方在某个时间段内提供物品的使用权或服务；对于资源使用者，其无法获得资源的所有权而是通过租赁等共享方式获得资源的使用权。共享经济的实质是用户资源的置换，是对现有资源的重复利用，而不是开发新的资源，其适应了智慧城市建设中共享、绿色、开放、协调和创新的要求。随着 Airbnb、Uber、共享单车等共享经济模式的出现，共享经济的价值得到了全世界的广泛认可，人们可以通过互联网实现约车、房屋分享、服装租用和 WiFi 网络共享等，从而在赋予了个人闲置资源更多价值的同时也在一定程度上缓解了当前城市发展的压力。然而目前，互联网中对资源共享的研究还主要局限于消费共享领域，对于各细分领域如个人闲置设备共享、公共设施的共享等的研究还很少。因此个人设备资源共享将是本文研究的重点。

1.1.1 物联网设备共享

物联网[6]（Internet of Things, IoT）是一种物与物相连的互联网，它是一种动态性的全球基础网络，由各种传感器、通信设备、网络设备和信息处理设备组成。物联网通过采用标准化的和各种通用的通信协议可以将任何“物”与互联网相连接进行通信和信息交换，从而实现物的智能化识别、定位和管理。据统计[7]，当前已有 120 亿个传感器装载在道路系统、车辆、房屋、办公室、家庭和自然资源中，其不断的将大数据传输到物联网中。

物联网技术的出现和发展为用户设备共享提供了可能，设备共享，顾名思义就是用户将自己的个人设备作为公共资源分享给其他用户使用。人类社会很早就有了共享的概念，在传统社会中人们可以相互分享书籍或共享一条消息，邻里之间相互借东西也是一种形式的共享，这是最基本的共享模式。在 2000 年以后，随着互联网 Web2.0 时代的到来，出现了一系列的信息资源共享平台如 QQ、大众点评、美团、BBS 论坛等，人们可以通过互联网在信息共享平台中向陌生人分享自己的观点和信息，但是这种模式的共享主要还是集中在信息的分享和内容的提供，并不包含任何实体设备的共享。2010 年以后，国内外出现了一系列的实体资源的共享平台，如表 1.1 所示。人们可以利用这些资源共享平台对外共享自己闲置的设备和服务。以 Airbnb 旅行房屋租赁平台为例，房主将家里的空房信息发布在 Airbnb 平台上，旅行者通过 Airbnb 平台得到可以预定的房屋信息和房主信息，并通过与房主线下的联系获得房屋的使用权。在该共享模式中，Airbnb 的实质还是一个信息资源的共享平台，Airbnb 只负责向旅游者共享房屋租赁信息，其不能代替房主直接向旅游者提供房屋的使用权限，旅游者需要在线下与房主直接联系，向房主提出房屋租赁申请和提供身份证明信息，从而获得房屋的使用权。

表 1.1 国内外资源共享平台

分享类型	代表平台	常用场景
产品分享	滴滴出行、Uber、Rent the Runway、易科学	汽车、设备、玩具、服装
空间分享	Airbnb、小猪短租、Wework、Landshare	住房、办公室、停车位
知识分享	猪八戒网、知乎网、Coursera、名医主刀	智慧、知识、技能、经验
劳务分享	河狸家、阿姨来了、京东到家	生活服务行业
资金分享	LendingClub、Kickstarter、京东众筹、陆金所	P2P 借贷、产品众筹
生产分享	Applestore、WiFi 万能钥匙、阿里巴巴淘工厂	协作生产方式

除了利用互联网共享平台发布共享设备的信息实现设备共享外，人们可以将自己闲置的设备直接连接到物联网中，设备所有者利用移动智能手机通过互联网中介平台远程的向其他用户共享闲置的设备，而设备所有者无需与用户直接接触。该物联网设备共享模式中主要包括四大主体：设备所有者、中间代理、用户和物联网设备，如图 1-1 所示。

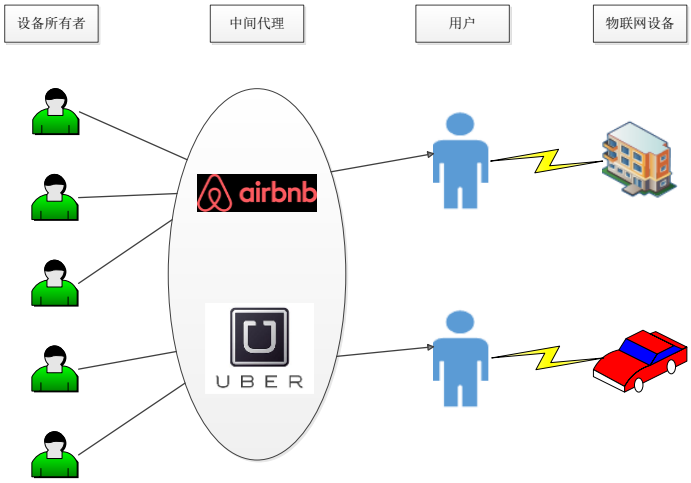


图 1-1 物联网设备共享模式

在图 1-1 中物联网设备是连接在物联网中的，设备所有者将物联网设备委托给中间代理平台，由中间代理平台负责向用户直接授权，用户利用中间代理颁发的授权凭证与物联网设备交互，从而获得物联网设备的使用权限。在该共享模式下，设备所有者、用户、物联网设备都是通过一个第三方的中间代理进行线上交互的，设备所有者无需与用户进行线下的联系。该通过中间代理授权的共享模式适用于智慧城市中用户共享个人闲置设备，其主要原因有：一般而言，设备所有者都有自己的工作，他不是专职的从事设备租赁的工作，其通常无法实时的对用户的请求做出回应，中间代理平台相当于设备所有者与用户之间的中介，其可以代替所有者处理设备租赁的工作，从而减轻了设备所有者管理设备的负担也提高了整个系统的效率；另外，由于设备所有者提供的设备是闲置的或空余的，并非为用户专门提供的，通过中间代理平台的集客效应，可以使得设备所有者向更多的用户共享设备。

1.1.2 共享中的安全问题

在通过中间代理授权的物联网设备共享场景中，设备所有者将设备的授权权限委托给中间代理，再由中间代理对用户授权，用户利用授权凭证与物联网设备进行线上的交互从而获得设备的使用权限。在该共享场景下，设备所有者通过中间代理实现了设备权限的委托传递，从而向更多的用户共享了他的闲置设备，实现了资源的有效再利用，但是其中存在的安全问题也限制了该设备共享模式的广泛应用，下面对这些问题进行具体分析：

1) 用户间的信任是影响设备共享的关键。在物联网设备共享场景中，设备所有者将自己私人设备的授权权限委托给中间代理，由中间代理代替设备所有者向访问用户颁发授权凭证。但是由于设备所有者的权限是存在敏感性差异的，其中某些高敏感度的权限可能是涉及到设备所有者隐私的，而设备所有者对不同中间代理的信任程度是不同的，设备所有者当然不希望对不同信任度的中间代理委托相同的授权权限，然而目前的应用中并没有一个有效的权限管理分配方案实现对不同信任度的用户委托不同的权限，但这对于用户私人设备共享系统来说非常重要。
2) 在该共享模式中存在中间代理过度授权的问题。首先，中间代理可能将物联网设备的权限授权给某些信任度不高或不可信的用户，从而可能引起设备所有者敏感权限的隐私泄露；另外，中间代理也可能将授权使用设备的能力继续委托给下一级中间代理，随着权限委托深度的增加，离设备所有者越远的用户他的信任度是减小的，但是目前并没有一个通用有效的机制实现对不同信任度的用户的访问控制并且防止设备权限的无限传递。
3) 用户在获得中间代理颁发的授权凭证后需要与物联网设备交互进行身份认证，但通常用户需要提供自己的身份、地址、手机号等隐私信息对设备证明自己的身份，而设备在获得这些隐私信息后可能通过物联网随意的发布或非法使用，造成用户隐私的泄露，这对用户的隐私安全造成极大的威胁，然而当前的设备

共享系统中缺乏一个有效的隐私保护机制实现用户身份认证时的个人隐私的保护。另外，物联网环境中，用户是通过智能手机与其他用户或设备通信的，而智能手机和物联网设备等设备的计算能力有限、存储空间较小，且都采用电池供电，所以设计的协议必须满足计算量小、交互简单等特点，而当前一些隐私保护方面的协议大多交互复杂，不适用于小型的轻量型设备。

1.2 国内外研究现状

目前，国内外学者对物联网设备共享的安全研究主要集中在设备权限管理和用户隐私保护两方面，接下来分别对物联网设备权限管理和用户隐私保护的国内外研究现状进行介绍。

1.2.1 物联网设备权限管理研究

Sandhu[8]最早提出了物联网权限管理方案，他提出了一种基于角色的访问控制（Role Based Access Control，RBAC）模型，该模型中引入了角色的概念从而将设备所有者和其权限逻辑分离，并将权限授予角色，用户通过设备所有者为其分配的角色获得相应的权限，该模型有效的减小了授权管理的代价，并较好的实现了对授权的约束。Zhang[9]等基于角色的访问控制提出了一种可扩展的访问控制模型，该方案通过上下文信息对权限的访问控制的实现方法进行描述，将对象的操作转换成对服务的操作，通过收集系统和用户环境中的上下文信息对授予权限的规则进行定义。Fu 和 Ye[10]基于属性的访问控制（Attribute-Based Access Control，ABAC）提出了通过可扩展的访问控制标记语言对环境属性进行描述，设备所有者通过属性对权限和对用户的约束条件进行描述，从而在开放的物联网环境中实现基于用户身份和属性的访问控制方案。在物联网设备共享环境中，存在许多对设备所有者来说陌生的用户，当这些用户申请访问设备时，设备所有者很难直接依据其陌生的身份进行授权，Li[11]等基于角色的信任管理，提出了一种在开放的物联网环境中陌生用户间进行授权的方案，其核心的授权机制是通过委托实现的，但是其没有很好的处理委托深度的问题。Kolev[12]通过信任关系实现陌生用户到设备所有者权限角色的分配，同样该文也没有对委托深度的进行限制。翟征德[13]等提出了一种基于角色的细粒度的可控委托授权模型，该模型根据用户的信任度对设备所有者的敏感权限进行传播控制，但是该方案中用户间的信任度值都是由用户主观指定的不具有通用性。

目前，对物联网共享设备权限管理主要集中在设备所有者对用户进行直接授权管理上，对于通过中间代理进行权限的可控传递缺乏研究，同时也没有一个通用客观的信任度计算方法来表示用户间的信任程度。

1.2.2 物联网设备隐私保护研究

Juels 和 Watenberg[14]提出了基于生物特征的身份认证方案，生物特征是人所独有的信息且不会被遗忘，但是人的生物特征并不是完全隐私的，例如攻击者就可以从杯子上获取用户的指纹信息或者从照片上获取用户的面部特征。为了解决这些问题，Yu[15]等人

提出了综合口令、智能卡和生物信息的身份认证架构，改进了双因素认证的方案的安全性和保护了用户的隐私，但是该方案需要生物识别模块获取生物信息，这对有限计算和通信能力的移动设备来说是困难的。许多学者提出了在物联网环境中对用户的身份信息进行匿名，从而使得设备不知道用户的真实身份。Xue[16]等人提出了针对物联网环境的基于临时证书的双向认证和密钥协商协议，其能实现身份和口令的保护，但是不能抵抗身份猜测攻击，跟踪攻击，内部攻击以及智能卡丢失攻击。攻击者可能通过线下猜测揭露用户的身份，也可以通过某个用户的登录信息追踪特定用户，除此之外内部人员也可以通过同样的方法获取登录用户的口令。Jiang Qi[17]等人基于 Xue 的方案进行改进，提出带有不可关联性的增强型认证方案，改进了前者的安全漏洞并且节省了计算开销。Anna Lysyanskaya[18]研究了假名系统，在假名系统中用户可以使用假名与物联网设备进行匿名的身份交互，假名之间不会相互关联，但是用户可以用一个称作“证书”的声明他与其他用户之间的关系，该文献对已有的假名系统进行分析，发现之前的系统存在不诚实的用户使用相同的假名和证书的情况，比如共享身份等。Brickell[19]等人提出了直接匿名认证方案（Direct anonymous attestation，DAA），该方案基于 CL 群签名和知识证明，其已经被可信计算组织所采纳，用于作为对可信计算平台的远程认证方案，可以实现对平台的匿名身份认证，从而保护了平台隐私信息。甄鸿鹄[20]对直接匿名方案中可以无限次使用 DAA 证书的情况进行改进，将其变为有时间和次数限制的使用 DAA 证书，相比直接匿名认证方案具有更强的匿名性。Derler[21]等研究了智能手机中的近距离无线通信模块（Near Field Communication，NFC），利用直接匿名认证技术向手机和 NFC 提供匿名服务，保护手机和设备所有者的隐私，但是直接匿名认证协议复杂且会产生较大开销。杨力[22]等针对直接匿名认证方案中移动终端只能在单可信域里进行身份认证的问题，采用信任委托的思想实现了在无线网络环境中移动终端漫游到不同可信域中时与可信计算平台之间的匿名身份认证，该方案可以抵抗伪装攻击与重放攻击。无线射频识别[23]（Radio Frequency Identification，RFID）系统是利用小型、低成本的 RFID 标签实现非接触式自动身份识别。RFID 标签和阅读器通过无线射频信道通信，消息可能被非法用户获取，这就会造成其中有关用户身份和位置等隐私信息的窃取。针对 RFID 的匿名认证系统主要分为公钥加密系统和非公钥加密系统，公钥加密系统主要是基于椭圆曲线加密算法，其中包含的模指数运算不适用于计算能力有限的标签；Gope[24]等基于一次 Hash 函数和异或运算提出了轻量的身份认证方案，不足之处是不能抵抗中间人攻击和伪造攻击；陆颖颖[25]针对物联网 RFID 技术的身份认证问题进行研究，为了防止用户隐私信息的泄露，改进了服务器与 RFID 阅读器之间基于公钥基础设施的认证方案以及 RFID 标签和阅读器之间基于哈希函数的协议。张俊松[26]对椭圆曲线加密的身份认证方案进行研究，提出将安全的 Hash 算法结合椭圆曲线加密方案实现物联网系统中的安全的信息交互，同时利用随机假名的隐私保护方案可以保护用户的身份。张磊[27]研究分析了车载自主网络中的已有的隐私认证方案，实现了在车联网环境下的基于车主身份的匿名批认证方案，最后设计了一套针对 IBV 实现汽车管理和通信的车联网系统。姜顺荣[28]对车载自组网中相关隐私保护技术进行研究，自组网中车辆会定期性广播与自身交通有

关的信息，其中涉及车辆身份、速度和位置等隐私信息，攻击者在无线环境中获得这些信息对车辆进行分析，对用户实现进一步的攻击，作者针对攻击提出了基于 HMAC（Hash-based Message Authentication Code）的对车辆匿名批量认证的方案，其可以撤销已经被攻击者攻击成功的车辆。

目前，国内外学者主要对物联网设备的隐私保护方案进行研究，针对物联网设备共享场景中的用户隐私保护机制研究还很少，还未见到公布的研究成果。

1.3 论文研究内容及意义

目前国内外学者对物联网设备权限管理和隐私保护技术做了大量的研究，对其中存在的问题进行分析，并提出了相应的解决方案。但是可以发现，当前专门针对物联网设备共享场景中的设备权限管理和用户隐私保护技术的研究以及相应的实现方案还很少。在本文的物联网设备共享场景中，设备所有者将设备委托给中间代理，中间代理负责向用户授权，用户利用中间代理颁发的授权凭证获得设备的使用权限，在整个交互过程中，设备的权限通过中间代理在线传递，设备所有者无需与用户直接接触。因此如何在设备所有者、中间代理和用户之间进行设备所有者敏感权限的可控传递和提供相应的隐私保护将是本文研究的重点，本文的具体研究和工作内容如下：

1) 针对在物联网设备共享场景中设备所有者委托授权时权限敏感度保护的问题，在 Mambo-Usauda-Okamoto 代理签名方案的基础上进行了改进，提出了一种基于信任度的可控的部分权限委托授权机制，该机制中中间代理在设备所有者委托的权限范围内对用户进行授权，中间代理使用的代理签名私钥由其和设备所有者共同生成，最后对该委托授权方案的安全性进行分析。
2) 针对用户访问物联网设备时中间代理过度授权的问题，提出了一种基于信任度的访问控制机制。该机制采用基于角色和信任度的访问控制模型，物联网设备根据由用户动态社交关系所生成的信任度和设备所有者设定的信任度阈值确定是否对访问用户授权。
3) 提出了一种基于动态社交关系的信任度生成方案，该方案根据用户间联系人、社交联系地址和使用的手机软件的相似性实时的生成用户当前的信任度。
4) 针对用户访问物联网设备进行身份认证时的隐私保护问题，提出了一种基于代理签名和知识签名技术的匿名身份认证机制。通过代理签名实现了设备对用户提交的中间代理身份信息合法性的验证，通过用户提交知识签名在无需用户身份的前提下间接的验证该用户是否是由合法的中间代理授权的用户；采用 Canetti-Krawczyk 对该机制的认证安全性进行了形式化分析和证明。
5) 设计并实现了一个可控委托授权的可保护隐私的房屋租赁共享原型系统。该系统由房屋所有者、中间代理、房屋租赁者以及房屋智能锁系统构成，可实现租赁房屋的委托授权、用户匿名身份认证和基于社交信任度的访问控制功能，所有流程交互均通过移动智能终端进行。移动智能终端程序基于 Android 平台开

发，房屋智能锁系统基于 Arduino 平台开发，最后对系统进行功能和性能测试。

1.4 论文组织结构

本文共分为 6 章，每章的主要内容具体如下：

第一章为绪论。本章主要介绍了本文的研究背景，首先介绍了物联网设备共享，并对物联网设备共享场景下存在的安全问题进行分析；然后总结了现阶段国内外物联网设备权限管理和基于隐私保护的身份认证方案的研究现状；最后提出了本文主要研究内容和意义。

第二章为相关技术。主要对在物联网设备共享中实现可保护隐私的委托授权方案的相关技术进行介绍。首先介绍了委托授权的基本概念、代理签名技术以及经典的 M-U-O 代理签名方案；其次介绍了隐私保护方面的相关技术，包括零知识证明，直接匿名认证方案和基于嵌入式设备的直接匿名认证方案，并对这些协议的优缺点进行分析；然后介绍基于信任度的访问控制的相关知识，包括基于信任度的访问控制模型、信任的概念以及计算信任度的形式化方法；最后介绍了分析协议安全性的安全模型，并主要介绍了 CK 模型。

第三章是基于信任度的委托授权和访问控制。本章首先根据物联网设备共享场景中设备所有者委托授权时权限敏感度保护的问题，提出了一种基于信任度的可控的部分权限委托授权机制，介绍了该方案的具体流程以及实现原理。该机制在 Mambo-Usauda-Okamoto 代理签名方案的基础上进行了改进，中间代理在设备所有者委托的权限范围内对用户授权，中间代理使用的代理签名私钥由其与设备所有者共同生成，最后文中给出了相应的安全性分析；然后根据用户访问物联网设备时中间代理可能过度授权的问题，提出了一种基于信任度的访问控制机制，该机制采用基于角色和信任度的访问控制模型，根据用户动态社交关系所生成的信任度和设备所有者设定的信任度阈值确定是否授权；最后，提出了一种基于动态社交关系的信任度生成方案，该方案根据用户间联系人、社交联系地址和使用的手机软件的相似性实时的生成用户当前的信任度。

第四章是可保护隐私的用户身份认证机制。本章针对用户访问物联网设备进行身份认证时的隐私保护问题，提出了一种基于代理签名和知识签名技术的匿名身份认证机制，该机制通过代理签名技术实现了设备对用户提交的中间代理身份信息合法性的验证，再通过用户提交的知识签名在无需用户身份信息的前提下验证用户是否是合法的中间代理授权的用户，最后采用 Canetti-Krawczyk 模型对该机制的认证安全性进行了形式化分析和证明。

第五章是一个可控委托授权的可保护隐私的房屋租赁共享原型系统的设计与实现。本章首先介绍了系统的整体架构，由房屋所有者、中间代理、房屋租赁者以及房屋智能锁系统构成，可实现房屋租赁的委托授权、用户匿名身份认证和基于社交信任度的房屋访问控制，所有的交互流程通过移动智能终端进行。移动智能终端程序基于 Android 平台开发，房屋智能锁基于 Arduino 平台开发。最后对系统的功能和认证性能进行测试。

第六章为总结与展望。本章主要是对全文的内容与工作进行总结，并提出工作中存在的不足之处，并对以后的物联网设备共享中隐私保护技术的研究工作进行展望和规划。

第二章 相关技术

本文针对由中间代理授权的物联网设备共享场景提出了一个可保护隐私的委托授权方案，该方案主要实现了设备所有者的委托授权、可保护隐私的用户身份认证和基于社交信任度的访问控制，本节将对这三方面相关的基本概念和技术知识进行介绍，另外介绍了分析协议安全性的安全模型进行。

2.1 委托授权

2.1.1 基本概念

委托授权[29]的一般定义是在一个系统中某个用户实体将其权限委托给其他的用户实体，获得权限的用户实体可以在前者的要求下执行某些任务或者操作。在委托授权的过程中主要有三个对象：委托者、委托的对象以及被委托者。如图 2-1 所示是一种委托权限的授予方式，委托者对委托的对象具有所有权，被委托者向委托者申请委托对象的使用权，委托者在一定的约束条件下向被委托者共享或者传递权限。

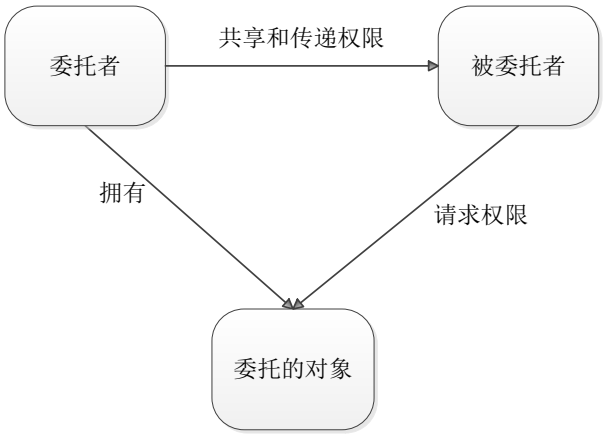


图 2-1 委托授权方式

接下来对委托授权的特点进行介绍，这些都是实现权限委托传递的基础。

1) 持久性

用户实体之间委托的持久性可以分为永久委托和临时委托。永久委托是指被委托的用户实体可以一直代替作为委托方的用户实体执行某些任务和操作且一旦委托方将权限委托给被委托方，就不能撤销该委托关系。临时委托是指权限的委托是受时间限制的，超过委托时间，委托关系就会失效，被委托者就会失去权限。

2) 单一性

委托授权的单一性可以分为单一委托和非单一委托，是指在委托者向被委托者委托权限后是否还拥有该权限。单一委托后委托者仍然拥有该权限，而非单一委托是指在委托关系存在期间，委托者不再拥有该权限，委托关系结束后，委托者恢复权限。

3) 全体性

委托授权的全体性是指委托者对被委托者委托的权限是全部委托还是部分委托，全

部委托中被委托者获得委托者的全部权限，而部分委托中被委托者只获得委托者的一部分权限。

4) 委托管理

委托关系管理分为自己管理和代理管理，其实质是指委托关系是由委托者自己管理还是由一个可信的第三方机构来管理。在自己管理模式中，被委托者向委托者发出委托申请，由委托者向被委托者委托权限或者撤销权限，而代理管理是由可信的第三方负责权限的委托和撤销。

5) 可撤销性

可撤销性是指委托方可以撤销委托给被委托者的权限，分为授权与非授权依赖撤销和级联撤销。

2.1.2 代理签名

数字签名技术是通过对信息的内容进行杂凑运算生成消息摘要，发送消息的人用其私密密钥对消息摘要加密，由此生成一个数字签名，该数字签名只能被一个用户的公钥解密。当接受者收到消息和数字签名后，先用发送者的公钥解密数字签名还原出消息摘要，再用相同的杂凑算法计算收到的消息的摘要值，将此摘要值与解密得到的摘要值进行比对，如果两者相等则说明消息在传递的过程中没有被篡改。数字签名是实现用户身份认证、消息认证、授权以及不可抵赖性的重要技术。但是在物联网设备共享中，数字签名只适用于设备所有者对用户直接授权的场景，其不适用于设备所有者委托中间代理授权的场景。

Mambo[30]等最早提出了代理签名的概念。代理签名是数字签名的一个扩展分支，整个代理签名系统由原始签名者、代理签名者和验证者构成。其主要思想是原始签名者不仅可以自己对消息进行数字签名，还可以将签名的权限传递给代理签名者，当原始签名者无法签名时，代理签名者可以代替原始签名者对消息签名。以物联网设备共享平台为例，设备所有者相当于原始签名者，中间代理相当于代理签名者，用户就是验证者，用户可以对中间代理签发的授权证书进行验证，判断该中间代理是否由设备所有者授权。图 2-2 是一个代理签名的流程图。

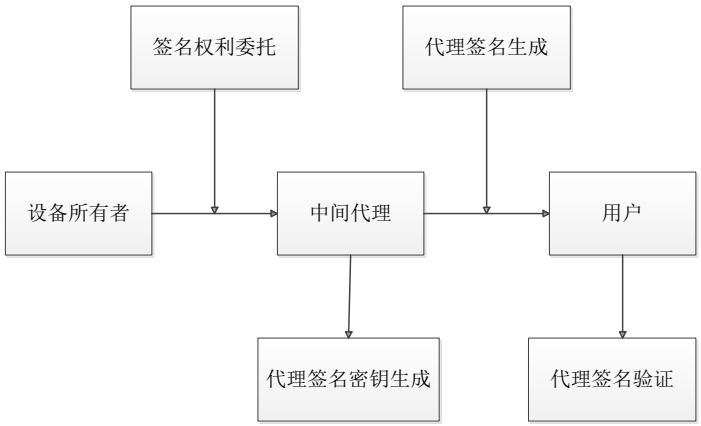


图 2-2 代理签名图

根据设备所有者向中间代理委托权限的程度不同可以分为三种授权方式：全部授权、

部分授权和基于证书的授权，分别所述如下：

- 全部授权（Full delegation）：在全部授权中，中间代理和设备所有者拥有相同的签名私钥，中间代理和设备所有者生成的数字签名完全一样。因此其中存在的一个很大的问题是，当中间代理对未经设备所有者授权的用户进行消息签名时，该欺骗性行为无法被检测出来，中间代理可以否认其违法行为，因为任何人都不能辨别出该签名是由中间代理还是由设备所有者生成的。
- 部分授权（Partial delegation）：在部分授权中，设备所有者首先生成自己的私钥 s，然后在其私钥 s 的基础上生成新的私钥 s'，并将 s' 作为代理签名密钥通过安全的方式传递给中间代理，中间代理不可以通过 s' 推断出设备所有者的私钥 s，然后中间代理用 s' 对发送给用户的消息签名。用户计算出新的代理签名验证公钥后对代理签名的正确性进行验证。由于中间代理与设备所有者生成的数字签名不同，所以当设备所有者发现未经授权的用户获得权限时，可以判断出是由谁对该用户进行授权的。部分授权代理签名可以分为无保护代理的代理签名和有保护代理的代理签名，无保护代理的代理签名是指除了设备所有者和中间代理外，任何第三方都不可以生成与该中间代理相同代理签名，有保护代理的代理签名是指只有中间代理可以生成有效的代理签名。
- 授权书授权（Delegation by warrant）:是指设备所有者通过授权书向用户说明该中间代理是由其授权的合法的中间代理。目前有两种类型的授权书授权方案：
  1) 授权证书由两部分组成：授权消息和设备所有者对中间代理的公钥的签名，或者设备所有者直接在授权书中说明中间代理的身份。中间代理利用其私钥进行普通的数字签名操作，有效的代理签名由签名和授权书组成。需要说明的是该方案中的代理签名的验证与设备所有者的公钥无关。
  2) 授权书由授权消息和设备所有者对新生成的公钥的签名组成，与新生成的公钥对应的代理签名私钥通过安全信道传递给中间代理。中间代理利用代理签名私钥对消息进行签名，用户对签名的验证方法与 1)类似，但是签名的验证与设备所有者的公钥有关。

可以根据上述不同的授权类型设计代理签名方案，在方案设计的过程中还需要考虑代理签名的安全性、签名消息的大小和用户的计算能力等因素。部分授权和授权书授权相对完全授权代理签名方案更加的安全，并且在这两个方案中，中间代理和设备所有者生成的签名是可以区分的。

Lee[31]等定义了一个安全的强代理签名方案应满足的安全属性，如下所述：

- 可区分性（Distinguishability）：在有效的多项式时间内，可以区分出是由中间代理生成的代理签名还是由设备所有者生成的普通数字签名。
- 可验证性（Verifiability）：用户对收到的代理签名进行验证，从而可以确信该签名是有效的且是由设备所有者授权的中间代理签发的。
- 强不可伪造性（Strong unforgeability）：只有中间代理可以代替设备所有者生成有效的代理签名，设备所有者与任何第三方都不可以生成与中间代理相同的代

理签名。

- 身份可识别性（Identifiability）：设备所有者可以通过代理签名知道生成该签名的中间代理的身份。
- 不可否认性（Non-deniability）：中间代理不能否认由其生成的任何代理签名；
- 不可滥用性（Prevention of misuse）：中间代理只能用设备所有者授予的代理签名密钥生成有效的代理签名，且不能对未经设备所有者授权的消息进行签名，也不能将代理签名密钥分享给其他用户使用。

2.1.3 M-U-O 代理签名方案

Mambo、Usauda 和 Okamoto[30]三位学者提出的 M-U-O 代理签名方案是物联网权限管理中应用最为广泛的代理签名方案之一，该方案可以有效的解决设备所有者利用数字签名不能进行签名权利委托的问题，该方案的参与者包括设备所有者（原始签名者）、中间代理（代理签名者）和用户（验证者）：

1) 系统参数生成

p 是一个大素数，$g \in Z_p^*$ 是 p 的生成元，设备所有者的公私钥对是 (s,v)，其中私钥 $s \in Z_p^*$，公钥 $v = g^s \bmod p$。

2) 代理签名的生成

设备所有者生成随机值 $k \in Z_p^*$，计算 $K = g^k \bmod p$ 和 $s' = s + kK \bmod p - 1$；通过安全信道将代理签名密钥 (s', K) 发送给中间代理。

3) 代理签名密钥验证

中间代理需要对代理签名密钥的有效性进行验证，首先计算等式 $g^{s'} = vK^K \bmod p$ 是否成立，若该等式成立，则 (s', K) 是有效的代理签名密钥，否则中间代理拒绝接收该代理签名密钥，并要求设备所有者重新发送新的代理签名密钥或终止该协议。

4) 代理签名生成

设 m 是需要签名的消息，中间代理用代理签名私钥 s' 对消息 m 签名，生成普通的数字签名 $Sign(s', m)$，最后将 $(m, Sign(s', m), K)$ 发送给用户。

5) 代理签名验证

用户计算代理签名验证公钥 $v' = vK^K \bmod p$，并用 v' 对代理签名的有效性进行验证。

M-U-O 代理签名方案是部分授权的代理签名方案，根据上述安全的强代理签名方案应满足的安全属性对该方案进行分析可以发现其中存在的安全问题如下所述：

1) 不满足不可否认性：设备所有者和中间代理都可以声称该代理签名是由对方生成的，因为设备所有者知道中间代理生成代理签名使用的代理签名密钥 s'，设备所有者可以伪装成中间代理生成有效的代理签名，因此无法判断真正的代理签名者。
2) 不满足身份可识别性：在代理签名中没有包含中间代理的信息，因此设备所有者无法从代理签名 $(Sign(s', m), K)$ 中识别出中间代理的真实身份。

3) 滥用签名权力：由于代理签名密钥中不包含中间代理的私钥信息，因此中间代理可以将代理签名私钥传递给其他未经设备所有者授权的用户，然后未授权用户可以继续向其他用户授权，中间代理同样可以否认其滥用权限的行为。
4) 密钥依赖性低：代理签名私钥中只包含了设备所有者的私钥信息，不含有中间代理的私钥信息。

2.2 隐私保护技术

在由中间代理授权的物联网设备共享场景中，为了解决用户对物联网设备进行身份证明时身份等隐私信息的保护问题，可以使用 Brickell 等提出的 DAA 直接匿名认证方案，该方案基于群签名和知识证明的方法，用户在对物联网设备证明身份时不会泄露自己的身份等隐私信息，但是在该方案中用户与物联网设备间的交互复杂、计算量大，不适用于无线移动环境中计算能力有限的轻量型设备，He 等提出了基于嵌入式系统的直接匿名认证方案，该方案简化了 DAA 方案中的群签名协议，适用于计算资源受限的系统。本节对本文提出的匿名身份认证方案所用到的相关概念与技术知识进行介绍。

2.2.1 群签名

Chaum 和 Van Heyst[32]首先提出了群签名的概念及其相应的协议方案，一个基本的群签名协议包括群成员和群管理员。群签名的核心思想是一个群中的成员可以匿名的代表群对消息进行签名，该签名与普通的数字签名相同，并且群签名可以只用单个群公钥进行公开验证。群签名有三个基本的属性要求：第一，群中只能有一个成员对某个消息签名，不能有多个成员对同一个消息签名；第二，收到群签名的用户可以对签名的有效性进行验证，但是他不知道生成该签名成员的真实身份；第三，当发生纠纷时，群管理员可以揭露是由哪个成员签发的群签名。当用户访问某个物联网设备时，其并不想让设备知道他的身份，但是他又必须让设备知道他是由正确的所有者授权的用户时可以利用群签名的概念。在物联网设备共享中设备所有者相当于群管理员，用户就是某个群成员，而物联网设备就是验证者。如图 2-3 所示是一个群签名流程图。

群签名的具体工作步骤如下所述：

1) 群系统建立：作为群管理员的设备所有者建立群系统，生成群公开参数、群公钥和群私钥，并且生成所有者打开群签名的打开密钥。
2) 群成员加入算法：该算法由设备所有者和用户共同执行，想要加入该群的用户与设备所有者交互，然后设备所有者生成该用户的私钥并将生成的群证书发送给用户，用户成为该群中的一员。
3) 群成员签名:输入消息 M、某个群成员的群证书和签名密钥后，生成群签名。
4) 签名验证算法：输入群签名、系统公开参数、群公钥和消息 M，验证该群签名是否有效。
5) 打开群签名：该算法由设备所有者来完成，输入系统公开参数、消息 M、所有者的打开密钥和群签名，最后输出签名用户的身份。

6) 群成员撤销算法：设备所有者可以撤销群中的某个用户，然后该用户就不能再生成有效的群签名。

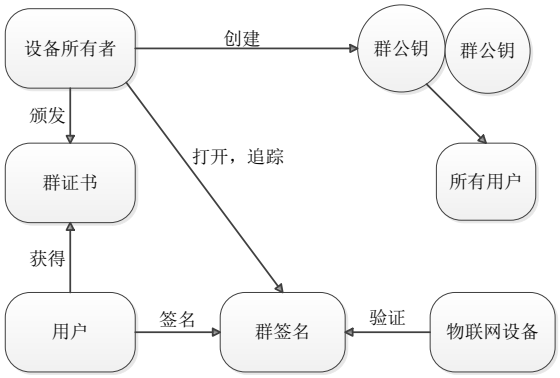


图 2-3 群签名流程图

安全的群签名需要满足不可伪造性、匿名性和不可关联性等属性。在物联网设备共享环境中，签名的不可否认性是指只有群中的用户可以对消息签名，同时当作为群管理员的设备所有者打开签名时，签名的用户不能否认该消息是由其签发的；签名的匿名性是指在不知道设备所有者打开密钥的情况下是无法知道签名该消息的用户的身份；签名的不可关联性是指除了设备所有者外任何其他用户无法判断两个群签名是否是由同一个用户签发的。

Camenisch 和 Michels[33]两人提出了基于强 RSA 假设的群签名协议，也称为 CM 协议，该协议与其他群签名协议相比可以更加自由的增添新的用户到设备所有者的群中，且生成的群签名长度很短，签名算法和验证算法的交互步骤也很少，非常适用于物联网设备、移动智能终端等计算能力较弱、能量有限制的轻量型设备，所以相对来说 CM 协议是一个高效率的群签名方案。

2.2.2 知识证明

知识证明[34]（Zero-knowledge proof）的概念最早是由 Shafi Goldwasser、Silvio Micali 和 Charles Rackoff 提出的。以物联网设备共享为例可以将其简单的理解为用户能够在不对设备提供任何有关其身份等有用信息的情况下，使设备相信其是经过所有者授权的合法用户，从而获得服务。知识证明可以分为交互式知识证明和非交互式知识证明，交互式是指拥有知识的一方需要和验证方之间有交互的过程，而非交互式则需要利用某个随机序列代替交互的过程。下面通过一个开门的例子对知识证明进行说明，如图 2-4 所示。

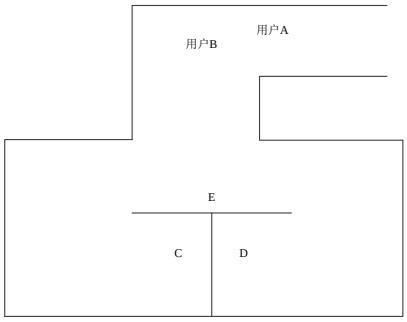


图 2-4 开门例子

图中 C 处与 D 处中间是一扇门，用户 A 和用户 B 如果想打开门就必须要有钥匙。用户 A 想打开门但是他有没有钥匙，用户 B 声称他有钥匙且能打开门，所以他就通过以下办法向用户 A 证明他有钥匙：

第一步：用户 A 在入口不动，用户 B 移动到 C 点处或者 D 点处；

第二步：用户 A 移动到 E 点处，然后要求用户 B 用钥匙打开门从开始站的一边进入到另一边；

第三步：若用户 B 真的有钥匙，那么他就能按照用户 A 的要求从正确的边出来；

第四步：用户 A 重复多次要求用户 B 进行上述步骤，如果用户 B 每次都能从正确的门一边出来，那么用户 A 相信用户 B 有钥匙。

从以上的例子可以看出，用户 A 在没有亲自用用户 B 的钥匙开门的情况下，通过反复让用户 B 开门来确认用户 B 是否拥有钥匙，该过程称为知识证明，其中钥匙就代表秘密知识。根据投硬币的原理，用户 B 有从 C 点出来或者从 D 点出来有两种可能，因此若用户 B 没有钥匙但其每次都能成功欺骗用户 A 的概率是$1/2^n$，其中n是用户 A 让用户 B 反复试验的次数。

2.2.3 DAA 认证方案

可信计算组织 TCG（Trusted Computing Group）在 TPM 规范 1.1b[35]中定义了一种可信的第三方协议 Privacy-CA（Privacy Certification Authority）方案，该方案将可信的第三方 Privacy-CA 作为权威证书机构向可信计算模块 TPM（Trusted Platform Modules）颁发身份证书，当 TPM 要向验证者证明其身份时，将其身份证书发送给验证者，验证者再将 TPM 的身份证书发送给 Privacy-CA 证书机构，Privacy-CA 证书机构对 TPM 身份证书的合法性进行验证，最后将验证结果发送给验证者，从而验证者在不知道 TPM 的身份的情况下完成对 TPM 的身份认证。从上述认证过程中可以看出 Privacy-CA 参与到 TPM 身份认证的每一步中，因此 Privacy-CA 会成为整个认证方案的瓶颈，其可能和验证者勾结或因为某些原因将 TPM 的身份泄露给验证者。针对 Privacy-CA 认证方案中的问题，Brickell[19]等提出了直接匿名认证方案（Direct Anonymous Attestation, DAA）用于实现平台间的匿名身份认证，可信计算组织已经在 TPM 规范 1.2[36]中采纳了该方案。DAA 直接匿名认证方案基于 Camenisch-Lysyanskaya 群签名方案[37]和基于知识的离散对数算法。DAA 认证可以看做一种特殊的不能公开成员身份的群签名方案，在该方案中证书发布中心不需要每次都参与，用户只要向证书发布中心申请一次 DAA 证书，就可以在一段时间内多次使用，从而减小了对第三方可信机构的依赖。该方案中主要包含两个子协议:加入（Join）协议和签名/验证（Sign/Verify）协议。如图 2-5 所示是直接匿名认证方案的模型图，包括 DAA 证书发布中心、用户（TPM）和物联网设备。用户选择一个秘密的随机值，然后通过一个安全的两方协议与 DAA 证书发布者交互，DAA 证书发布者向用户颁发 DAA 证书，即 DAA 证书发布者在秘密值上的 CL 签名，当用户向物联网设备证明身份时，其利用秘密随机值和 DAA 证书对消息进行 DAA 签名，物联网设备对消息进行验证，并相信用户通过知识证明得到了匿名签名，并且物联网设备不

知道用户的真实身份。

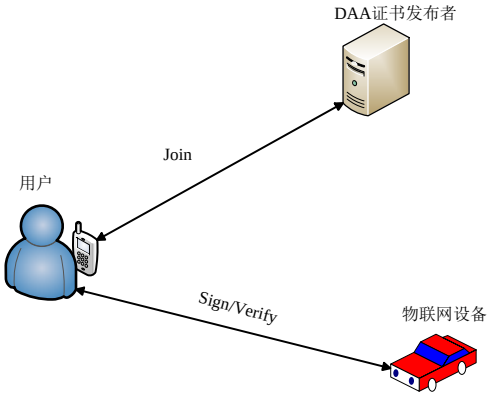


图 2-5 直接匿名认证方案模型图

该 DAA 认证方案中存在的问题是协议交互过程复杂，并且计算量很大，不适用于物联网设备共享系统中计算资源有限的移动智能终端等轻量型设备。

2.2.4 DAA-ED 匿名认证方案

目前，大多数学者们提出的直接匿名认证方案只适合于个人电脑、服务器等大型的计算能力强的设备，对于计算能力弱、电池能力有限的移动终端而言这些协议都过于复杂并且有很高的计算开销。He Ge 和 Stephen R.Tate[38]两人提出了一种基于嵌入式设备的直接匿名认证方案，简称 DAA-ED 匿名认证方案，其能有效解决传统 DAA 方案认证过程复杂，计算开销大等问题。基于嵌入式设备的直接匿名认证方案的主要内容如下：

该方案基于 CM 群签名方案，证书的生成过程与 CM 群签名群成员证书的生成方法相同。其中的主要参与者是设备所有者、用户和物联网设备，证书发布中心相当于群管理员，具体协议的内容如下所述

1. 系统参数生成：
   1) 设备所有者随机选择安全参数 σ ，n 是特殊 RSA 模数，$n = pq$，其中 p 和 q 为 σ 比特长素数（即 $p,q>2^{\sigma}$），并且 $p=2p'+1$， $q=2q'+1$。g 是循环群 QR_n 的生成元。n， g 是系统的公开参数， p 和 q 是秘密值；
   2) 生成安全参数 α,l_c,l_s,l_b，其值均大于 1，生成整数 X,Y；
   3) $H_1:\{0,1\}^{*}\rightarrow Z_n^{*}$， $H_2:\{0,1\}^{*}\rightarrow\{0,1\}^{l_c}$，且均为抗强碰撞哈希函数。
2. 成员加入协议：

该协议采用了与 CM 群签名方案相同的方法生成群成员证书，也就是用户加入了设备所有者为群管理员的群。设备所有者向用户颁发群成员证书 (E,s)，其中 s 为素数且 $X<s<X+2^{l_s}$， s 是用户的私钥，且满足 $E^s\equiv g(\mathrm{mod}\,n)$。

3. 匿名认证过程：
   1) 用户选择随机整数：$b\in_R[Y-2^{l_b},Y+2^{l_b}]$，$t_1\in_R\pm\{0,1\}^{\alpha(l_s+l_c)}$，$t_2\in_R\pm\{0,1\}^{\alpha(l_b+l_c)}$，同时计算 $T_1=E^b(\mathrm{mod}\,n)$，$T_2=g^b(\mathrm{mod}\,n)$，$d_1=T_1^{t_1}(\mathrm{mod}\,n)$，$d_2=g^{t_2}(\mathrm{mod}\,n)$；

2) 用户生成：$c = H_2(g \| T_1 \| T_2 \| d_1 \| d_2 \| m)$，$v_1 = t_1 - c(s - X)$，$v_2 = t_2 - c(b - Y)$；

3) 用户将(c, T_1, T_2, v_1, v_2)发送给物联网设备；物联网设备收到(c, T_1, T_2, v_1, v_2)后，对知识签名进行验证，计算$c' = H_2(g \| T_1 \| T_2 \| T_1^{v_1 - cX} T_2^c \| g^{v_2 - cY} T_2^c \| m)$，如果$c' = c$，且$v_1 \in \pm\{0,1\}^{\alpha(l_c + l_s)+1}$，$v_2 \in \pm\{0,1\}^{\alpha(l_b + l_c)+1}$，则物联网设备就接收该签名，完成对用户的认证。

该方案适用于物联网设备共享中的移动智能终端等轻量型设备，但是该方案存在安全缺陷，其不能有效抵抗用户的伪装攻击和重放攻击。

2.3 基于信任度的访问控制

访问控制是一种限制和控制那些通过通信连接对设备或服务进行访问的能力，其主要功能有：首先，防止非法用户访问资源；其次，允许合法的用户访问资源；最后，防止合法的用户对未授权的资源进行访问。在基于角色的访问控制中通过定义角色将用户与权限分开，并将权限赋予角色。在物联网设备共享中用户通过设备所有者为其分配的角色获得权限，然而对于陌生的用户，设备所有者很难直接依据其身份分配角色。针对这些问题人们又提出了基于信任度的访问控制策略，其核心思想是物联网设备所有者根据“上下文”计算对访问用户的信任度，再根据用户的信任度来判断用户是否可以获得角色，也就是说信任度决定了用户是否能获得角色对应的权限。如图 2-6 所示是一种基于信任度的访问控制模型[39]，下面通过形式化的定义对访问控制模型进行介绍。

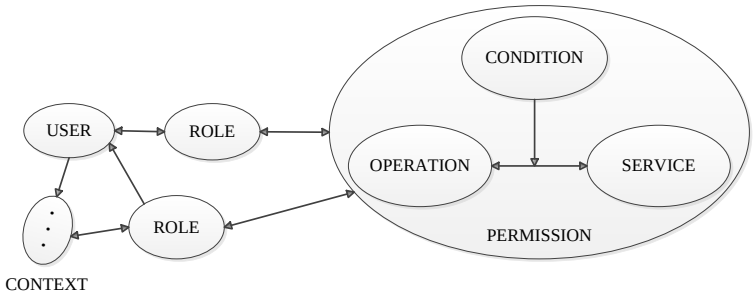


图 2-6 基于信任度的访问控制模型

定义 2.1 访问控制模型的形式化定义如下：

1) 用户$USER = \{u_1, u_2, ..., u_n\}$，是物联网设备共享场景中用户的集合，其可以独立访问物联网设备；

2) 服务$SERVICE = \{s_1, s_2, ..., s_n\}$，是设备所有者为用户提供的各种服务的集合；

3) 操作$OPERATION = \{o_1, o_2, ..., o_n\}$，用户对物联网设备权限进行操作的集合；

4) 权限$PERMISSION = \{p_1, p_2, ..., p_n\}$，是设备所有者权限的集合；

5) 条件$CONDITION$是用户可以对设备权限进行操作的条件，即若用户想要获得某个权限，其信任度必须大于设备所有者设定的该权限的信任度阈值。

6) 角色$ROLE = \{r_1, r_2, ..., r_n\}$，是设备所有者设置的角色集合，不同的角色对应不同的权限。

7) 信任度*TRUSTVALUE*，是由设备所有者根据某个信任算法得到的可以衡量用户可信度的值，定义域为[0,1]。

8) 上下文*CONTEXT*是设备所有者计算信任度值时物联网设备共享中的上下文环境。

在该模型中实现对不同用户的访问控制的关键是设备所有者与用户间的信任，因此首先对信任[43]的属性和类型进行介绍。

1) 信任的属性

- 主观性：在用户的社交关系网络中，用户间的信任是主观的，通常是由用户根据另一个用户的行为表现主观确定的，不同的用户对信任会有不同的判断标准；
- 动态性：用户间的信任关系不是一层不变的而是动态变化的，随着时间、环境、判断标准的变化，用户间的信任会增加或降低；
- 模糊性：用户间的信任不是绝对的，具有很强的不确定性、随机性，在开放的物联网环境中用户不能精确的对与其他用户将来的关系作出判断；
- 可度量性：用户间的信任关系是可以用数值度量的，这个数值通常是一个概率值又称为信任度，由用户根据目标用户的历史行为表现推断得到，其反映了用户对目标用户的熟悉度、可靠度以及诚实度等的认识。
- 单调性：信任关系一般都是单方面的、非对称的，用户 A 对用户 B 的信任程度不代表用户 B 对用户 A 的信任程度，不同的用户对信任关系有不同的评价标准。

2) 信任的类型

- 直接信任：在直接信任关系中，用户与目标用户之间是认识的或者具有共同的交互经历，用户自己依据历史信息对目标用户的行为表现、身份、诚实度以及过去交互的结果好坏做出判断后得出目标用户的信任度值；
- 间接信任：在间接信任关系中，用户与目标用户间是不认识的也没有共同的交互经历，但是彼此间有共同认识的人，用户为了评估目标用户的信任度可以依据共同认识的人对目标用户的评价做出最终判断；
- 综合信任：综合信任关系就是结合直接信任关系和间接信任关系，用户同时结合自己的判断和共同认识的人的评估确定对目标用户的可信度。

根据上述信任的特点，可以发现用户间的信任度是一个主观值，其受很多不确定因素的影响，而对于一个实用的物联网设备共享系统而言必须有一个客观、一致的信任度计算方法对用户间的信任进行度量。在不同的物联网环境中计算用户的信任度时需要考虑不同的信任特征，并且使用的信任度计算方法也不同，下面对一般的计算信任度的形式化方法进行定义：

定义 2.2 $CONTEXT=(TF,TA)$，其中：

1) $TF=\{tf_1,tf_2,...,tf_n\}$表示设备所有者对用户的信任特征集，其中$tf_i$中的*character*表示特征名称，$w_i$表示该特征在计算信任度时所占的比重；

2) $TA=\{ta_1,ta_2,...,ta_n\}$，表示计算信任度的算法名称，可以是加权算法、乘法算法、取最小值算法等。

用户在某个物联网环境中的的信任度为：$TrustValue^i = ta_i(w_1,tf_1,w_2,tf_2,...,w_n,tf_n)$，如图 2-7 是物联网环境中信任度与信任特征的关系图，其中 c_i 表示某个物联网场景。

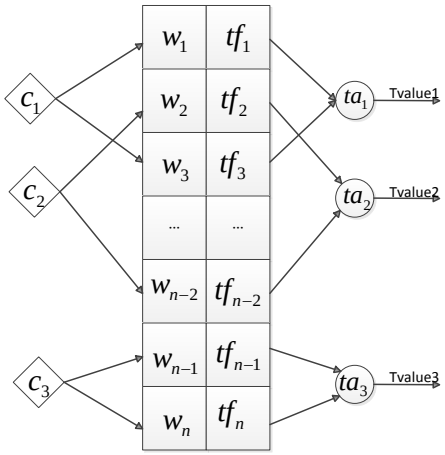


图 2- 7 物联网环境中信任度与信任特征的关系图

2.4 安全模型

在物联网环境中[40]，安全的交互协议可以保证通信双方的安全，并可以有效解决通信双方交互过程中的身份认证、消息的完整性验证、抗抵赖性等问题。虽然安全协议可以实现信息的安全传递，但其中仍然存在着一些漏洞和问题，并不一定能达到人们预期的结果，因此需要对协议的安全性进行分析。形式化分析方法是当前主流的安全协议分析方法，其分为两个主要的演化方向：基于逻辑符号的形式化分析和基于计算复杂度的形式化分析。基于逻辑符号的形式化分析是基于简单有效的形式化语言的方法，对公理的应用是基于定理证明技术或状态搜索技术，其已形成三大类形式化分析方法，包括攻击构造法、推理构造法和证据构造法；基于计算复杂度的形式化分析是基于一个详细的计算模型，其安全性是通过构造一个归约为矛盾类型的证明得到的。这里的矛盾通常是指密码学中的一个困难问题的解，如果困难问题难以攻破，那么可以保证协议的安全性，这种方法又称为可证明安全。

可证明安全法[41]是一种基于计算复杂度的协议分析方法，其通过估计攻击者攻击协议成功的概率和计算代价来定义协议的安全性。可证明安全使用一种“归约”的方法来证明协议安全性：首先需要确定设计的方案应该满足的安全目标，然后再根据攻击者的能力定义一个敌手模型，最后将攻击者攻击系统的方法转化到密码学中解决某个困难问题上。可证明安全方法中常用的模型包括：CK 模型、BCP 模型和 UC 模型。

Canetti 和 Krawczyk[42]提出了一种适用于认证通信安全的可证明安全模型，其利用模块化的方法分析和设计认证密钥协商协议。CK 模型通过不可区分性来说明协议的安全性，即在允许的攻击能力下，攻击者在概率多项式时间内不能区分任意独立的随机参数和由协议产生的会话密钥。基本思路是首先将分析的协议定义在一个理想的模型中，

通过计算不可区分性来证明理想模型的安全，然后将理想模型中的协议转换成现实模型中的协议，最后证明在现实模型中的攻击者可以仿真理想模型中的攻击者，仿真的含义是指攻击者在现实模型中攻击的结果和在理想模型中攻击的结果是一样的。在 CK 模型中定义了非认证链路攻击模型 UM 和认证链路攻击模型 AM 两种攻击模型，这两种攻击模型的唯一区别是攻击者对两个主体间通信链路的控制程度不同。UM 模型类似于现实世界，其中攻击者可以控制整个在使用的网络，而 AM 模型是 UM 模型的受限版本，AM 模型中的攻击者只能被动的运行协议，攻陷通信双方，查询、暴露和测试会话密钥，并且只可以单纯的传送同一消息一次，不能进行消息重放、修改和伪造，而 UM 模型中攻击者不仅涵盖所有 AM 模型中的所有攻击方式，还可以对消息进行重放、修改和伪造。

2.5 本章小结

本章主要对在物联网设备共享中实现可保护隐私的委托授权方案的相关技术进行介绍。首先介绍了委托授权的基本概念、代理签名技术以及经典的 M-U-O 代理签名方案；其次介绍了隐私保护方面的相关技术，包括零知识证明，直接匿名认证方案和基于嵌入式设备的直接匿名认证方案，并对这些协议的优缺点进行分析；然后介绍基于信任度的访问控制的相关知识，包括基于信任度的访问控制模型、信任的概念以及计算信任度的形式化方法；最后介绍了分析协议安全性的安全模型，并主要介绍了 CK 模型。

第三章 基于信任度的委托授权和访问控制

在由中间代理授权的物联网设备共享场景中设备的权限管理包括两方面：第一，设备所有者对中间代理的委托授权；第二，物联网设备对获取权限用户的访问控制。针对设备所有者委托授权时的权限敏感度保护问题，本节在 M-U-O 代理签名方案的基础上进行改进，提出了一中基于信任度的可控的部分权限委托授权机制，设备所有者根据其对中间代理的信任度委托其相应范围的授权权限，分析表明该方案可以提供权限敏感度保护并满足权限委托传递时所需的安全属性；然后针对中间代理过度授权的问题，提出了一种基于信任度的访问控制机制，该机制采用基于角色和信任度的访问控制模型，通过根据用户动态社交关系生成的信任度和设备所有者设定的信任度阈值确定是否向访问用户授权，该机制可以有效防止中间代理过度授权的问题。

3.1 基于信任度的可控的部分委托授权机制

3.1.1 协议设计

物联网设备共享中常见的场景之一就是通过中间代理授权的物联网设备共享平台，在该场景下设备所有者将物联网设备委托给中间代理，由中间代理负责向用户授予权限，用户利用中间代理颁发的授权凭证获得设备的使用权，整个交互过程中通过中间代理进行权限的传递，设备所有者无需与用户进行线下的交互。中间代理可以有效解决设备所有者有自己的工作，无法实时的对设备的访问权限进行管理的问题，其可以代替设备所有者对用户的申请作出快速有效的回应，从而减轻设备所有者管理设备的负担，提高了物联网设备共享系统的效率；另外，由于设备所有者通常是向其认识或熟悉的用户分享自己的个人闲置设备，所以可获得使用权的用户范围有限，通过中间代理的集客效应可以扩大用户的范围，提高系统的实用性。

在该物联网设备共享场景下需要考虑的一个重要问题是如何实现设备所有者权限的可控传递，即如何实现设备所有者对中间代理的可控委托授权，在该场景下设备所有者对中间代理的委托授权需要满足以下性质：

- 临时委托：设备所有者对中间代理的委托是有时间限制的，超过委托时间，中间代理的授权能力就会失效；
- 单一性：当设备所有者委托中间代理对用户授权后，设备所有者还是可以对申请权限的用户进行授权；
- 部分委托：由于设备所有者的权限是存在敏感性差异的，设备所有者将依据中间代理的信任度委托其相应范围的授权权限；
- 自己管理：委托关系由有设备所有者直接管理的，其可以撤销行为表现不好的中间代理。

本文设计的方案满足上述的性质。委托授权的密码学实质就是代理签名技术，由中

间代理代替设备所有者对用户签发授权证书。Mambo 等提出的基于部分授权的代理签名方案是物联网权限管理中应用最为广泛的代理签名方案，由于其通信交互复杂度低、运算量小，适用于移动智能终端等轻量型设备。本节在 M-U-O 代理签名方案的基础上进行改进，中间代理的代理签名私钥是由其和设备所有者共同生成，该方案可以有效的实现委托授权的可验证性和不可否认性、授权者的可区分性和授权能力的不可滥用性，确保了设备所有者权限的可控安全传递。

下面对本文用到的符号进行说明：

$\|$：消息的串联

ID_A：用户 A 的身份标识

T_A：用户 A 生成的时间戳

$Cert_A$：用户 A 的证书

$H(m)$：哈希函数

x_A, y_A：用户 A 的私钥和公钥

$E_k(\cdot), D_k(\cdot)$：消息加密和解密

$Sign(x,m)$：私钥 x 对消息 m 签名

$Verify(y,m,Sign(x,m))$：用公钥验证消息 m 的签名

委托授权协议中的参与者包括设备所有者、中间代理、用户和物联网设备，如图 3-1 所示是委托授权的流程图。

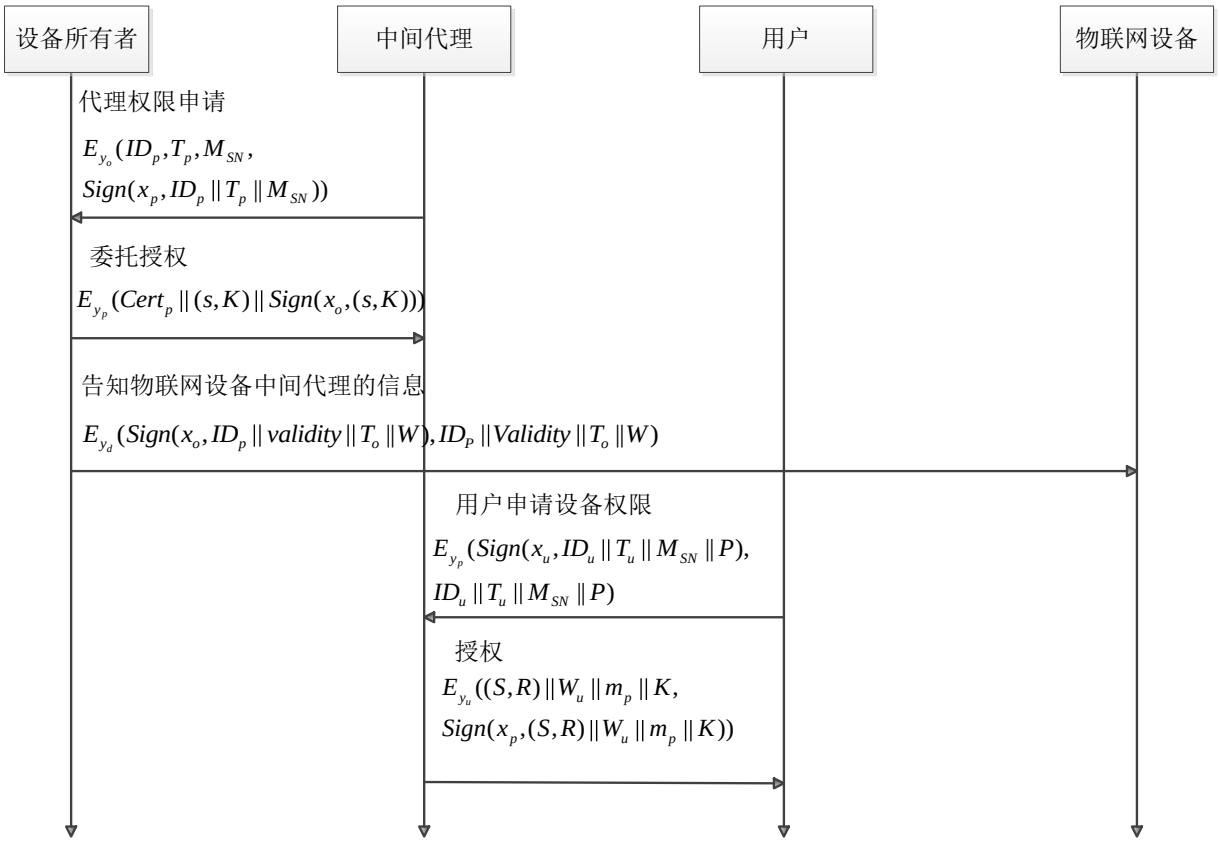


图 3-1 委托授权流程图

接下来对委托授权协议运行的具体步骤进行介绍：

1) 系统参数生成：生成一个大素数 p，q 是 $p-1$ 的素因子，选择 $\alpha \in Z_p^*$ 满足 $\alpha^q = 1 \bmod p$，其中 α， p 和 q 是整个系统的公钥参数。设备所有者、中间代理、用户和物联网设备分别选择随机数 $x_o, x_p, x_u, x_d \in Z_q^*$，其相应的公钥为 $y_o = \alpha^{x_o} \bmod p$，$y_p = \alpha^{x_p} \bmod p$，$y_u = \alpha^{x_u} \bmod p$ 和 $y_d = \alpha^{x_d} \bmod p$。

2) 中间代理向设备所有者申请代理权。中间代理用其私钥 x_p 对其身份标识 ID_p、社交信息 M_{SN} 和时间戳 T_p 进行签名生成数字签名 $Sign(x_p, ID_p \| T_p \| M_{SN})$，并用所有者的公钥 y_o 加密生成密文 $E_{y_o}(ID_p, M_{SN}, T_p, Sign(x_p, ID_p \| M_{SN} \| T_p))$ 发送给所有者进行设备代理权的申请。所有者收到申请后用其私钥 x_o 解密得到申请信息，验证该中间代理的身份是否合法以及时间戳的新鲜性。设备所有者搜索他的中间代理表查看当前是否有用户已经获取该设备的代理权，如果当前没有中间代理占用该设备则所有者同意该用户的代理申请，然后根据该中间代理的社交信息计算对其的信任度，并根据信任度授予其相应的授权范围。设备所有者生成对中间代理的授权证书：$Cert_p = (ID_o \| validity \| T_o \| W, Sign(x_o, ID_o \| validity \| T_o \| W))$，其中 ID_o 是所有者的身份标识，$validity$ 是所有者设置的代理权的有效时间，T_o 是所有者颁发证书的时间，W 是所有者的授权内容，其中包括授予中间代理的角色及对应的权限和对他的信任度值。所有者在他的中间代理表上登记获得授权的中间代理的身份标识、代理有效时间、授权时间和授权内容等，从而可以方便的对具有不良行为的中间代理进行追踪。最后，所有者将获得代理权的中间代理的身份、代理有效期、授权时间等发送给物联网设备，方便物联网设备对中间代理的身份进行验证。

3) 设备所有者选择随机值 $k \in Z_q^*$，计算 $K = \alpha^k \bmod p$ 和 $s = x_o + kK \bmod q$，生成代理签名密钥 (s, K)。设备所有者对 (s, K) 进行签名，并将授权证书 $Cert_p$，(s, K) 和签名 $Sign(x_o, (s, K))$ 加密生成密文 $E_{y_p}(Cert_p, (s, K), Sign(x_o, (s, K)))$ 发送给中间代理，通知中间代理其已经获得授权。

4) 中间代理收到所有者发送的授权消息，得到代理签名密钥 (s, K)，首先验证等式 $\alpha^s = y_o K^K \bmod p$ 是否成立，其中 y_o 代表设备所有者的身份，若成立，则中间代理就确认 (s, K) 是由设备所有者生成的代理签名密钥，否则中间代理拒绝接受所有者的授权；中间代理用所有者的公钥 y_o 验证授权证书的有效性，并对授权证书进行解析获取他的授权范围。

5) 中间代理社交关系网络中想要访问设备的用户将其身份标识 ID_u，社交信息 M_{SN}，申请的权限 P 和时间戳 T_u 发送给中间代理。中间代理对用户的身份进行验证，判断用户申请的权限是否在自己的授权范围内，若在则依据用户的社交信息 M_{SN} 计算对该用户的信任度。最后生成权限凭证 W_u，其中包括授予用户的角色和信任度。

6) 中间代理同意用户的申请后，计算代理签名私钥 $r = s + x_p y_p \bmod q$，其在设备所有者生成的代理签名私钥 s 的基础上加入自己的私钥信息 x_p，并用 r 其代替 x_p 进行普通的数字签名。本文采用 ElGamal 数字签名方案对 W_u 进行签名，中间代理计算 $m_p = H(ID_o \| y_o \| W_u)$，选择随机值 $t \in Z_q^*$，计算 $R = \alpha^t \bmod p$ 和 $S = t^{-1}(m_p - rR) \bmod q$，$(R,S)$ 是代理签名，其将密文 $E_{y_u}((R,S) \| W_u \| m_p \| K, Sign(x_p,(R,S) \| W_u \| m_p \| K))$ 发送给用户。中间代理在用户列表中记录该用户的身份和对应的权限 (ID_u, W_u)。

7) 用户收到授权信息并验证其有效性。计算代理签名验证公钥 $h = y_O K^K y_P^{y_P} \bmod p$ 和 $m_p' = H(ID_O \| y_O \| W_u)$，验证 $m_p' = m_p$ 是否成立，若成立，则用代理签名验证公钥 h 对收到的代理签名进行验证，判断 $Verify(h, W_u, (S,R)) = 1$，也即 $\alpha^{m_p} = R^S h^R \bmod p$ 是否成立，若该等式成立，则用户认为该中间代理的身份时合法的，并接受授权凭证。

3.1.2 协议安全性分析

委托授权过程的本质就是代理签名技术的应用，而一个安全的代理签名方案需要满足可验证性、不可否认性、可区分性、可识别性和不可滥用性等安全属性，本文将根据这些属性对所有者委托授权协议的安全性进行分析。

1) 可验证性

可验证性是指用户根据系统的公开参数和代理签名对代理签名的正确性以及中间代理的身份的有效性进行验证。本方案中验证等式 $Verify(h, W_u, (S,R)) = 1$ 是否成立，如果成立则说明代理签名是有效的且中间代理的身份是合法的。用户在验证代理签名时使用的验证公钥 h 中既包含所有者的公钥 y_o，又包含中间代理的公钥 y_p，并且他们是满足特定关系的，也就是说验证代理签名的公钥只有在设备所有者和中间代理的共同授权之下才可以生成，因此用户会相信该代理签名是合法的。ID_o 和 y_o 是设备所有者的公开信息，用户已经事先知道，他收到中间代理的授权信息后首先验证 $m_p = H(ID_o \| y_o \| W_u)$ 是否成立，若成立则对代理签名的有效性进行验证，证明如下：

$$h = y_o K^K y_p^{y_p} \bmod p = \alpha^{x_o} \alpha^{kK} \alpha^{x_p y_p} \bmod p = \alpha^{x_o + kK + x_p y_p} \bmod p = \alpha^r \bmod p \tag{3-1}$$

$$V_1 = \alpha^{m_p} \bmod q \tag{3-2}$$

$$V_2 = R^S h^R \bmod p = (\alpha^t)^{t^{-1}(m_p - rR)} \cdot (\alpha^r)^R \bmod p = \alpha^{m_p - rR + rR} \bmod p = \alpha^{m_p} \bmod p = V_1 \tag{3-3}$$

2) 不可否认性

中间代理的代理签名私钥是 $r = s + x_p y_p \bmod q = x_o + kK + x_p y_p \bmod q$，其中同时包含了设备所有者和中间代理的私钥信息，因此设备所有者不能否认是他授权中间代理对消息进行签名，而中间代理也不能否认该代理签名是由其生成的，二者都不能对委托授权的过程做出抵赖。

3) 不可伪造性

代理签名私钥中包含了中间代理的私钥信息，而该私钥只能由中间代理掌握，因此任何人都不能伪造出有效的代理签名，另一方面由代理签名验证公钥：$h = y_o K^K y_p^{y_p} \bmod p = \alpha^r \bmod p$ 求出代理签名私钥 r，需要计算离散对数问题，在本方案中求解离散对数问题是困难的，所以除了中间代理包括设备所有者在内都无法以中间代理的名义生成有效的代理签名。

4) 可识别性

由于在代理签名私钥中包含了中间代理的私钥，因此只有利用中间代理的公钥才能生成正确的代理签名验证公钥，从而对代理签名的有效性进行验证。任何得到代理签名的用户都可以通过中间代理的公钥确定中间代理的真实身份。

5) 不可滥用性

在传统的 M-U-O 方案中可能存在中间代理将代理签名私钥 $s = x_o + kK \bmod q$ 分享给其他用户进行授权的情况，代理签名私钥中只包含设备所有者的私钥信息，因此当出现纠纷时，中间代理可以否认该签名是由他生成的并且可以说是由设备所有者生成的，目的是掩盖滥用代理权限的事实。在本方案中，$r = s + x_P y_P \bmod q = x_o + k \cdot \alpha^k + x_p y_p \bmod q$，中间代理如果要伪造有效的代理签名私钥需要知道 x_o 和 k，这需要通过求解离散对数问题才能得到，而在本方案中求解离散对数问题是困难的。

6) 强密钥依赖性

中间代理签名时的代理签名私钥依赖于设备所有者的私钥和中间代理的私钥。

综上所述，改进的代理方案具有更高的安全性和实现效率，并且计算量小和通信复杂度低。该方案可以实现委托授权时设备所有者权限的可控传递，有效的解决当设备的使用出现纠纷问题时，设备所有者和中间代理相互抵赖的问题；另外该方案还根据由中间代理的社交关系计算的信任度授予其相应的授权范围，保护了设备所有者的敏感权限。

3.2 基于动态社交信任度的访问控制机制

在可委托授权的物联网设备共享场景中，设备所有者将授权使用设备的能力委托给中间代理，由中间代理向用户授权，同时中间代理也可以将授权使用设备的能力委托给下一级中间代理，依次类推，形成了设备权限的多次委托传递。在此场景下存在中间代理过度授权的问题，随着权限委托深度的增加，距离设备所有者越远的用户，其信任度越低且随着用户间社交关系的变化，信任度也会动态变化，然而设备的不同权限是存在敏感度差异的，其中某些权限可能涉及设备所有者的隐私是具有高度敏感性的。因此为了实现设备所有者敏感权限保护，针对不同信任度的用户释放相应的权限就显得尤为重要。本文在文献[13]的基础上提出了一种改进的基于动态社交关系信任度的访问控制机制，该机制采用基于角色和信任度的访问控制模型，通过根据用户动态社交关系所生成的信任度和设备所有者设定的信任度阈值确定是否对访问用户授权，从而可以防止中间代理过度授权的问题。

3.2.1 基于角色和信任度的授权策略

在物联网设备共享系统中，将设备所有者与其权限分离，为设备所有者设置角色，并为不同的角色分配不同的权限如图 3-2 所示，因此角色实质上是一组权限的集合且每个角色对应的权限是存在敏感性差异的。

图 3-2 用户、角色和权限的关系图

本文通过给每个角色对应的权限设置一个关联的信任度阈值来反映权限的隐私程度，信任度阈值越大权限越敏感，只有当某个用户获得该角色且其信任度大于角色对应的权限的信任度阈值时才可以激活该角色。在本方案中，用户不仅可以拥有其被分配的角色的权限，还可以继承角色链中低于其角色的所有角色的权限如图 3-3 所示。

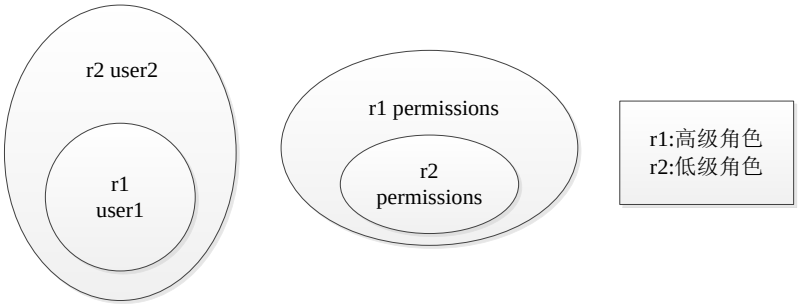


图 3-3 高级角色与初级角色中用户与权限关系的比较

图 3-3 中 r1 是高级角色，r2 是低级角色，r1 可以获得 r2 的所有权限，拥有角色 r1 的 user1 也就自动拥有了角色 r2。因此将上述高级角色与低级角色的关系称为继承关系，然而其中存在的问题是怎样确定高级角色从低级角色中继承来的权限的信任度阈值，如果只是逐一的指定每个继承的权限的信任度阈值是不具有拓展性的，且实现过程很繁琐。本文在角色继承关系中引进了信任度阈值衰减因子的概念，在角色继承关系链中，对于同一个权限，低级角色中该权限关联的信任度阈值通过信任度阈值衰减因子后就获得了高级角色中该权限对应的新的信任度阈值，下面进行具体说明。

模型的基础定义

定义 3.1 用户集$User=\{u_1,u_2,u_3,...\}$：指设备所有者和想要访问设备的用户；角色集$Role=\{r_1,r_2,r_3,...\}$和权限集$Permission=\{p_1,p_2,p_3,...\}$：设备所有者将他的权限进行分散，并将每个权限分配给角色。

定义 3.2 信任度集合$TD=[0,1.0]$，$t\in TD$且是根据 3.3 节中基于用户间的动态社交关系计算而来；信任度阈值集合$TDT=[0,1.0]$；信任度阈值衰减因子集合$TDTD=[0,1.0]$；用户-角色-权限-信任度集合$URPT\subseteq User\times Role\times Permission\times TL$，则$(u,r,p,t)\in URPT$表示用户$u$以信任度大于$t$的信任度得到角色$r$并获得权限$p$；角色链$URL\subseteq Role\times Role$，是由设备所有者的角色形成的树形结构。

用户角色链

用户的角色链是一种树形结构，树中的每一个节点是角色，每个角色按照对应权限敏感度大小由高到低排列，角色链中上级角色可以继承下级角色的所有权限，但是同一级的角色是并列的不可以相互继承。以房屋共享为例对角色链的概念进行说明，表 3.1 说明了房主设定的角色、对应的权限和权限的信任度阈值。

表 3.1 房屋共享角色权限分配

角色	权限	信任度阈值
Guest	p_garage	0.3
General	p_livingroom	0.6
General	p_bathroom	0.7
Close	p_guestroom	0.8
VIP	p_studyroom	0.7
VIP	p_masterroom	0.92

从表中可以看出，房主设定了四种角色，在角色链中由低到高为 Guest，General，Close 和 VIP，其中 General 和 Close 是并列角色。角色 Guest 可以获取房主车库的使用权 p_garage，角色 General 获得客厅 p_livingroom 和卫生间 p_bathroom 的权限，角色 Close 获得客房 p_guestroom 的权限，角色 VIP 获得书房 p_studyroom 和主卧 p_masterroom 的权限。具有角色 VIP 的用户可以继承角色 Guest，General 和 Close 的所有权限。从表中可以看出，主卧是一个高度涉及房主隐私的权限，因此将它的信任度阈值设定的比较高。

角色链上的信任度阈值衰减

以房屋共享为例，在房主的角色链中，角色 VIP 级别最高，General 和 Close 是同等级的角色，没有继承关系，角色 Guest 级别最低。角色 VIP 继承了角色 Close 的所有权限，只获得角色 Close 的用户的信任度必须高于信任度阈值 0.8 才可以获得权限 p_guestroom，那么角色 VIP 继承到的权限 p_guestroom 需要设置多大的信任度阈值呢？通常可以认为对于同一个权限，具有角色级别高的用户比角色级别低的用户更加可信，比如对于房主而言父母比普通朋友更加可信，因此可以认为获得角色 VIP 的用户可以以低于信任度阈值 0.8 的信任度获得权限 p_guestroom，所以通过在角色链中引入信任度阈值衰减因子可以使得高级角色继承来的低级角色的权限的信任度阈值变小是符合实际情况的。如图 3-4 所示是房主设定的角色链中的信任度阈值衰减图，图中角色与角色之间边上的值是信任度阈值衰减因子。

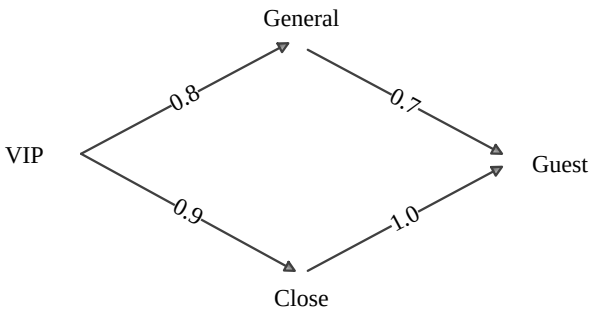


图 3-4 角色链上的信任度阈值衰减

在角色链中高级角色和低级角色之间权限的继承分为直接继承如 VIP → Close 和间

接继承如 VIP → Close → Guest。用函数 $f(r_i, r_{i+1})$ 表示具有直接继承关系的角色之间的信任度阈值衰减因子，而具有间接关系的两个角色间的信任度阈值衰减值是路径中每一段直接关系的信任度阈值衰减因子的累积，函数表示为：

$$faccum(r_n, r_0) = \prod_{i=n-1,\ldots,0} f(r_{i+1}, r_i) \quad r_i \in Role \tag{3-4}$$

例如，间接关系 VIP → Close → Guest 的角色累积衰减值为 0.9 × 1.0 = 0.9。另外当从一个角色到另一条角色有多条路径时，如 VIP→Guest 有 VIP → Close → Guest 和 VIP → General → Guest 两条路径，选择累积衰减值最小的一条路径作为最终的信任度阈值衰减因子。本方案中信任度阈值衰减因子是由设备所有者设置的，如果设备所有者认为某一角色继承来的权限对该角色来说也是比较隐私的，就可以将衰减因子设置的较大，若希望该权限被高等级角色很容易的获得就可以将衰减因子设置的较小。表 3.2 列出了图 3-4 中每一角色获得的权限（包括继承的权限）以及相应的信任度阈值。

表 3.2 角色权限及对应的信任度阈值

角色	权限	信任度阈值
Guest	p_garage	0.30
General	p_garage	0.21
General	p_livingroom	0.60
General	p_bathroom	0.70
Close	p_garage	0.30
Close	p_guestroom	0.80
VIP	p_garage	0.17
VIP	p_livingroom	0.48
VIP	p_bathroom	0.56
VIP	p_guestroom	0.72
VIP	p_studyroom	0.70
VIP	p_masterroom	0.92

从表 3.2 分析可知每个角色都对应多个权限且每个权限都有不同的信任度阈值，所以只要拥有某个角色的用户的信任度值大于多个权限最低的信任度阈值就可以激活该角色。

3.2.2 设备对用户的综合信任度计算

在上节中介绍了房屋共享系统的访问控制方案，房主设置了四种类型的角色，并对每个角色赋予相应的权限，在一个角色链中等级高的角色可以继承等级低的角色的权限，并且为了保护房主敏感度高的权限，对每个权限设置信任度阈值，只要获得该角色的用户的信任度大于对应权限的信任度阈值时就可以激活角色的访问权限。本节将介绍房主对用户的综合信任度的计算方法，根据 2.3 节描述的信任关系的类型，本方案将用户划分为直接关系用户和间接关系用户，本节将讨论包括直接关系用户信任度计算和间接关

系用户综合信任度计算。

直接关系用户信任度计算

直接关系用户与房主认识，其可以与房主直接建立联系，他向房主发出访问请求并将其社交文件和申请的权限 p 发送给房主，房主利用第 3.3 节的方法计算对该用户的社交信任度值 t，然后再根据用户申请的权限和信任度值的大小向其分配角色 r。房主生成授权凭证 $Cert_u = (E_{y_d}(r,t,p,Sign(x_o,r \| t \| p)))$ 发送给直接关系用户，该授权证书中包含房主分配给该用户的角色 r、申请的权限 p 以及对该用户的信任度 t，房主用私钥 x_o 对 (r,p,t) 签名并用设备的公钥 y_d 加密，直接关系用户收到授权凭证后是无法打开授权凭证的，目的是防止用户伪造角色和信任度值。用户将授权凭证转发给房屋的智能锁系统，智能锁系统获得该用户的角色、权限和信任度后，在表 3.2 中检索用户的角色并将其信任度与角色中用户申请的权限的信任度阈值相比较，然后释放权限。

间接关系用户信任度计算

在开放的物联网环境中，想要获取房屋使用权的用户也可能是设备所有者陌生的用户，他无法与房主直接建立联系，但是他可以通过一个或多个“中介”间接的向房主申请权限，因此就形成了一个以房主为核心的社交关系网络。如图 3-5 所示是本文假设的房主社交关系图，图中的节点表示用户，图中通过边相连的用户是相互认识的。

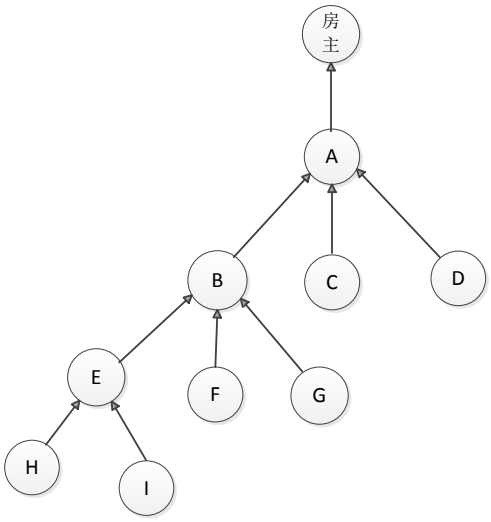


图 3-5 房主的社交关系图

图 3-5 中，用户 B、C、D 通过用户 A 与房主建立联系，用户 E、F、G 通过用户 B 再通过用户 A 与房主建立联系，用户 H、I 通过用户 E，再通过用户 B、用户 A 与房主建立联系。从而就分别形成了以用户 A、B、C 为中心的代理域，相当于形成了房主权限的多级代理传递，其中用户 A、B、C 称为各自域的管理员，房主本身也形成一个管理域，他们域中的成员是各自下一级的用户。每个域管理员只能对其域内的成员颁发授权凭证 $Cert_{p\to u} = (E_{y_d}(r,t,p,Sign(x,r \| t \| p)))$ 表明该成员获得的角色、权限和信任度，其中域内成员也不能对授权凭证进行解析。房主域外的用户，即不与房主直接关联的用户是无法得到房主颁发的授权凭证的，间接用户得到的是一个授权凭证链，其中包含了间

接用户到房主的关系链中中间每个域管理员向其成员颁发的授权凭证，例如用户 H 想要获得房屋的某个权限，其获得的授权凭证链是 $(Cert_{M\to A}, Cert_{A\to B}, Cert_{B\to E}, Cert_{E\to H})$，其中的信任度是由上级用户根据与下级用户的社交相似性计算而来。间接用户获得的授权凭证链是一个简单的线性结构，其综合信任度是由房屋的智能锁系统根据证书凭证链中每个凭证中包含的信任度值综合计算而来。设函数 *trust* 计算两个直接关系用户间的基于动态社交关系的信任度，用函数 *path* 计算关系链 $u_n \leftarrow u_0$ 中 u_0 对 u_n 的信任度，其中包含的直接关系为 $u_n \leftarrow u_{n-1}$，$u_{n-1} \leftarrow u_{n-2}$，...，$u_1 \leftarrow u_0$。

$$path(u_n \leftarrow u_0) = \begin{cases} trust(u_1 \leftarrow u_0) & n = 1 \\ f(path(u_n \leftarrow u_{n-1}), trust(u_{n-1} \leftarrow u_0)) & n > 1 \end{cases} \tag{3-5}$$

f 是房屋智能锁系统综合计算用户信任度的函数，其应满足两个条件：第一点，对于函数 f，输入任何合法的信任度后，输出的是新的合法的可信度，即对于 $\forall t_1, t_2 \in [0,1]$，满足 $f(t_1, t_2) \in [0,1]$；第二点，f 输出的信任度不大于输入的任意的信任度，即随着权限传递的深度增加，获得权限用户的信任度是衰减的。因此，满足上述条件的 f 函数有乘法算法和取最小值算法。对于乘法算法，智能锁系统将用户授权凭证链中的每个凭证中的信任度相乘；而对于取最小值算法，则选择授权凭证链中最小的信任度值作为间接用户的综合信任度。

下面通过一个例子说明间接用户综合信任度的计算方法，基于图 3-5 的社交关系图，用户 C、F、H 获得房主 VIP 角色的场景如下：

House:
House.VIP ← Owner.friend with 1.0　　//设备所有者的朋友可以获得 VIP 角色
Owner.friend ← UserA with 0.95　　//用户 A 是设备所有者的朋友
Owner.friend ← UserA.allow with 0.93　　//设备所有者委托用户 A 允许新用户使用设备的能力
UserA:
UserA.allow ← UserB with 0.87　　//用户 A 允许用户 B 使用设备
UserA.allow ← UserB.allow with 0.86　　//用户 A 委托用户 B 允许新用户使用设备
UserA.friend ← UserC with 0.97　　//用户 C 是用户 A 的朋友
UserB
UserB.allow ← UserE with 0.91　　//用户 B 允许用户 E 使用设备
UserB.friend ← UserF with 0.95　　//用户 F 是用户 B 的朋友
UserE：
UserE.friend ← UserH with 0.96　　//用户 H 是用户 E 的朋友

图 3-6 所示是上述场景的示例图，该场景中上级用户对下级用户的信任度是根据第 3.3 节中基于用户间的动态社交关系计算而来，而上级用户委托下级用户推荐新用户使用设备的能力的信任度是根据两者之间的直接社交关系信任度大致推算而来的，一般比两者间的社交信任度小。

用户 C、用户 F 和用户 H 想要获得房屋的 VIP 角色取决于用户到 Owner.friend 路径中最终取得的信任度，也就是取决于用户到 Owner.friend 路径中每步委托的信任度。例如，Owner.friend←UserH 的路径包括 Owner.friend←UserA.allow←UserB.allow←UserE←UserH，可以看出路径中每条边的类型均为 $A.r \leftarrow e$，其表示将 A 中的角色 r 授予 e，e 可以是用户实体或角色等。当 f 函数是乘法运算时，每步委托的信任度对用户综合信任度的影响都是相同的，路径中任何一次的信任度较低都会使得用户获得角色的综合信任度较低，从而用户很难获得角色中敏感度较高的权限。

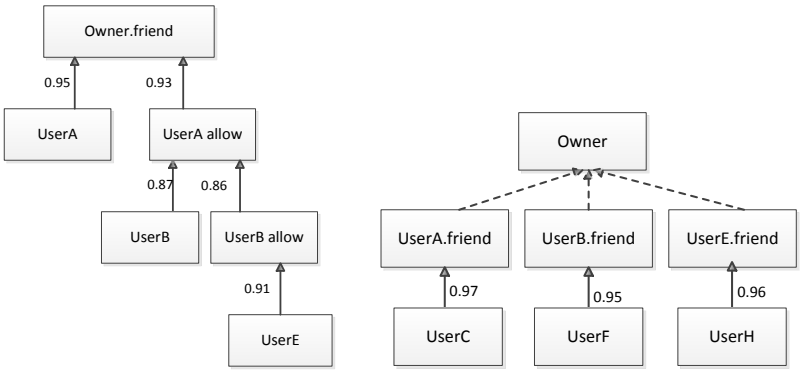


图 3-6 场景示例图

下面用乘法运算综合计算用户 C、F、H 获得角色 Owner.friend 的信任度。

$$\begin{aligned} path(owner.friend \leftarrow UserC) &= trust(UserA \leftarrow UserC) \bullet trust(Owner \leftarrow UserA) \\ &= 0.97 \times 0.95 = 0.9215 \end{aligned} \tag{3-6}$$

$$\begin{aligned} path(owner.friend \leftarrow UserF) &= trust(UserB \leftarrow UserF) \bullet trust(UserA.allow \leftarrow UserB) \bullet \\ &\quad trust(Owner \leftarrow UserA.allow) \\ &= 0.95 \times 0.87 \times 0.93 = 0.77 \end{aligned} \tag{3-7}$$

$$\begin{aligned} path(owner.friend \leftarrow UserH) &= trust(UserE \leftarrow UserH) \bullet trust(UserB.allow \leftarrow UserE) \bullet \\ &\quad trust(UserA.allow \leftarrow UserB.allow) \bullet trust(Owner \leftarrow UserA.allow) \\ &= 0.96 \times 0.91 \times 0.86 \times 0.93 = 0.69 \end{aligned} \tag{3-8}$$

表 3.3 用户取得 VIP 角色的信任度

用户	角色	信任度值
C	House.VIP	0.92
F	House.VIP	0.77
H	House.VIP	0.69

表 3.3 是用户 C、F、H 获得 VIP 角色的信任度，根据表 3.1 可以知道只有用户 C、F 可以获得角色 VIP，用户 C 可以获得全部的权限，用户 F 可以获得除 p_masterroom 权限以外的全部权限，而用户 H 无法获得角色 VIP 的直接分配的权限。

综上所述，用户最终获取角色的信任度值取决于权限传递链中每一段直接关系间的信任度，且权限越往下传递，用户的信任度是衰减的，离房主关系越远的用户获得房屋角色 VIP 权限的可能性就越低。本方案实现了物联网设备共享环境中房主权限的可控传递，同时针对房主权限存在隐私性的问题，实现了对不同信任度的用户的访问控制，保

护了房主具有高敏感度的权限。同时，用户间的信任度不是固定不变的而是在一段时间内随着用户间的社交关系变化而动态变化的，直接或间接用户最终获得的信任度也是动态变化的，因此，用户在不同时间登录房屋共享系统时可能获得不同的访问权限。

3.3 一种基于动态社交关系的信任度生成方案

3.3.1 社交关系表示

在物联网设备共享系统中，设备所有者、中间代理和想要访问设备的用户形成了一张社交关系网，用户或者与所有者认识直接申请权限或者通过某个中间代理间接的与所有者建立联系申请权限，因此在该社交关系网中直接建立联系的用户间是相互认识、彼此间有共同的交集，否则当两个用户之间是陌生、没有交集的时候是不能产生信任的。由于影响用户间关系的因素很多且有些因素是不能定量的表示的，所以无法综合每一点可变因素计算信任度，因此本文选取三个比较有代表性的特征对用户间的社交关系[44]进行描述，分别是联系人、社交联系地址和共同兴趣：

- 联系人（Contact persons）：通过用户间拥有的共同联系人的数量表示用户间的亲密度；
- 社交联系地址（Social Contact）：通过用户间共同的用于社交联系的地址信息表示用户双方有相同的联系频繁紧密的联系人；
- 共同兴趣（Community of Interest）：通过用户双方对同一事物的兴趣和接受程度表现彼此间的熟悉程度，比如两个用户使用相同的手机 APP 很多在某种程度上表示两人是志趣相投的。

当前，人们几乎可以利用移动智能手机进行一切社交活动，因此本文社交信息的获取都是来源于用户的移动智能手机。每个用户在其智能手机中存储一份他本人的社交文件如图 3-7 所示，其中记录了他个人的社交信息，用户会定时的对其中的信息进行更新。当用户 B 想要获取用户 A 的权限时，用户 B 将他的社交信息发送给用户 A，用户 A 根据社交信息计算对用户 B 的信任度。

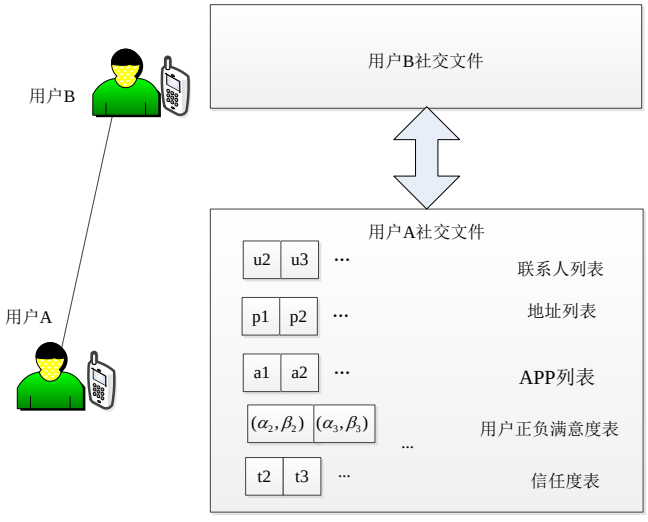


图 3-7 用户社交文件

下面对用户 u_x 的社交文件进行具体的介绍：

社交文件中包括联系人列表、社交联系地址列表、共同兴趣列表、用户正负满意度表和信任度值表：

1) 联系人列表（Contacts List）：记录用户 u_x 当前所有联系人的身份信息，用集合 $F_x=\{u_a,u_b,...\}$ 表示；
2) 地址列表（Location List）：记录用户 u_x 频繁用于社交联系的地址信息，用集合 $P_x=\{p_{x,1},p_{x,2},...\}$ 表示；
3) 共同兴趣列表（COI List）：记录用户 u_x 经常使用的手机 APP 名称，用集合 $A_x=\{a_{x,1},a_{x,2},...\}$ 表示；
4) 用户正负满意度列表（User Satisfaction Experience List）：记录了用户 u_x 对其他用户的满意度与不满意度，用集合 $B_x=\{(\alpha_{x,i},\beta_{x,i}),(\alpha_{x,j},\beta_{x,j}),...\}$ 表示，其中 $(\alpha_{x,i},\beta_{x,i})$ 表示用户 u_x 对用户 u_i 的满意度与不满意度；
5) 信任度列表（Trust List）：记录用户 u_x 对其他用户的信任度值，用集合 $T_x=\{t_{x,i},t_{x,j},...\}$ 表示，其中 $t_{x,i}$ 表示用户 u_x 对用户 u_i 的信任度。

3.3.2 信任度计算

本节将基于用户间的社交关系和 β 分布计算信任度值，首先对相关的基本定义进行介绍。

定义 3.3[45] 先验概率（Prior Probability）：是指事情还没有发生时，根据过去的知识和经验分析得到的概率，其中客观先验概率是指根据过去的历史记录计算出的先验概率，而主观先验概率是指没有历史资料，只是依靠主观经验计算出的先验概率。

定义 3.4[45] 后验概率（Posterior Probability）：是指在通过一定方式获取了新的附加信息后，用贝叶斯公式修正先验概率而得到的概率。

定义 3.5[45] 共轭分布（Conjugacy）：β 分布与二项分布是共轭先验的，即先验分布是 β 分布，则后验分布也是 β 分布。

定义 3.6[46] β 分布（Beta Distribution）可以理解为二项分布中参数 p 的概率分布，其分布密度函数为：

$$Beta(p\mid a,b)=\begin{cases}\dfrac{1}{B(a,b)}p^{a-1}(1-p)^{b-1}, & 0\le p\le 1\\ 0, & \text{其它}\end{cases} \tag{3-9}$$

其中 $B(a,b)=\int_0^1 p^{a-1}(1-p)^{b-1}dp \quad (a>0,b>0)$ 为 Beta 函数。

β 分布的期望值为 $E[p]=\dfrac{a}{a+b}$，方差为 $D(p)=\dfrac{ab}{(a+b+1)(a+b)^2}$。

通俗的理解，β 分布可以看成是一个概率的概率分布，其定义域为(0,1)。当不知道

某一件事发生的具体概率时，β 分布可以给出所有概率出现的可能性。例如，在运动员打球前，要对其击中球的概率进行预测，假设已知该运动员击中球的历史记录，可以用二项分布表示运动员击球的一系列的成功或失败。可以基于先验知识（运动员击中球的历史记录）用 β 分布对击中球的先验概率进行估计，表示为 $Beta(a,b)$，a,b 分别表示击中与没击中的个数，从而可以在赛前对运动员击中球的概率有个大致了解。当运动员打球后，根据每次击球的结果对先验信息进行更新从而获得后验概率分布，由于 β 分布是共轭分布的，因此运动员击中球的后验概率分布也满足 β 分布，其分布形式为 $Beta(a+N_1,b+N_2)$，其中 N_1,N_2 分别表示击中与没击中的个数。

上述利用 β 分布估计击中率的方法也可以用于估计用户的信任度值，一个用户首先基于先验知识也就是另一个用户的社交文件计算对该用户的先验信任度，再根据每段时间内对该用户行为表现的满意值评估对先验信任度进行更新，具体计算过程如下所述：

用户 u_x 用 $f_{x,i}$ 表示对用户 u_i 的主观满意值，$f_{x,i}=1$ 表示 u_x 对 u_i 满意，$f_{x,i}=0$ 表示 u_x 对 u_i 不满意，由此可以将 u_x 对 u_i 的满意值 $f_{x,i}$ 看成一次独立的伯努利试验的结果且其满足二项分布。同时 u_x 对 u_i 的综合满意度 $\theta_{x,i}$ 的先验概率分布满足 β 分布，表示为 $Beta(\alpha_{x,i},\beta_{x,i})$，其中 $\alpha_{x,i}$ 和 $\beta_{x,i}$ 表示满意度与不满意度的先验知识。根据共轭分布的性质，综合满意度 $\theta_{x,i}$ 的后验概率分布也满足 β 分布 $Beta(\alpha_{x,i}^{(now)},\beta_{x,i}^{(now)})$，可以利用 u_x 对 u_i 的主观满意值 $f_{x,i}$ 对先验概率进行更新，表示如下：

$$\alpha_{x,i}^{(now)} = e^{-\varphi\Delta t}\cdot\alpha_{x,i}^{(before)} + f_{x,i} \tag{3-10}$$

$$\beta_{x,i}^{(now)} = e^{-\varphi\Delta t}\cdot\beta_{x,i}^{(before)} + 1 - f_{x,i} \tag{3-11}$$

等式 3-10 和 3-11 表示 u_x 每隔一段时间 Δt 对 u_i 的满意值 $f_{x,i}$ 进行评估，并且用指数衰减因子 $e^{-\varphi\Delta t}$ 分别乘以前一个 Δt 时间段内 u_x 对 u_i 的满意度与不满意度 $\alpha_{x,i}^{(before)}$ 和 $\beta_{x,i}^{(before)}$，从而对当前的 $\alpha_{x,i}$ 和 $\beta_{x,i}$ 进行更新，可以看出当前的满意度与过去的满意度以及当前的主观满意值 $f_{x,i}$ 有关。

u_x 对 u_i 的直接信任关系信任度 $t_{x,i}^d$ 用综合满意度 $\theta_{x,i}$ 的期望值表示：

$$t_{x,i}^d = E[\theta_{x,i}] = \frac{\alpha_{x,i}}{\alpha_{x,i}+\beta_{x,i}} \tag{3-12}$$

将 $\alpha_{x,i}$ 和 $\beta_{x,i}$ 的初始值分别设置为 $sim(u_x,u_i)$ 和 $1-sim(u_x,u_i)$，其中 $sim(u_x,u_i)$ 是依据社交文件中 u_x 和 u_i 的联系人、社交联系地址和共同兴趣的相似性计算出的余弦相似度，$1-sim(u_x,u_i)$ 表示非相似度，$sim(u_x,u_i)$ 的定义域是[0,1]。

下面分别从用户间联系人的余弦相似度、社交联系地址的余弦相似度和共同兴趣余弦相似度三个方面对整个社交关系的相似度 $sim(u_x,u_i)$ 进行估算：

1) 联系人相似度 sim_f：

用户 u_x 获得 u_i 的联系人列表信息，将每个联系人作为一个节点标在网格坐标系中，分别计算二元向量 $\overrightarrow{VF_x}$ 和 $\overrightarrow{VF_i}$，u_x 和 u_i 两者所有的联系人数量为 $|F_x \cup F_i|$，当某个联系人在 F_x 或 F_i 中时 $\overrightarrow{VF_x}$ 或 $\overrightarrow{VF_i}$ 为 1，否则就为 0。用 $\|\overrightarrow{VF_x}\|$ 表示向量 $\overrightarrow{VF_x}$ 的范数，$|\overrightarrow{VF_x}|$ 表示向量 $\overrightarrow{VF_x}$ 的模，通过计算向量 $\overrightarrow{VF_x}$ 和 $\overrightarrow{VF_i}$ 之间的余弦角度表现两者之间联系人的余弦相似度，如图 3-8 所示。向量间的角度越小，余弦值也就越大，相似度也就越大，关系就越亲密。选择余弦函数计算相似性是因为它计算简单适用于计算能力有限的轻量型设备。

$$sim_f(u_x,u_i)=\frac{\overrightarrow{VF_x}\cdot\overrightarrow{VF_i}}{\|\overrightarrow{VF_x}\|\|\overrightarrow{VF_i}\|}=\frac{|F_x\cap F_i|}{\sqrt{|F_x|\cdot|F_i|}} \tag{3-13}$$

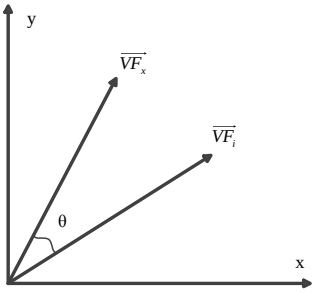


图 3-8 向量夹角图

2) 社交联系地址相似度 sim_l：

社交联系地址相似度表示两个用户间具有相似的物理联系，比如喜欢联系某个相同的联系人。同理可以计算社交联系地址的余弦相似度：

$$sim_l(u_x,u_i)=\frac{\overrightarrow{VP_x}\cdot\overrightarrow{VP_i}}{\|\overrightarrow{VP_x}\|\|\overrightarrow{VP_i}\|}=\frac{|P_x\cap P_i|}{\sqrt{|P_x|\cdot|P_i|}} \tag{3-14}$$

3) 共同兴趣相似度 sim_a

共同兴趣相似度表示两个用户之间具有相似的兴趣爱好，比如喜欢使用类似的手机 APP。同理可以计算共同兴趣余弦相似度：

$$sim_l(u_x,u_i)=\frac{\overrightarrow{VA_x}\cdot\overrightarrow{VA_i}}{\|\overrightarrow{VA_x}\|\|\overrightarrow{VA_i}\|}=\frac{|A_x\cap A_i|}{\sqrt{|A_x|\cdot|A_i|}} \tag{3-15}$$

用户 u_x 和 u_i 社交关系相似度需要综合上述三个方面的相似度计算：

$$sim(u_x,u_i)=\sum_{k\in\{f,l,a\}} w_k sim_k(u_x,u_i) \tag{3-16}$$

其中 w_k 满足 $w_f+w_l+w_a=1$ 且 $0\le w_f,w_l,w_a\le 1$，w_k 是每一个社交关系所占的权重，由用户 u_x 根据其需求自己确定。直接信任关系信任度 $t_{x,i}^d$ 中 $\alpha_{x,i}$ 和 $\beta_{x,i}$ 的初始值分别设置为 $sim(u_x,u_i)$ 和 $1-sim(u_x,u_i)$。

上述方法只是计算了用户 u_x 对 u_i 的直接信任关系信任度 $t_{x,i}^d$，直接信任关系信任度是用户 u_x 直接根据其本身与用户 u_i 的社交文件相似性计算出的信任度值，其还不能全面

的表现出用户u_i的可信度，因此用户u_x可以通过与用户u_i的共同联系人对u_i的信任度反馈计算出对u_i的间接信任关系信任度$t_{x,i}^r$来进一步完善用户u_i的综合信任度值，具体的计算方法如下：

用户u_x每隔Δt时间向两者间的共同联系人发出请求推荐用户u_i的申请，每个共同联系人u_y收到请求推荐申请后，计算他对用户u_i的直接信任关系信任度$t_{y,i}^d$，并将信任度值发送给u_x。用户u_x选择共同联系人中与其社交文件相似度最高的前k用户的直接关系信任度计算间接信任关系信任度，用户u_x对u_i间接信任关系信任度$t_{x,i}^r$是：

$$t_{x,i}^r = \sum_{u_y \in U} \frac{sim(u_x, u_y)}{\sum_{u_y \in U} sim(u_x, u_y)} \cdot t_{y,i}^d \tag{3-17}$$

式(3-17)中，集合U中是前k个与u_x社交文件相似度最高的用户u_y，$t_{y,i}^d$是u_y对u_i的直接信任关系信任度，每个$t_{y,i}^d$所占的权重值是该共同联系人与u_x社交文件相似度占每个共同联系人与u_x社交文件相似度之和的比重。最后，用户u_x结合其对u_i的直接信任关系信任度与间接信任关系信任度计算出综合信任关系信任度：

$$t_{x,i} = \mu \cdot t_{x,i}^d + (1-\mu) \cdot t_{x,i}^r \tag{3-18}$$

其中，μ是权重参数$(0 \le \mu \le 1)$用来说明直接信任关系信任度相对共同联系人反馈的间接信任关系信任度的重要性，由用户u_x根据需要自己确定。

由于用户的联系人信息、兴趣爱好是在不断变化的，他会定时的更新他的社交文件，同时每隔一段时间用户u_x会对u_i的满意值重新评估，因此用户间的信任度也会随时间和社交关系的变化而自适应的调整。

3.4 本章小结

本章主要介绍了基于信任度的委托授权和访问控制方案，首先针对物联网设备共享场景中设备所有者委托授权时权限敏感度保护的问题，在 M-U-O 代理签名方案的基础上进行改造，提出了一种基于信任度可控的部分权限委托授权机制，中间代理在设备所有者许可范围内将部分被委托的权限授权给用户，其使用的代理签名私钥由设备所有者和中间代理共同生成防止设备所有者和中间代理相互抵赖的问题和提供了权限敏感度保护；针对用户访问物联网设备时中间代理过度授权的问题，提出了一种基于信任度的访问控制机制，该机制采用基于角色和信任度的访问控制模型，根据用户动态社交关系所生成的信任度和设备所有者设定的信任度阈值确定是否授权；提出了一种基于动态社交关系的信任度生成方案，该方案根据用户间联系人、社交联系地址和使用的手机软件的相似性实时的生成用户当前的信任度。该信任度可根据社交关系的变化进行自适应的调整。

第四章 可保护隐私的用户身份认证机制

4.1 整体设计

在可委托授权的物联网设备共享场景中，设备所有者社交关系中的直接关系用户可以直接向其申请权限，其在得到授权凭证后与物联网设备交互进行身份认证；而设备所有者社交关系中的间接用户则通过某个中间代理获得访问设备的授权凭证，由于其与所有者并不认识，因此间接用户不希望设备所有者知道他的真实身份等隐私信息，因为这些信息存储在物联网设备中可能被泄露或被攻击者窃取，从而可能对用户的隐私安全造成极大的威胁。所以间接用户希望设备可以在不知道其真实身份的情况下对其进行身份认证，也就是说在认证的过程中用户对设备来说是匿名的。国内外学者已经提出了一些匿名认证方案，Liu[47]等提出了一种适用于无线移动环境的基于 DAA 认证的匿名身份认证方案，但是该方案中引进了 DAA-CA 作为权威的证书机构和公钥基础设施 PKI（Public Key Infrastructure），而其中的 DAA-CA 会成为整个系统的安全瓶颈，并且方案的整体性能较差。Yang[48]等提出了一个基于知识证明的直接匿名认证方案，该方案可以实现设备间的匿名认证和隐私信息保护，但是该方案认证交互复杂，不适用于无线移动环境。本文针对上述问题，提出了一个基于代理签名和知识签名技术的匿名身份认证机制，其基本思路是通过代理签名的信任委托思想使得物联网设备相信用户来自合法的中间代理域，即中间代理基于部分授权的代理签名方案将其生成的代理签名私钥授予用户，用户利用代理签名私钥对消息签名，物联网设备利用中间代理的公钥生成代理签名验证公钥对代理签名进行验证，确认中间代理身份的合法性，如图 4-1 所示是委托关系图；然后物联网设备再通过用户提交的知识签名在无需用户身份信息的前提下验证该用户是由合法的中间代理授权的合法用户。

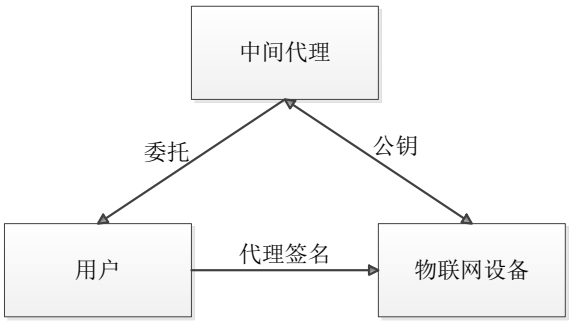


图 4-1 委托关系图

本方案交互过程简单、运算量小，适用于计算能力弱、电池能量有限的物联网设备，更具有实用性。如图 4-2 所示是匿名认证的系统模型，该系统由中间代理、用户和物联网设备组成，系统实现包括两步：

第一步：群加入，已经获得中间代理颁发的授权凭证的用户，向中间代理申请加入中间代理建立的群，中间代理对该用户的身份进行验证，确认该用户是由其授权的合法用户，然后向其颁发群成员证书；

第二步：匿名身份认证，用户利用中间代理生成的代理签名私钥对中间代理的身份信息和公钥信息进行代理签名，并根据群成员证书生成知识签名，物联网设备对代理签名和知识签名进行验证，在不知道用户真实身份的情况下确认该用户是由合法的中间代理授权的合法用户，从而完成对该用户的间接匿名身份认证。

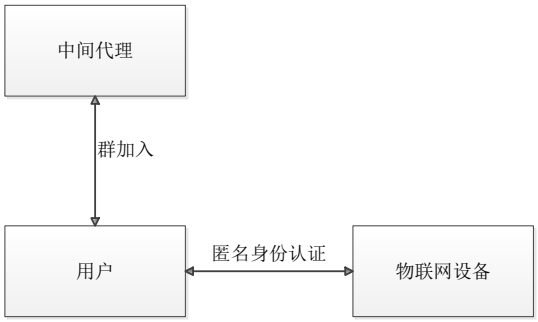


图 4-2 系统模型

4.2 群加入

本文根据群签名方案的思想，用户可以加入中间代理建立的群，作为群管理员的中间代理向用户颁发群成员证书，加入群的用户可以代表中间代理群对消息进行匿名的签名，从而完成与物联网设备的身份认证且不需要交互自己的真实身份等隐私信息。本文采用 CM[33]群签名方案中的成员加入协议获得用户的群成员证书，CM 群签名方案可以实现在不改变群的公开参数的条件下自由的添加新成员，相比其他群证书生成方案效率更高。图 4-3 所示是用户加入群的流程图。

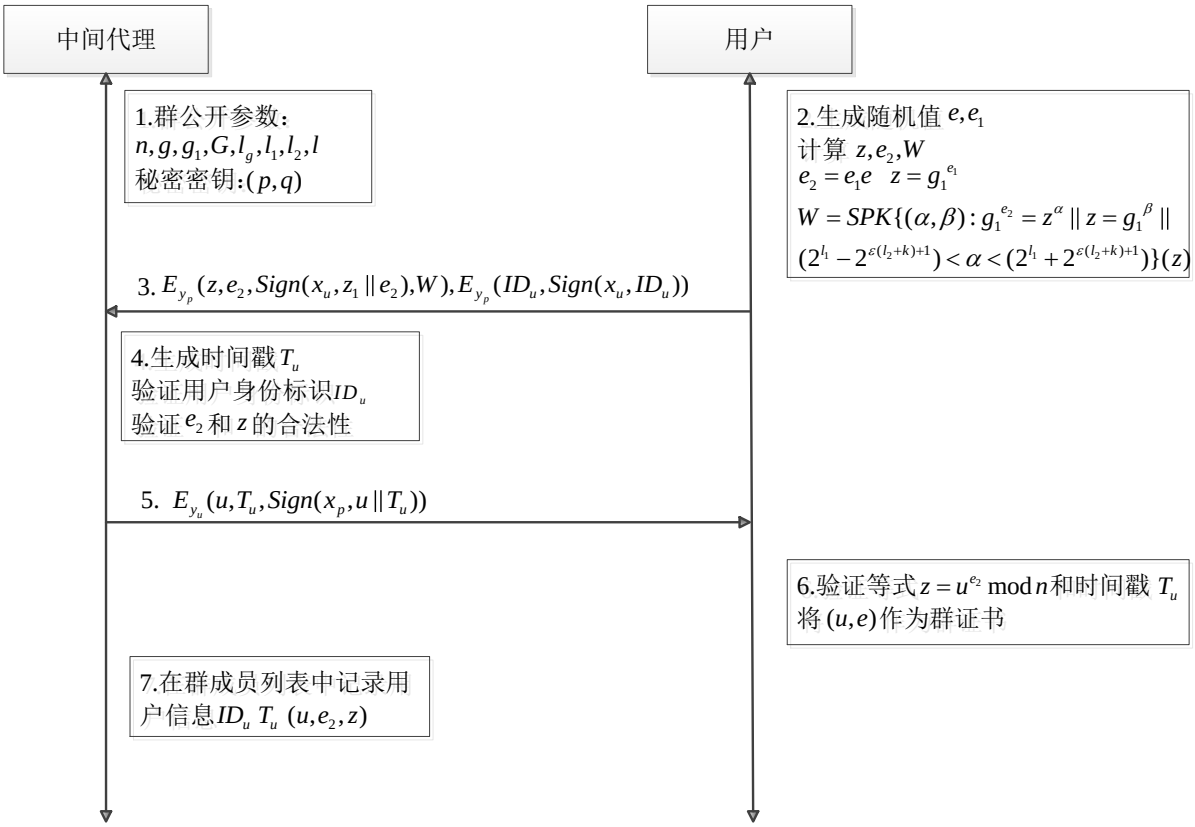


图 4-3 用户加入中间代理群的过程图

4.2.1 群系统参数生成

中间代理建立用户群，需要生成建立群系统的公开参数和秘密参数，中间代理是群管理员，用户是群成员，具体步骤如下：

1) 群管理员中间代理建立群$G=<g>$，群G是群Z_n^*的子群，且满足雅克比符号（Jacobl symbol）$(g|n)=1$，其中$n=pq$，p和q是长度为$l_g/2$的素数，且$p=2p'+1$，$q=2q'+1$，p'和q'也为素数，p和q满足$p,q\neq 1(\mathrm{mod}8)$和$p\neq q(\mathrm{mod}8)$，中间代理生成随机数$g_1\in G$；将(p,q)作为其秘密参数，并且公开n。
2) 中间代理公布群公共参数g，g_1，G和l_g。
3) 中间代理设置安全参数l，l_1，l_2，l_g。

综上所述，群的公开参数是$\{n,g_1,g,G,l_g,l_1,l_2,l\}$，中间代理的秘密密钥是$\{p,q\}$。

4.2.2 群证书生成

群系统参数设置成功后，用户生成秘密值并利用该秘密值生成知识签名，知识签名是一种数学构造，用户利用这种数学构造在不向中间代理泄露秘密值的前提下，证明其拥有该秘密值，从而获取群证书。群成员获取证书的具体步骤如下：

1) 用户首先选择两个随机值$e_1\in_R\{2^{l-1},...,2^{l-1}-1\}$和$e\in_R\{2^{l_1},...,2^{l_1}+2^{l_2}-1\}$，且$e_1$和$e$满足$e_1,e\neq 1(\mathrm{mod}8)$和$e_1\neq e(\mathrm{mod}8)$。
2) 用户计算$e_2=e_1e$和$z=g_1^{e_1}$，并用其私钥x_u对z和e_2签名生成数字签名$Sign(x_u,z\|e_2)$做出承诺。
3) 用户计算知识签名：
$W=SPK\{(\alpha,\beta):g_1^{e_2}=z^{\alpha}\|z=g_1^{\beta}\|(2^{l_1}-2^{\varepsilon(l_2+k)+1})<\alpha<(2^{l_1}+2^{\varepsilon(l_2+k)+1})\}(z)$（$SPK$表示知识签名的符号），用中间代理的公钥$y_p$加密$z$、$e_2$、签名承诺$Sign(x_u,z\|e_2)$和知识签名$W$，生成密文$E_{y_p}(z,e_2,Sign(x_u,z\|e_2),W)$发送给中间代理，最后用户对其身份标识$ID_u$签名并生成密文$E_{y_p}(ID_u,Sign(x_u,ID_u))$发送给中间代理，用户通过知识签名向中间代理[50][51]证明e_2是两个素数e_1和e的乘积，且其知道e_1和e。
4) 中间代理收到用户发送的消息后，首先根据用户的身份标识ID_u判断该用户是否是已获授权的合法用户，若用户身份合法，则验证e_2和z的正确性以及用户是否知道e_1和e。
5) 中间代理计算$u=z^{1/e_2}\bmod n$，生成时间戳T_u记录中间代理向该用户颁发群证书的时间，用于揭露行为表现不好的用户，然后加密u和T_u生成$E_{y_u}(u,T_u,Sign(x_p,u\|T_u))$发送给用户。
6) 用户验证等式$z=u^{e_2}\bmod n$，即$g_1=u^e\bmod n$是否成立，时间戳T_u是否新鲜，如果都正确，则用户将密钥对(u,e)作为其群成员证书，其中u为公钥，e我私钥。
7) 中间代理将该用户的身份标识ID_u，颁发证书的时间戳T_u和(u,e_2,z)一起存储在群成

员列表中。

本文的群成员证书生成方案相较直接匿名认证方案的 CL 群签名协议，交互复杂度减小了，协议的计算量简化了，因此适用于物联网环境下的计算能力有限的移动智能终端和物联网设备。

4.3 匿名身份认证

在上节中用户得到了他的群成员证书(u,e)，其中e是私钥被用户秘密保存，用户随机选择盲化值b将用户的群成员私钥信息进行隐藏，并利用代理签名和知识签名向物联网设备间接的证明他是由合法的中间代理授权的合法用户，如图 4-4 所示是用户与设备匿名认证的流程图。下面对具体的匿名认证过程进行介绍：

1) 首先，设置匿名认证所需的系统参数，其中包括上节中生成用户群证书的部分参数。中间代理选择随机值 p_2 和q_2，p_2，q_2均为大素数且满足$q_2 \mid p_2-1$，选择阶数为q_2的生成元$g_2 \in Z_{p_1}^*$；中间代理选择秘密值$x \in Z_{p_2-1}^*$，计算其公钥为$y=g_2^x \bmod p_2$；设置安全参数U，V，α，l_c，l_s，l_b，其中α，l_c，l_s，l_b的值大于 1，U和V是整数常量，且满足$U>2^{\alpha(l_c+l_b)+1}$，$V>2U+2^{\alpha(l_c+l_s)+2}$；定义强碰撞哈希函数：$H_1:\{0,1\}^* \to Z_n^*$，$H_2:\{0,1\}^* \to \{0,1\}^{l_c}$；综上，系统的公开参数为$\{n,y,g_1,g_2,p_2,q_2\}$，中间代理的秘密值为$\{p,q,x\}$。
2) 中间代理选择秘密值$s \in Z_{p_2-1}^*$，计算$K=g_2^s \bmod p_2$，$\sigma=x+sK \bmod(p_2-1)$得到代理签名密钥(σ,K)，利用用户的公钥y_u对代理签名密钥(σ,K)加密生成密文$E_{y_u}(\sigma,K)$发送给用户。
3) 用户收到密文后用私钥x_u解密得到(σ,K)，再用公开的中间代理的公钥y验证等式$g_2^{\sigma}=yK^K \bmod p_2$是否成立，若成立则用户可以判断该代理签名密钥是由合法的中间代理生成的，否则终止该协议或者要求中间代理重新发送代理签名密钥。
4) 中间代理向其社交关系中获得授权证书的用户颁发群成员证书(u,e)，且满足$u \equiv g_1^e \bmod n$。
5) 中间代理的身份标识ID_p和公钥y是公开参数，用户计算哈希值$h=H_2(ID_p,y)$，用代理签名密钥σ代替s对包含中间代理身份信息的摘要值h进行数字签名。本方案采用 DSA 数字签名方案，用户任意选择随机值$k<q_2$，计算$S_1=(g_2^k \bmod p_2) \bmod q_2$，$S_2=k^{-1}(h+\sigma S_2) \bmod q_2$，$(S_1,S_2)$是对$h$的代理签名。
6) 用户随机选择盲化值$b \in_R \{V-2^{l_b},V+2^{l_b}\}$和随机值$t_1 \in_R \pm\{0,1\}^{\alpha(l_s+l_c)}$、$t_2 \in_R \pm\{0,1\}^{\alpha(l_b+l_c)}$，计算$X_1=u^b \bmod n=g_1^{e^{-1}b} \bmod n$，$X_2=g_1^b \bmod n$，$a_1=X_1^{t_1} \bmod n$，$a_2=g_1^{t_2} \bmod n$；用户计算知识签名$(C,w_1,w_2)$，其中：$C=H_2(g_1 \| X_1 \| X_2 \| a_1 \| a_2 \| h \| S_1 \| S_2 \| K \| W_u)$，$w_1=t_1-C(e-U)$，$w_2=t_2-C(e-V)$，用户利用物联网设备的公钥$y_d$对整个消息

(h,S_1,S_2,K,W_u)加密生成密文$E_{y_d}(h,S_1,S_2,K,W_u)$，并将密文，知识签名和X_1，X_2一起发送给物联网设备进行身份认证。

7) 物联网设备收到消息$(m,S_1,S_2,K,W_u,C,w_1,w_2,X_1,X_2)$后，首先验证用户提交的中间代理身份的合法性，再通过知识签名判断该用户是否是中间代理授权的合法用户。设备所有者在向中间代理委托代理权时已经告知物联网设备获得代理权的中间代理的身份ID_p并且中间代理的公钥y是公开参数。因此物联网设备首先对中间代理的身份进行验证，计算$h'=H_2(ID_P,y)$，判断$h=h'$是否成立，然后计算代理签名验证公钥$y'=yK^K \bmod p_2$并用该公钥验证代理签名的正确性；计算$f=(S_2)^{-1} \bmod q_2$，$h_1=(h\cdot f) \bmod q_2$，$h_2=(S_1\cdot f) \bmod q_2$，$v=[(g_2^{h_1}(y')^{h_2}) \bmod p_2] \bmod q_2$，如果等式$v=S_1$成立，则物联网设备可以认为用户发送的中间代理的身份信息是正确的。

8) 物联网设备验证知识签名(C,w_1,w_2)的正确性。物联网设备根据收到的消息计算C值，$C'=H_2(g_1 \| X_1 \| X_2 \| X_1^{w_1-CU}X_2^{C} \| g_1^{w_2-CV}X_2^{C} \| h \| S_1 \| S_2 \| K \| W_u)$，其中：$X_1^{w_1-CU}X_2^{C}=a_1$，$g_1^{w_2-CV}X_2^{C}=a_2$。当且仅当$C=C'$，$w_1 \in \pm\{0,1\}^{\alpha(l_s+l_c)+1}$，$w_2 \in \pm\{0,1\}^{\alpha(l_b+l_c)+1}$三者同时成立时，物联网设备认为知识签名$(C,w_1,w_2)$是有效的，该用户拥有中间代理颁发的群成员私钥，从而物联网设备相信该用户是经过合法中间代理授权的合法用户。

9) 物联网设备完成对用户的匿名身份认证后打开授权凭证W_u，授权凭证W_u中包含该用户到设备所有者之间所有上级用户对其下级用户颁发的授权凭证，每个授权凭证的形式都是$Cert=(E_{y_d}(ID_p \| r \| t \| p, Sign(x_p, r \| t \| ID_p \| p)))$。物联网设备首先用私钥$x_d$对消息解密，然后用中间代理的公钥$y_p$对签名进行验证，确认角色$r$和信任度值$t$没有被用户篡改。物联网设备将每个授权凭证中的信任度值相乘计算出该用户的综合信任度，然后检索访问控制表中用户获得角色对应的权限，并判断用户的信任度是否大于该角色对应权限的信任度阈值，然后释放权限。

10) 物联网设备中维护了一张用户撤销列表，当某个用户被攻击者攻陷后，他的密钥可能被泄露了，所以物联网设备需要识别来自假冒用户的身份认证请求。为了实现对假冒用户的检测，被攻陷的用户广播他已被攻陷的消息，设备将该用户的密钥信息(y_u,u,e)登记在用户撤销列表中，每次有用户要访问设备时，设备首先利用撤销列表中每个登记的用户密钥计算$X_1^e=X_2 \bmod n$是否成立，若成立则可识别出伪装成已撤销用户的攻击者。另外，中间代理生成用户的代理签名密钥(σ,K)时将K与用户的身份标识ID_u绑定，因此当中间代理看到被攻陷用户广播的消息后，就可以通过K识别出该用户的身份，从而撤销对该用户的授权和用户的群成员身份，然后广播消息说明K已经失效，其相应的代理签名也就失效了，最后设备更新他的用户撤销列表为(y_u,u,e,σ,K)。

本方案基于代理签名实现了中间代理对用户的信任委托，其实质就是使得物联网设

备相信该用户是由设备所有者授权的中间代理授权的；为了实现用户与设备间的匿名身份认证，用户利用随机选择的盲化值 b 和群证书 (u,e) 计算 X_1 和 X_2，并构造出知识签名 (C,w_1,w_2) 向物联网设备证明 (X_1,X_2) 是由合法的群密钥对构建而来，并且用户的密钥信息都隐藏在了 X_1 和 X_2 中。物联网设备验证了知识签名的正确性后，在不知用户真实身份的情况下相信该用户确实拥有合法的中间代理颁发的群证书，实现了用户与设备间的匿名身份认证。

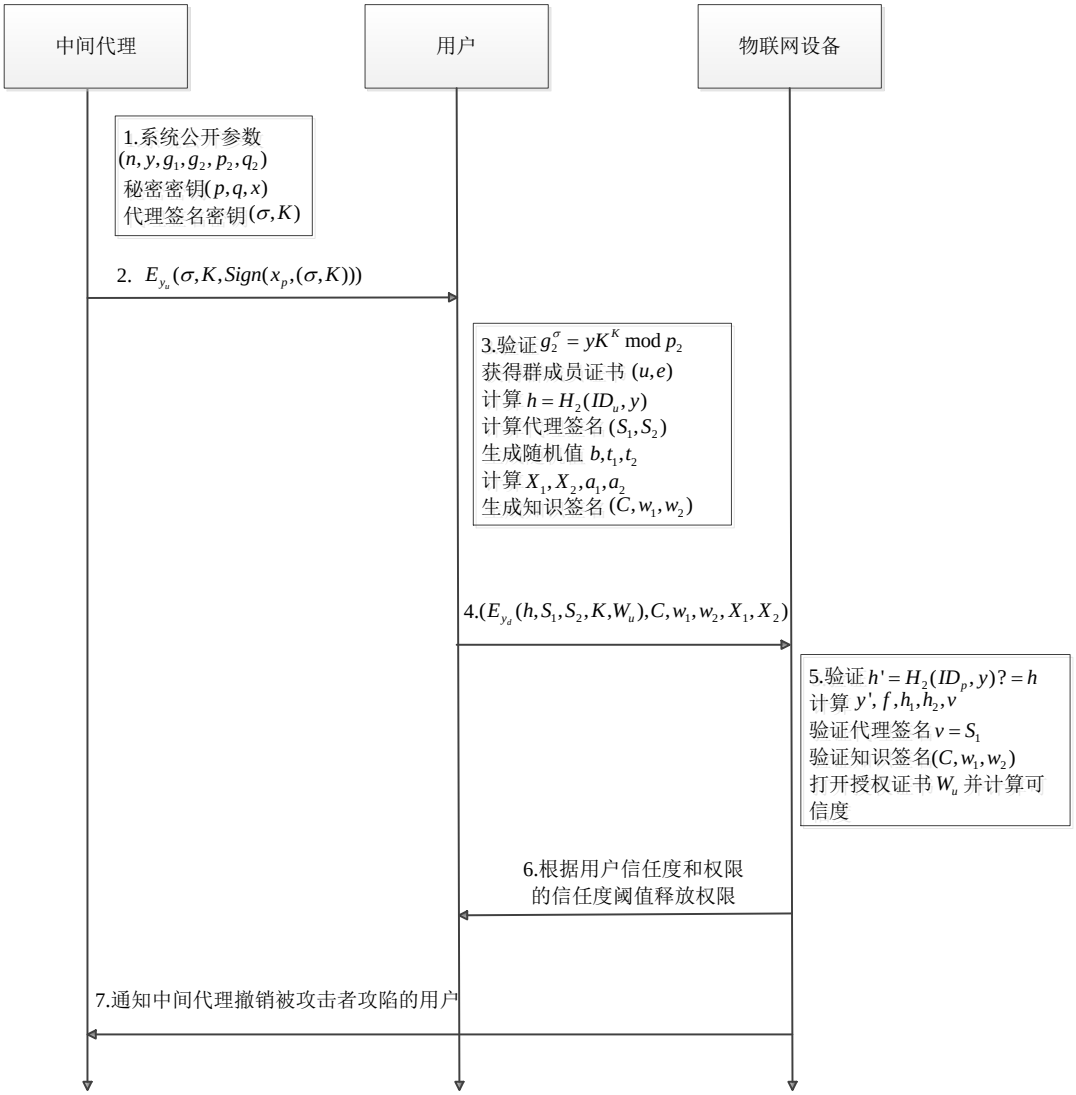


图 4-4 用户-设备匿名认证的过程图

4.4 协议安全性分析

本文将基于 Canetti-Krawczyk（CK）模型对匿名认证协议的安全性进行形式化的分析和证明，CK 模型定义了会话密钥（session key）安全并基于该定义说明了设计和分析安全协议的模块化方法。

4.4.1 基本定义

定义 4.1[22] 离散对数问题

对于给定的素数 p，生成元 $\alpha \in Z_p^*$ 和一个元 $\beta \in Z_p^*$，求整数 $x \in [0, p-2]$ 使得 $\beta \equiv \alpha^x \bmod p$ 成立是困难的。

定义 4.2[22] 判定的 Diffie – Hellman 问题

循环群 $G =< g >$ 的阶数为 q，其中的任意两个元素 g^a 和 g^b，$a,b \in \{0,...,q-1\}$，而元素 g^{ab} 就像是群 G 中的任意一个元素，也就是说无法区分三元组 $\{g^a, g^b, g^{ab}\}$ 中的元素，即从中随机抽取元素 a 和 b 的概率与从三元组 $\{g^a, g^b, g^c\}$ 中随机抽取元素 a，b 和 c 的概率是一样的。

定义 4.3[49] 假设 π 和 π' 都是 n 方的消息驱动协议，π 在 AM 模型中运行，π' 在 UM 模型中运行，若对任意的 UM 模型中的攻击者 U，都有 AM 模型中的攻击者 A 使得 $AUTH_{A,\pi}$ 和 $UNAUTH_{A,\pi'}$ 是计算不可分的，那么就认为 π' 在 UM 攻击模型中仿真 π。其中 AM 是理想认证链路模型，UM 是现实认证链路模型。

定义 4.4[49] 编译器 C 是指输入是协议描述，输出也是协议描述的算法，称编辑器是认证器时是指对任意协议 π，都有 $C(\pi)$ 在 UM 模型中仿真 π。

定义 4.5[49] 会话密钥安全是指对于 AM 模型中的任何攻击者，满足：（1）未被攻击的通信双方在完成会话密钥协商后，两者都能得到一样的会话密钥；（2）当攻击者进行会话查询攻击时，得到正确会话的概率小于 $1/2+\varepsilon$，其中 ε 是可以任何忽略不计的数值；则在 AM 模型中会话密钥是安全的。

定义 4.6[49] 假设 λ 是消息传输（MT）认证器，也就是指 λ 在 UM 模型中仿真了 MT 协议，MT 认证器主要是应用在单个消息上。若 C_λ 是在 λ 的基础上定义的编译器，那么 C_λ 就是一个认证器，且 C_λ 是多个 MT 消息传输认证器 λ 的组合。通过认证器可以让在 AM 中安全的协议转化为 UM 中安全的协议。

基于 CK 模型的认证和密钥协商协议的设计方法有四步：

- 设计一个在 AM 模型下安全的基本协议，该基本协议一般是通过常规的设计方得到；
- 构造一个可证安全的消息认证器；
- 利用该可证安全的消息认证器将 AM 模型下的基本协议的转化为在 UM 模型下安全的认证协议；
- 对转换后 UM 模型下的认证协议进行重新的排序、消息的重组等优化操作。

4.4.2 AM 模型中的认证协议

设计 AM 模型下用户与物联网设备间的基本的认证与会话密钥协商协议，本文设计的 AM 模型下的基本协议是基于 Diffie-Hellman 密钥交换协议的。在 AM 模型中攻击者只能被动的运行协议，不能进行重放攻击等，因此不用考虑使用时间戳或随机数，只要注意对必要的秘密信息加密防止被攻击者窃取。协议中的每条消息必须包含本次会话的标识，否则将不能用于协议转换，其主要原因是避免攻击者在 UM 模型中进行并行会话攻击。可以由用户和物联网设备分别生成会话标识的一部分，从而确保整个会话标识的

唯一性， AM 模式下的认证过程如图 4-5 所示：

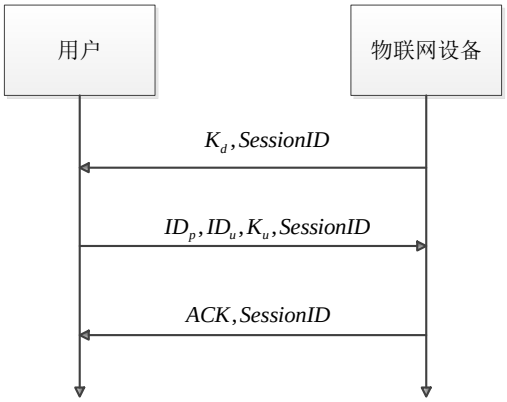


图 4-5 AM 模型中的认证协议

下面对其具体的运行步骤进行说明：

1) 物联网设备选择公共参数 g_v 和 n_v 用于协商与用户间的会话密钥，再随机选择秘密值 r_d， 计算 $K_d = g_v^{r_d} \bmod n_v$， 将 K_d 和会话标识 $SessionID$ 发送给用户；

2) 用户收到消息，随机选择秘密值 r_u，计算 $K_u = g_v^{r_u} \bmod n_v$ 和会话密钥 $K_{ud} = g_v^{r_u r_d} \bmod n_v$，将 ID_p, ID_u, K_u 和会话标识 $SessionID$ 发送给物联网设备；

3) 物联网设备验证会话标识的正确性，计算 $K_{ud} = g_v^{r_u r_d} \bmod n_v$，并向用户发送确认消息 ACK 和会话标识 $SessionID$。

在上述协议中，物联网设备与用户通过 Diffie-Hellman 密钥交换协议生成会话密钥 $K_{ud} = g_v^{r_u r_d} \bmod n_v$， 在 AM 模型中攻击者不能伪造、重放和修改物联网设备或用户的消息只能真实的转发用户和物联网设备生成的消息，且基于定义 4.2 中的 DDH 假设，Diffie-Hellman 协议已被 Canetti 和 Krawczyk [42]证明在 AM 模型下是会话密钥安全的，所以上述设计的基本协议在 AM 模型中也是会话密钥安全的。

4.4.3 UM 模型中的认证协议

已经证明基本协议在 AM 模型中是安全的，因此需要构造一个有效的认证器将 AM 模型中安全的基本协议转化为在 UM 模型中安全的协议。文献[52]中提出了一个基于数字签名的消息传输认证器并对其安全性进行了证明，如图 4-6 所示。

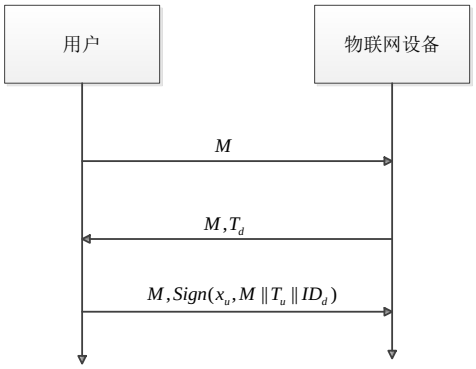


图 4-6 基于签名的消息认证器

基于数字签名的消息传输认证器包括三次交互，其中 M 是 AM 模型中的消息，T_d, T_u

分别是设备和用户生成的时间戳，$Sign(x_u,m)$ 是指用户对某个消息的签名，M 虽然出现在 UM 模型中的每条消息中，但是其也可以只在第一条或者最后一条消息中出现，在 M 中要有会话标识确保通信是在同一个会话中。

在 UM 模型中，需要对 AM 模型中的每条消息进行仿真，因此 UM 模型中的一个完整的认证器是由每条消息的消息认证器组成的，因为单条消息认证器是可证安全的，所以由定义 4.6，整个用户和设备匿名身份认证协议的认证器也是可证安全的。通过图 4-6 中可证安全的认证器可以将 AM 模型中安全的基本协议转化为在 UM 中可证安全的协议。一般消息认证器是将 AM 模型中协议的每条消息转换成 UM 中的多条消息，因此只要 AM 中消息的逻辑顺序正确，可以将 UM 中相同方向的消息合并，从而减少消息的个数。

下面介绍 UM 模型中的用户与设备匿名认证过程，如图 4-7 所示：

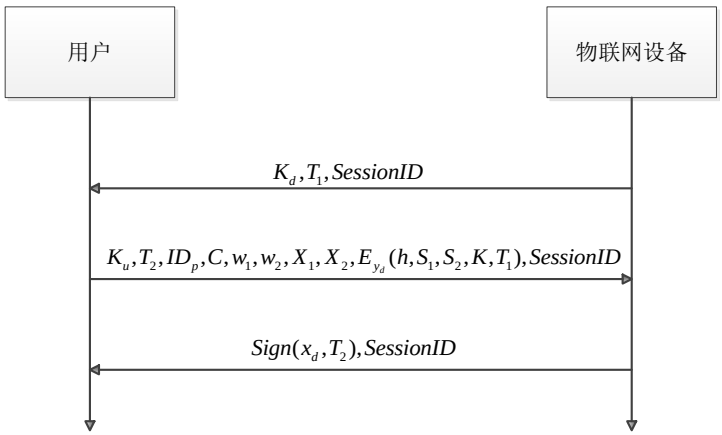


图 4-7 UM 模型中的认证协议

1) 物联网设备选择公共参数 g_v 和 n_v，随机选择秘密值 r_d，计算公钥 $K_d = g_v^{r_d} \bmod n_v$，生成时间戳 T_1，将 K_d, T_1 和会话标识 $SessionID$ 发送给用户；

2) 用户收到物联网设备发来的消息，验证时间戳 T_1 的新鲜性，计算其公钥 $K_u = g_v^{r_u} \bmod n_v$，会话密钥 $K_{ud} = (K_d)^{r_u} = g^{r_d r_u} \bmod n_v$，计算 $h = H_2(ID_p, y)$，用代理签名密钥 σ 对 h 签名，选择随机数 $k < q_2$，计算代理签名 (S_1, S_2)，其中：$S_1 = (g_2^k \bmod p_2) \bmod q_2$，$S_2 = k^{-1}(h + \sigma S_2) \bmod q_2$，最后对消息 h, S_1, S_2, K, T_1 加密，生成密文 $E_{y_d}(h, S_1, S_2, K, T_1)$；

3) 用户计算 $M = (K_u \| T_1)$，生成时间戳 T_2，$b \in_R \{V - 2^{l_b}, V + 2^{l_b}\}$，$t_1 \in_R \pm\{0,1\}^{\alpha(l_s + l_c)}$，$t_2 \in_R \pm\{0,1\}^{\alpha(l_b + l_c)}$，计算 $X_1 = u^b \bmod n = g_1^{e^{-1}b} \bmod n$，$X_2 = g_1^b \bmod n$，$a_1 = X_1^{t_1} \bmod n$，$a_2 = g_1^{t_2} \bmod n$，计算知识签名 (C, w_1, w_2)，其中：
$C = H_2(g_1 \| X_1 \| X_2 \| a_1 \| a_2 \| h \| S_1 \| S_2 \| K \| M)$，$w_1 = t_1 - C(e - U)$，$w_2 = t_2 - C(e - V)$，将消息 $ID_p, C, w_1, w_2, X_1, X_2, E_{y_d}(h, S_1, S_2, K, T_1), K_u, T_2, SessionID$ 发送给物联网设备；

4) 在 UM 模型中攻击者可以对消息流进行修改、重放或伪造，物联网设备首先验证时间戳 T_2 的新鲜性，用私钥 x_d 对密文解密，验证对中间代理身份信息 h 的代理签名

(S_1,S_2)的正确性，即判断用户提供的中间代理的身份信息是否正确，验证时间戳T_1是否正确，计算$C'=H_2(g_1\|X_1\|X_2\|X_1^{w_1-CU}X_2^{C}\|g_1^{w_2-CV}X_2^{C}\|h\|S_1\|S_2\|K\|M)$并验证$C=C'$，$w_1\in\pm\{0,1\}^{\alpha(l_s+l_c)+1}$，$w_2\in\pm\{0,1\}^{\alpha(l_b+l_c)+1}$三者是否同时成立，从而确定该用户是否是合法中间代理群中的合法成员；

5) 物联网设备用私钥x_d对时间戳T_2签名$Sign\ (x_d,T_2)$，并将签名和会话标识$SessionID$发送给用户，然后用户与物联网设备间完成相互的身份认证和会话密钥协商。

4.4.4 安全性分析

1) 抗伪装攻击

本方案在UM模型中可以抵抗伪装攻击，假设攻击者攻陷了某个用户窃取了他的公私钥(r_u,K_u)，然后伪装成合法用户与物联网设备协商会话密钥K_{ud}，但是攻击者不知道中间代理对该用户生成的代理签名密钥(σ,K)，从而他无法伪造出有效的代理签名(S_1,S_2)，并且攻击者也不知道用户的群成员私钥e，他也不能伪造出合法的知识签名(C,w_1,w_2)。当设备对攻击者身份认证时，通过验证代理签名和知识签名，设备可以发现该伪装者不是由他提供身份信息的中间代理授权的用户。

2) 抗重放攻击

本方案能够抵抗重放攻击，假设攻击者截获了某次会话消息，但是用户每次与设备通信时都会生成新的公钥K_u与物联网设备协商出新的会话密钥K_{ud}，并且用户会对K_u进行签名保护。另外，攻击者无法获取用户群证书中的私钥，其无法仿造用户生成知识签名，所以其伪造的C值是无法通过设备的验证的。本方案还在协议引入了时间戳T_1和T_2用于阻止重放攻击。

3) 匿名性

用户在对物联网设备进行身份证明时，设备和其他用户或攻击者都无法知道该用户的真实身份。用户使用知识签名(C,w_1,w_2)作为身份真实性验证的凭证，凭证中不包含任何有关用户真实身份的信息，并且知识签名是经过随机值盲化处理的，不同的随机盲化值生成的知识签名是不同的，确保了用户的匿名身份认证，且用户利用随机值生成的知识签名是具有时变性的，因此设备或攻击者不能将同一个用户生成的两个知识签名联系在一起。

4.5 协议性能分析

本文的物联网设备共享系统中主要运用到的设备是智能手机和物联网设备等，这些设备的计算能力较弱和电池能量有限，无法负荷交互复杂的认证协议，因此计算量是本文在设计协议时需要着重考虑的问题。将本文方案与 Liu 和 Yang 的匿名认证方案的计算量进行比较，如表 4.1 所示，其中计算量主要由乘法运算量 T1、模指数计算量 T2 和哈希值计算量 T3 组成。

表 4.1 计算量比较

方案	证明方	验证者
Liu	33T1+23T2+3T3	5T1+24T2+4T3
Yang	24T1+22T2+2T3	15T1+19T2+2T3
本文方案	7T1+11T2+2T3	6T1+6T2+2T3

从表中可以看出本方案不论是证明者还是验证者的计算量都明显减小，尤其是耗时较大的乘法运算和模指数运算方面计算量大大减小，本方案的计算量大概是 Liu 和 Yang 方案计算量的 1/2 到 1/3 左右。另一方面，从交互消息的大小来看，本文方案中没有复杂的公钥证书交互，只有一些简单的知识签名及其验证工作，系统需要的安全参数也较少，消息长度明显缩短，因此本方案适用于使用轻量型设备的物联网设备共享系统。

4.6 本章小结

本文针对用户访问物联网设备进行身份认证时的隐私保护问题，提出了一种基于代理签名和知识签名的匿名身份认证方案。该方案通过代理签名技术实现了物联网设备对用户提供的中间代理的身份信息的验证，确认中间代理的合法性，然后通过用户提交的知识签名在无需用户真实身份信息的前提下验证用户是否是由合法的中间代理授权的合法用户。本文采用 CK 模型对该方案的认证安全性进行了形式化分析和证明，分析表明该方案可以抵抗重放攻击和伪装攻击；最后对协议的性能进行分析，分析表明该认证方案计算量小，适用于物联网中的轻量型设备。

第五章 物联网设备共享系统的设计与实现

随着移动通信技术的快速发展，人们可以随时随地的通过智能终端访问网络，与连接在网络上的智能设备进行通信，这使得物联网的网络架构由以往孤立封闭、垂直体系的网络环境逐渐演变成一个开放的、面向用户的泛在网络。在这个网络中，不仅包含各种智能设备，还存在大量的用户。用户与智能设备相互交互合作，设备根据用户的需求提供相应的服务，用户或设备可以通过网络搜寻所需的设备并与之交换数据。在物联网智慧城市中，用户可以同时是城市资源或服务的使用者和提供者，用户可以将自己闲置的设备分享给物联网中的其他用户使用，从而实现社会资源的有效利用。在上述方案的基础上，本文设计了一个应用于房屋共享租赁的可委托授权和可保护隐私的原型系统，该系统中房屋所有者将租房工作委托给中间代理，由中间代理向房屋租赁者提供租房服务，另外本系统中房屋租赁者在与房屋的智能锁相互身份认证时，无需提供他的真实身份信息，从而实现了对用户的安全认证和隐私保护。

5.1 系统设计

本节主要介绍可委托授权的房屋租赁共享原型系统的整体架构和系统中每个功能的具体实现流程。

5.1.1 整体架构

本系统主要由房屋所有者、中间代理、房屋租赁者（用户）和房屋智能锁构成，具体架构如图 5-1 所示。

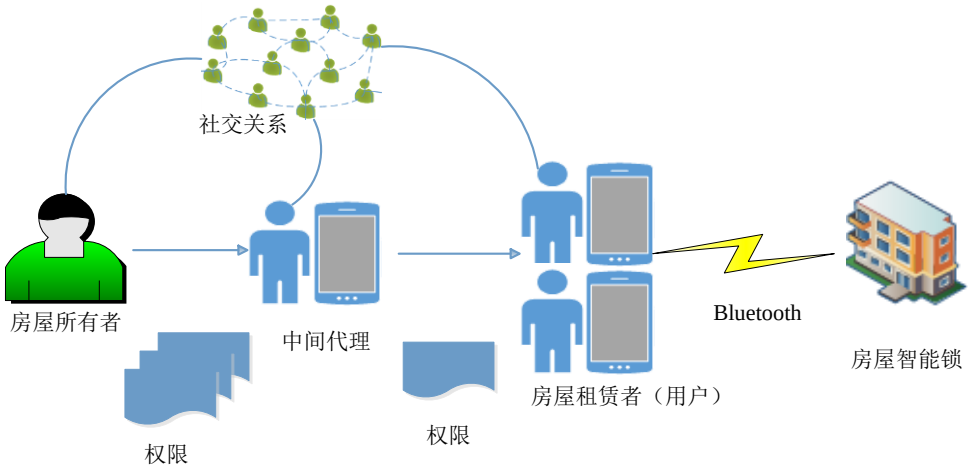


图 5-1 设备共享系统架构

房屋所有者是房屋的主人，他可以向他的直接用户授予使用房屋的权限，也可以将授权用户使用房屋的能力委托给中间代理，由中间代理代替他发放权限，从而扩大可以使用房屋的用户范围。

中间代理在本系统中是房屋所有者的一个直接关联用户，他可以代替房屋所有者向其社交范围内的用户授权使用房屋，他也可以作为普通用户向所有者申请使用房屋。

房屋租赁者（用户）是中间代理社交关系网络中的一名用户，他向中间代理申请房屋的使用权限，并与房屋智能锁交互认证获得最终使用权。

房屋智能锁在设备所有者的控制下向用户提供服务，其与租赁者通过蓝牙通信。

整个系统具体的交互流程如图 5-2 所示。

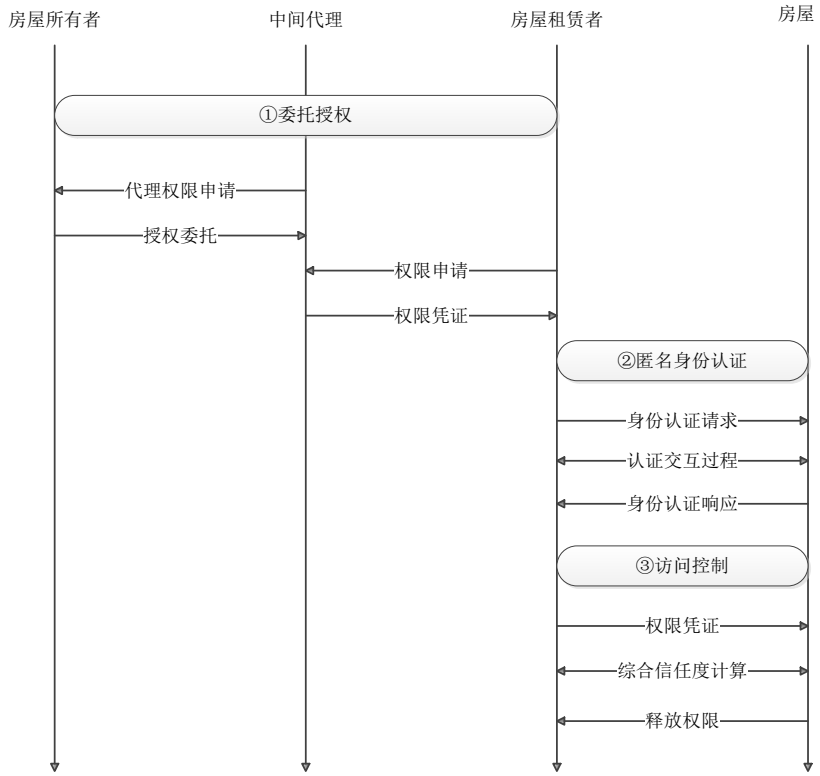


图 5-2 系统交互流程图

从图 5-2 可以看出系统实现的功能包括三部分：委托授权、匿名身份认证和访问控制。具体的房屋所有者、中间代理、租赁者和房屋智能锁之间的交互过程所述如下：

想要获得代理权的用户向房屋所有者发出代理权限申请，房屋所有者验证中间代理的身份，根据申请信息计算对该中间代理的信任度，分配其相应的授权角色（不同的角色对应不同的授权范围），并基于 X.509 公钥证书生成委托授权凭证发送给中间代理，从而完成权限的委托。中间代理对收到的委托授权凭证进行解析，得到他被分配的角色和授权范围。租赁者向中间代理发出权限申请，中间代理首先判断租赁者申请的权限是否在自己授权范围内，然后根据申请信息计算对该租赁者的信任度，最后基于 X.509 公钥证书生成权限凭证发送给租赁者。租赁者得到权限凭证后，首先向房屋智能锁发出身份认证请求，然后两者之间通过握手交互协议完成匿名身份认证以及安全的会话密钥协商。完成身份认证和密钥协商后，租赁者将权限凭证通过安全信道发送给智能锁，智能锁对权限凭证进行解析，获得租赁者被分配的角色并计算所有者对该租赁者的综合信任度，然后智能锁检索其存储的访问控制表中该租赁者角色对应的权限和信任度阈值，判断租赁者的信任度是否大于角色对应权限的信任度阈值，从而进行权限的释放。

5.1.2 委托授权

委托授权的参与者包括房屋所有者、中间代理、用户和房屋智能锁。所有者为了向

更多的用户分享他闲置的房屋，可以将授权使用房屋的能力委托给中间代理，中间代理可以在他的社交范围内代替所有者分享房屋的使用权限从而扩大用户的受众范围，委托授权的具体流程如图 5-3 所示。

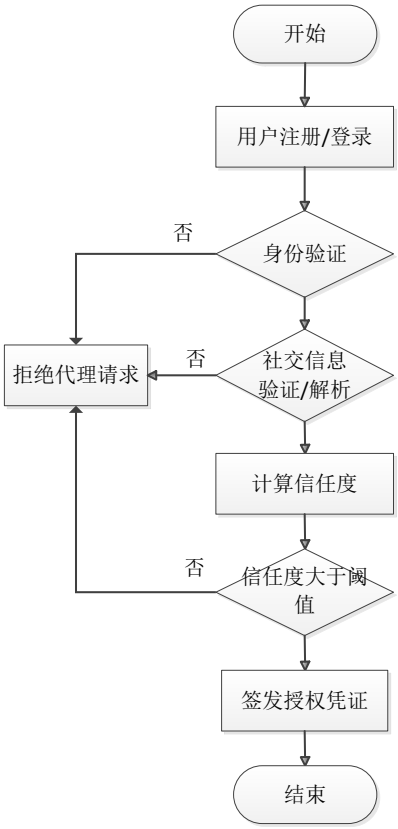


图 5-3 委托授权流程图

中间代理首先在系统中注册自己的个人信息包括用户名、密码、真实姓名和证件号等，房屋所有者将该中间代理的注册信息登记在他的用户注册列表中；对于之前已经注册的中间代理直接输入用户名和密码登录系统，房屋所有者则检索用户注册列表并验证中间代理的身份信息是否正确。然后中间代理基于他的移动智能手机生成他的社交文件，其中包括联系人名称、社交联系地址和常用的手机 APP 名称，然后将社交文件发送给房屋所有者，房屋所有者解析中间代理的社交文件并与自己的社交文件相比较，利用余弦相似性计算出对该中间代理的信任度。由于房屋所有者的不同的权限具有不同的敏感度，因此房屋所有者将根据中间代理信任度的大小授权其不同范围的授权能力，如图 5-4 所示是角色-权限配置的流程图。

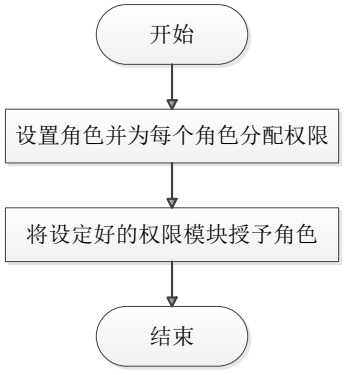


图 5-4 角色-权限配置图

本文依据表 3.2 设计了中间代理权限分配表，如表 5.1 所示。

表 5.1 中间代理权限分配

中间代理角色	信任度阈值	权限
R1	0.96	P1
R2	0.9	P2
R3	0.8	P3

房屋所有者将中间代理的信任度值与表 5.1 中的信任度阈值相比较，确定授予中间代理的角色及其获得授权权限范围，角色 R1 获得与房屋所有者相同的授权范围，用 P1 表示，角色 R2 获得角色 Close 和角色 General 全部权限的代理权，用 P2 表示，角色 R3 只获得角色 General 权限的代理权，用 P3 表示。最后房屋所有者签发对该中间代理的授权凭证，其中包括委托授权的内容、委托时间等信息，并通知房屋智能锁当前获得代理权的中间代理的身份、委托时间等。租赁者向中间代理发出权限申请，中间代理判断租赁者申请的权限是否在自己获得的角色的授权范围内，若在授权范围内则计算该租赁者的信任度，生成对该租赁者的权限凭证。房屋所有者存储了一张中间代理表，其中记录了中间代理的个人信息、委托的权限、委托时间、证书发布时间和撤销时间等信息。

当然，中间代理本身作为房屋所有者社交范围内的一名用户，不仅可以申请房屋的代理权，也可以作为普通租赁者直接申请房屋的使用权限。房屋所有者根据中间代理申请的权限以及对该中间代理的社交信任度生成权限凭证，同时房屋所有者在其注册用户表中更新租赁者的权限信息、证书发布与撤销的时间等。

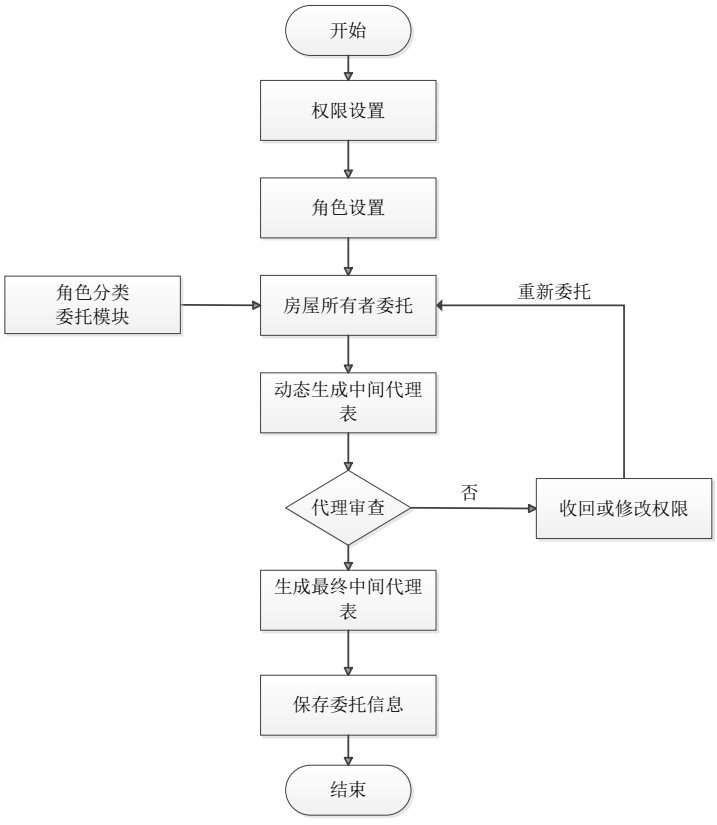


图 5-5 代理权限审查流程

对于初次申请代理权的用户，房屋所有者只能依据其社交信任度进行权限的委托。

当用户再次申请代理权时，所有者不仅依据用户的社交信任度，同时根据房屋智能锁的反馈信息动态的调整对中间代理委托的权限，因此需要一个权限审查机制对中间代理的权限进行约束。中间代理每次登录系统申请代理权时，所有者都需要重新对其进行角色和权限的分配，这是因为所有者和中间代理间的信任度是随着时间、社交关系的变化而动态变化的。中间代理每次完成代理后，所有者都会对房屋的状态和中间代理的行为表现作出评价，分为好和差两个等级，智能锁会在其中间代理表中记录所有者对该用户的评价信息。所有者生成对中间代理的权限凭证后，将其发送给智能锁进行代理审查，智能锁检索其中间代理表中该中间代理的历史行为评价信息，并将评价信息反馈给所有者，若评价为好则所有者可以维持或提升该中间代理的授权范围，若为差则撤销或者降低该中间代理的权限，最后所有者修改并存储中间代理表中的对该中间代理的委托信息。图 5-5 是代理权限审查流程图。

5.1.3 匿名身份认证

基于第四章的匿名认证方案设计了租赁者与房屋智能锁间的匿名身份交互过程。匿名身份交互的核心是通过租赁者与智能锁间的握手协议完成身份的认证与会话密钥的协商。租赁者向中间代理申请了匿名证书后，打开手机蓝牙扫描智能锁，智能锁同时进行初始化连接并监听租赁者的连接请求，当双方建立通信链路后，租赁者向智能锁发送握手请求，然后租赁者与智能锁交换身份证书完成匿名身份认证，协商彼此间安全的会话密钥，最后建立安全的通信通道。图 5-6 所示是匿名认证握手交互的流程图。

图 5-6 匿名认证握手交互过程

在租赁者与智能锁匿名交互的过程中，需要传递各种类型的消息。因此将消息的数据结构定义如下：

```
struct{
    MsgType message_type;       //消息的类型
    MsgLen message_length;      //消息的长度
    MsgBody message_body[];     //存放消息的主要内容
   }
```

下面具体介绍握手交互过程中涉及到的消息类型及其数据结构。

1) UserHello 消息

租赁者握手请求 UserHello 消息中包含租赁者端的随机数发生器生成的随机数 user_random 和扩展字段 user_require 说明默认的加解密算法 RSA。

```
struct
{
    MsgBody user_random[];
    MsgBody user_require;     //默认加解密算法为 RSA
   } UserHello;
```

2) DeviceHello 消息

智能锁的握手响应 DeviceHello 消息中包含智能锁端的随机数发生器生成的随机数 device_random 和扩展字段 device_require 说明默认的加解密算法 RSA。

```
struct
{
    MsgBody device_random[];
    MsgBody device_require;
   } DeviceHello;
```

3) DeviceCredential 消息

智能锁基于 X.509 的公钥证书，其中包含了密钥发生器生成的智能锁的 RSA 公钥。

```
struct
{
    MsgBody device_credential;
    }DeviceCredential;
```

4) UserAnonyCredential 消息

租赁者匿名身份证书 user_anony_credential，其不是普通的 X.509 证书，里面包含了基于 DSA 数字签名生成的代理签名、代理签名密钥和对租赁者群成员证书的知识签名。

```
struct
{
    MsgBody user_anony_credential;
    } UserAnonyCredential;
```

5) DeviceKeyNegotiate 消息

本方案中是基于 Diffie－Hellman 协议进行密钥的协商，智能锁生成密钥协商参数 deviceKNParam，其中包括 DH 公开参数和智能锁的密钥协商的公钥，同时消息中还有智能锁对 DH 公开参数、租赁者生成的随机数 user_random 和智能锁生成的随机数 device_random 的签名。

```
struct
```

```
{
    MsgBody deviceKNParam[];
    signature struct
    {
        MsgBody user_random[];
        MsgBody device_random[];
        MsgBody deviceKNParam[];
        } device_signature;
    } DeviceKeyNegotiate;
```

6) UserKeyNegotiate 消息

租赁者根据智能锁生成的 DH 公开参数生成的他的密钥协商公钥。

```
struct
{
    MsgBody userKNpubkey[];
    } UserKeyNegotiate;
```

7) DeviceSKCompleted 消息

智能锁通过握手完成消息通知租赁者已对其身份进行验证并且会话密钥协商成功。

```
struct
{
    MsgBody device_SKCompleted[];
    }DeviceSKCompleted;
```

其中，SKCompleted = hash(SK_secret，ID_d||hash(msg_handshake)),SK_secret 是协商的会话密钥，ID_d 是智能锁的身份标识，msg_handshake = (UserHello || DeviceHello || hash(DeviceCredential) || hash(UserAnonyCredential) || DeviceKeyNegotiate || UserKeyNegotiate)，Hash 算法选用 SHA-1。

8) UserSKCompleted 消息

租赁者生成握手完成消息通知智能锁会话密钥协商成功

```
struct
{
    MsgBody user_SKCompleted[];
    }UserSKCompleted;
```

其中，SKCompleted = hash(SK_secret，||hash(msg_handshake)),SK_secret 是协商的会话密钥，msg_handshake = (UserHello || DeviceHello || hash(DeviceCredential) || hash(UserAnonyCredential) || DeviceKeyNegotiate || UserKeyNegotiate)，Hash 算法选用 SHA-1。

9) ErrorWarning 消息

在租赁者和智能锁交互的过程中可能会发生错误或者双方身份认证不成功，需要发

送错误消息警告信息提示租赁者或智能锁发生错误。

```
struct
{
    MsgType msg_warning;
    MsgLen warning_length;   //警告消息的长度
    MsgBody warning_value;   //警告消息码
    } ErrorWarning;
```

其中，warning_value = 0x01，租赁者端反馈，智能锁端证书验证失败；0x02，智能锁端反馈，租赁者端的匿名身份证书验证失败；0x03，智能锁端反馈，租赁者的代理密钥验证失败；0x04，智能锁端反馈，中间代理的身份验证失败；0x05，租赁者端或智能锁端反馈，会话密钥协商失败。

5.1.4 访问控制

智能锁在完成了对租赁者的匿名身份认证后，需要根据租赁者的权限凭证释放相应的权限。图 5-7 是本系统的访问控制流程图。

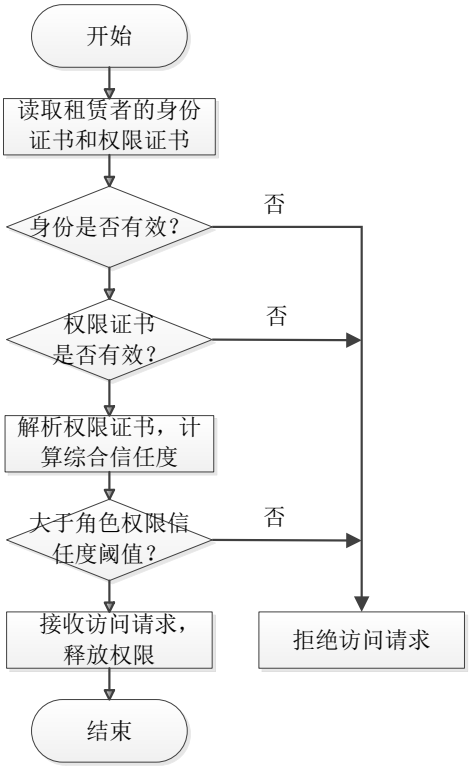


图 5-7 系统访问控制流程图

访问控制的参与者是房屋所有者、中间代理、租赁者和房屋智能锁。租赁者向中间代理发出授权请求 Apply（ID_{user}，ID_{proxy}，<Permission>，<SocialFile>），其中 ID_{user} 和 ID_{proxy} 是租赁者和中间代理的身份标识，Permission 是租赁者申请的权限，SocialFile 是租赁者的社交文件。中间代理首先对租赁者的身份进行验证，若租赁者的身份有效，再判断租赁者申请的权限是否在自己角色的授权范围内，若中间代理拥有授权该权限的能力，则依据租赁者的社交文件 SocialFile 计算对该租赁者的信任度并基于授权的角色和

信任度生成对该租赁者的权限凭证，中间代理将租赁者到房屋所有者之间各级代理域中上级用户对下级用户的权限凭证作为凭证链一起发送给租赁者。租赁者首先要完成与设备间的匿名身份认证，认证成功后，通过建立的安全通道将权限证书发送给智能锁，智能锁打开权限证书，获得租赁者分配的角色并将各级信任度相乘获得该租赁者的综合信任度，然后智能锁解析他的访问控制列表，判断租赁者的信任度值是否大于其角色对应权限中最低的信任度阈值，也即是否能激活该角色，如果激活角色就释放对应的权限，否则就拒绝访问。

5.2 移动终端设计

5.2.1 整体功能

本系统中房屋所有者、中间代理和租赁者之间以及他们与智能锁间都是利用移动终端通信交互的。移动智能终端的主要功能包括用户身份管理、信任度管理、授权证书的生成和验证。移动终端的设计分为三层：通信层、安全层和应用层，如图 5-8 所示。本节主要具体介绍安全层中的核心功能模块：信任度管理模块和证书管理模块。

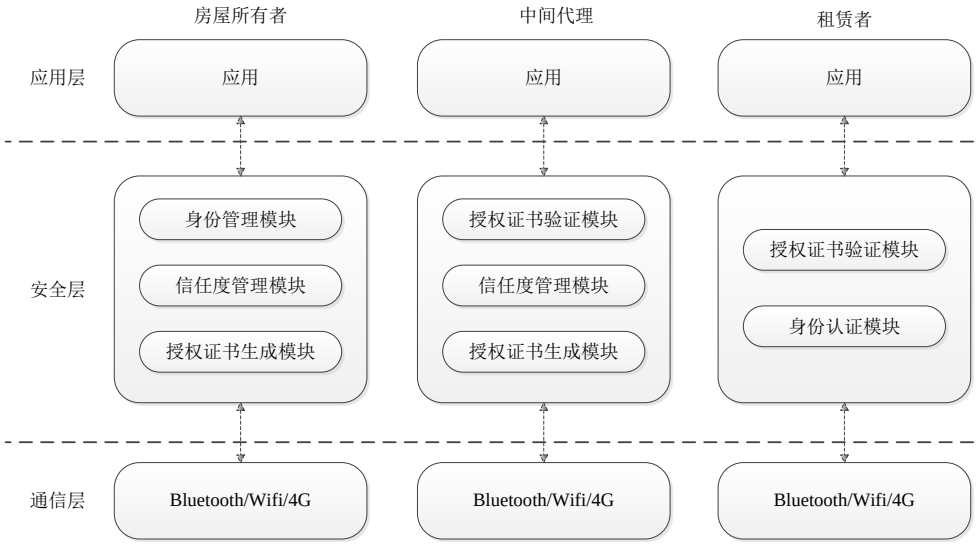


图 5-8 移动终端的功能模块

5.2.2 信任度管理模块

信任度管理模块的主要功能是计算两个用户间基于社交关系的信任度值。本文将根据两个用户的联系人信息进行社交相似度计算。作为智能手机的主要功能，每个手机用户都会使用手机的通讯录功能，因此为了获得用户的联系人信息，本文对用户的智能手机中的通讯录进行解析，提取用户联系人姓名和手机号。Android 手机中的通讯录数据存储在/data/data/com.android.provider.contacts 路径下的文件夹 databases 中的数据库文件 contacts2.db 中。在数据库文件 contacts2.db[53]内有许多张表，其中联系人信息存储在 data 表和 raw_contacts 表中，两表的主要字段如表 5.2 和 5.3 所示。表 5.2 中 data1-data15 字段存储的联系人信息包括联系人名称、联系人电话号码、电子邮件等。

表 5.2 data 表结构

字段名称	数据类型	说明
minitype_id	integer	存储数据类型
raw_contact_id	integer	联系人信息索引号
data1-data15	text	存储联系人的所有信息

表 5.3 raw_contacts 表结构

字段名称	数据类型	说明
_id	integer	索引号
contact_id	integer	联系人索引号
deleted	integer	删除标识
display_name	text	联系人名称
last_time_contacts	integer	最后联系时间
version	integer	版本号，监听变化

用户获得手机通讯录中联系人信息的关键代码如下所示：

```
//Android 系统中操作联系人的权限设置
<uses-permission android:name=”android.permission.READ_CONTACTS”/>
<uses-permission android:name=”android.permission.WRITE_CONTACTS”/>
//用户联系人信息定义
private static final String[] CELLPHONES_FILE = new String[]{
  Cellphone.DISPLAY_NAME, Cellphone.NUMBER
  };
//获取手机通讯录中联系人的信息
private void getCellphoneContacts(){
   ContentParser parser = uContext.getContentParser();
//获取手机联系人,private String URI_DATA=”content://com.android.contacts/data”
   Cursor cellphoneCursor = parser.query(Cellphone.URI_DATA , CELLPHONES_ FILE,null,null,null);
  //获取 data 表中的联系人的相关信息
  if (cellphoneCursor != null){
     while(cellphoneCursor.turnToNext())
     {
        //获得联系人手机号码
       String cellphoneNumber = cellphoneCursor.getString(PHONES_NUMBER_INDEX);
      //若手机号为空或是空字段，则跳过当前循环
```

```
        if (TextUtils.isEmpty(cellphoneNumber))
            continue;
          //获取联系人姓名
            String contactName = cellphoneCursor.getString(PHONE_DISPLAY_NAME_INDEX);
            uContactsName.add(contactName);  //向集合 uContactsName 中添加联系人姓名
             int size1 = uContactsName.size( ); //计算集合 uContactsName 的大小
             uContactsNumber.add(cellphoneNumber); //向集合 uContactsName 添加联系人手机号
             int size2 = uContactsNumber.size( ); //计算集合 uContactsNumber 的大小
             }
          cellphoneCursor.close ( );
          }
       }
```

用户提取了他的联系人姓名或手机号信息后生成社交文件，本系统中使用联系人手机号作为社交文件，因为不同用户对相同的联系人可能有不同的称呼，但是保存的手机号是一样的，因此用手机号可以唯一标识一个用户。用户根据另一个用户的社交文件基于余弦相似度计算信任度值，社交文件中实质上存储的是一系列的手机号字符串，计算的是两个字符串之间的余弦相似性，其中的关键代码如下所示：

```
    //用户的联系人手机号余弦相似性算法
    public class SocialSimilarity(String user1,String user2){
        //创建向量空间模型
        Map<String,int[]>userSpace = new HashMap<String,int[]>();
        int[] tempCountArray = null;
        for(String string1:user1.toCharArray()){
            if(userSpace.containsKey(String1)){
                userSpace.get(string1)[0]++;
                }else{
                  tempCountArray = new int[2];
                  tempCountArray[0]=1;
                  tempCountArray[1]=0;
                  userSpace.put(string1, tempCountArray);
                  }
            }
        for(String string2:user2.toCharArray(){
            if(userSpace.containsKey(string2)){
```

```
                userSpace.get(string2)[1]++;
               }else{
                  tempCountArray = new int[2];
                  tempCountArray[0]=1;
                  tempCountArray[1]=0;
                  userSpace.put(string2,tempCountArray);
                  }
            }
            //计算余弦相似度
            double user1Mod = 0.00; //用户 1 的向量模
            double user2Mod = 0.00; //用户 2 的向量模
            double userDotProduct = 0.00;//向量的点乘
            Interator iterator = userSpace.entrySet().interator();
            while(iterator.hashNext()){
                    Map.Entry entry = (Map.Entry)iterator.next( );
                    tempCountArray = (int[ ])entry.getValue( );
                    user1Mod += tempCountArray[0]* tempCountArray[0];
                    user2Mod += tempCountArray[1]* tempCountArray[1];
                    userDotProduct += tempCountArray[0]* tempCountArray[1];
                    }
            user1Mod = Math.sqrt(user1Mod);
            user2Mod = Math.sqrt(user2Mod);
            return (userDotProduct/(user1Mod* user2Mod)); //返回信任度初值
        }
```

5.2.3 授权证书管理模块

授权证书管理模块包括授权证书的生成与验证，其中授权证书的生成包括房屋所有者生成的委托授权证书和由中间代理生成的对租赁者的授权证书。本节将以房屋所有者的委托授权证书为例对具体的证书设计过程进行介绍，授权证书的设计是基于 X.509 公钥证书的。

X.509 是基于公钥密码体制和数字签名服务的，与用户相关的公钥证书是 X.509 的核心，这些证书是由可信的认证中心（CA）生成的，并且被存放在目录服务器中，目录服务器不负责创建公钥证书，只是为用户获取证书提供一个方便的存储方式。

X.509 证书主要包含以下内容：

- 版本号（Version）：用户区分不同版本的证书，默认设置为 1，版本号为 2 存在发行商唯一标志或者主体唯一标志，版本号为 3 则存在一个或者多个扩展域。
- 序列号（Serial number）：用证书表示是签证机构 CA 中的唯一证书。

- 签名算法标志（Signature algorithm identifier）：表示用于给证书签名的带参数的算法，由于在证书尾部的签名区域中还会参与，这里一般没有信息。
- 发行者名称（Issuer name）：X.509 中生成并签发证书的签证机构的名称。
- 有效期（Period of validity）：包含证书的生效日期和结束日期。
- 证书主体名（Subject name）：获得证书的用户名称以及证书证明主体的公钥。
- 证书主体公钥信息（Subject’s public-key information）：主体的公钥和将被使用的密钥算法标志。
- 发行商唯一标志（Issuer unique identifier）：一个可选串用于唯一标识签证机构。
- 证书主体唯一标志（Subject unique identifier）：一个可选串用于唯一标识证书主体。
- 扩展（Extension）：一个或者多个扩展项，主要有三类：密钥和策略信息，证书主体和发行者属性以及证书路径约束。
- 签名（Signature）：用签证机构的私钥对证书的所有域和对这些域的哈希值加密。

X.509 证书使用 ASN.1（Abstract Syntax Notation One，抽象语法标记）进行编码，它是一种 ISO/ITU-T 标准，可以对数据的表示、编解码和传输的数据结构进行描述，并且 ASN.1 的描述可以被转换成 Java 或 C/C++的数据结构。本文采用 ASN.1 编码设备所有者的委托授权证书，并通过一个个的数据块来描述整个数据结构，具体的编码如下所示：

```
ProxyDelegateCertificate :: = SEQUENCE{
    mainMsg MainMsg          //由房屋所有者签发的基本信息
    roleMsg   Roles            //房屋所有者委托给中间代理的角色类型
    trustMsg   Trust           //房屋所有者计算的中间代理的信任度值
    proxyKeyMsg ProxyKey     //房屋所有者生成的代理密钥信息
   }
  MainMsg :: = SEQUENCE{
        Version                    //证书的版本号
        SerialNumber               //证书的序列号
        OwnerMsg                   //房屋所有者的信息
        ProxyUserMsg               //中间代理的信息
        SignatureAlogorithm        //签名算法
        ValidityPeriod             //授权证书的有效时间
        SubjectName                //授权证书拥有者的主体名，实际就是中
                                   间代理注册的用户名
        Depth                      //委托授权的深度
        Signature                  //房屋所有者对证书的签名
        }
  OwnerMsg :: =SEQUENCE{
```

```
    OwnerPubID                  //房屋所有者 X.509 证书的序列号
    }
  ProxyUserMsg :: = SEQUENCE{
    ProxyUserPubID              //中间代理 X.509 公钥证书的序列号
   }
  Roles :: = SEQUENCE{
    roleMsg :: =SEQUENCE{
      OwnerPubID        //房屋所有者的公钥证书号
      roleType :: = Set{   //房屋所有者分配给中间代理的角色
        Ra       //角色 a
        Rb       //角色 b
        ......
       }
      OwnerSignature                //房屋所有者的签名值
      }
    }
  Trust :: =SEQUENCE{
      EncryptionTrustValue   //所有者用智能锁的公钥对他对中间代理的信任
度加密
      OwnerTrustSignature    //所有者对信任度值的签名
     }
  ProxyKey :: =SEQUENCE{
      EncryptionProxyKey  //所有者用中间代理的公钥加密生成的代理签名密
钥
      OwnerProxyKeySignature   //所有者对代理签名密钥的签名
     }
  Extensions
  OwnerDelegateLimit EXTENSIONS :: ={
      //定义了房屋所有者限制中间代理委托的角色或者其对应的某个权限。
   }
```

该委托授权证书是由房屋所有者签发的，其中包括证书的基本授权信息、签名的算法以及房屋所有者的签名。授权信息中包含房屋所有者委托给中间代理可授权的角色以及相应的权限、代理权的有效时间、所有者对中间代理的信任度值、代理签名密钥和房屋所有者对中间代理的委托限制等，其中中间代理无法得到加密的信任度而只能单纯的转发给租赁者，再由租赁者转发给智能锁，最后由智能锁解密并综合各级信任度计算租赁者的综合信任度。所有者利用 Depth 字段表示中间代理可以将权限继续传递下去的深度，每传递一次，Depth 就会自动减一，当 Depth=0 时表示中间代理不能继续传递权限。

同时，在权限凭证中包括了房屋所有者和中间代理的信息，这样可以使系统明确权限委托双方的情况。

中间代理收到房屋所有者的委托授权证书后需对其有效性进行验证，必须满足以下条件：

（1）委托授权凭证的签名是由正确的所有者签发的；

（2）验证授权凭证的有效期，确保中间代理获得证书的时间在证书有效时间和终止时间之间；

（3）授权凭证 Depth 字段不为 0，说明所有者允许中间代理传递权限；

（4）中间代理验证被委托的角色集合 Roles(Ra, Rb, Rc,...) 是否与自身的角色相互排斥。

中间代理发送给租赁者的授权证书以及对证书的验证与上述过程类似，这里就不在详述。

证书验证类具体定义如下：

```
//定义证书验证类 CertificateVerify
import java.io.*;
import java.security.cert.*;
import java.text.SimpleDateFormat;
import java.util*;
public class CertificateVerify{
    private static String tag = "CertificateVerify";
    public static Certificate readCert(File file){
        Certificate cert = null;
        try{   //创建 X.509 工厂类
            CertificateFactory cf = CertificateFactory.getInstance("X.509");
            //根据授权证书路径生成文件流
            FileInputStream f1 = new FileInputStream(file);
            //根据文件流获得 X.509 证书实例
            cert = cf.generateCertificate(f1);
            f1.close();
        }catch(Exception e){
            Log.e(tag,e.toString());
            }
            return cert;
        }
//验证授权证书签名的有效性
public static boolean verifyCertificate(Certificate cert){
```

```
    PublicKey pbk = cert.getPublicKey();
    try{
        cert.verify(pbk);
        return true;
        }
        catch(Exception e){
            Log.e(tag.”Certificate is invalid”);
            }
            return false;
    }
//验证证书的有效期
public static int verifyCertificateValidity(Date date,Certificate cert){
    Date d = new Date(); //获取当前时间
    int k = 0;
    X509Certificate time = (X509Certificate)cert;
    try{
        time.checkValidtity(d);
    }
    catch(CertificateExpiredException e){   //证书已过期
        Log.e(tag,”Certificate is Expired!”);
        k=-1;
        }
    catch(CertificateNotYetValidException e){ //证书还未生效
        Log.e(tag,”Certificate is early!”);
        k=-2
        }
        return k;
    }
//验证所有者签发的代理签名密钥的有效性
public static boolean verifyProxyKey(Certificate cert){
    ProxyKey pxk = cert.getProxyKey(); //获得所有者生成的代理签名密钥
    try{
        cert.proxyverify(pxk); //验证代理签名密钥的有效性
        }catch(Exception e){
            Log.e(tag.”ProxyKey is invalid”);
            return false;
            }
```

```
                return true;
        }
    //验证中间代理是否可以传递权限
    public static boolean verifyDepth(Certificate cert){
        Depth dp = cert.getDepth();//获得所有者授予的授权深度
        if(dp>0){ //判断授权深度是否大于 0
            return true;
        }else{
            return false;
            }
        }
    public void parseCertificate(Certificate cert){ //解析证书中的各类信息
        System.out.println(cert.getVersion( )); //获取版本号
        System.out.println(cert.getSerialNumber( )); //获取序列号
        System.out.println(cert.getIssuerDN( )); //获取签发者名称
        System.out.println(cert.getSigAlgName( )); //获取签名算法
        }
}
```

通过 CertificateVerify 类对证书的有效性进行验证，同时获得证书的各类信息。

5.3 房屋智能锁端设计

5.3.1 功能模块

房屋智能锁的功能模块包括匿名认证管理模块和访问控制模块，其主要工作是为用户提供开锁服务。如图 5-9 所示是房屋智能锁的功能模块图，其也分为通信层、安全层和应用层。房屋智能锁通过 Bluetooth 或 Wifi 与移动智能终端通信。

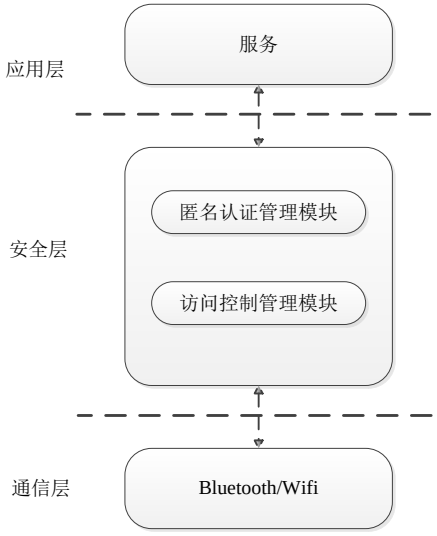


图 5-9 物联网设备功能模块

本系统中采用 Arduino[54]单片机对房屋的锁进行控制。Arduino 是一种方便灵活、容易操作的开源电子平台。Arduino 的构建是基于开放原始码的 simple I/O 介面板，同时具有类似 Java、C/C++语言的编程开发环境。其主要包括两个主要部分：一部分是硬件部分，是用来做电路连接的 Arduino 电路板，另一部分是 Arduino IDE，是计算机中的编程开发环境。通过在 IDE 中编写对应功能的代码，并将程序下载到 Arduino 电路板中，通知 Arduino 电路板执行相应的功能。Arduino 中还有各种开发板和传感器，包括许多基于蓝牙和 Wifi 的变种开发板，可以对其他的装置如灯光、马达、智能锁等进行控制。如图 5-10 所示是从代码到 Arduino 开发板的具体流程图。

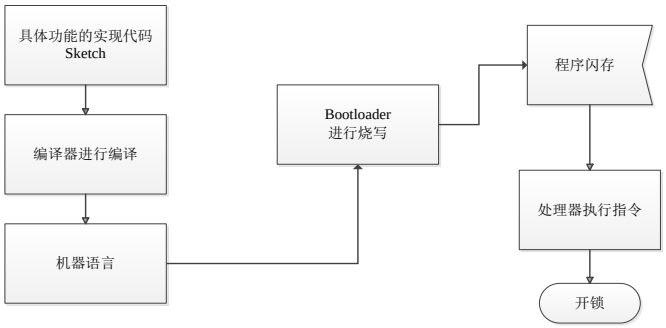


图 5-10 从代码到 Arduino 开发板的流程

在 Arduino IDE 编程并且将程序下载到电路板的过程，其实质是由编译器将代码编译为机器语言，也就是二进制语言，然后计算机又将二进制指令发送到单片机的程序闪存中，单片机再通过识别指令进行工作。Arduino 单片机可以单独完成的工作很少，其需要和其他装置比如传感器等协调进行工作。Arduino 电路板中有多个数字/模拟输入输出引脚，它通过这些引脚与其他装置建立连接，Arduino 通过驱这些引脚驱动外部装置并与它们进行交互。Arduino 单片机利用数字 I/O 接口处理数字信号，数字信号以二进制“0”和“1”或者高低电平的形式表示。

Arduino 的编程语法与 C/C++语言类似，就不再详细叙述。Arduino 官方设计了一套标准函数库[55]，如表 5.4 所示。

表 5.4 Arduino 标准函数库

函数库名称	功能说明
EEPROM	读写存储器
Ethernet/Ethernet2	以太网控制器库
GSM	GSM 控制器库
LiquidCrystal	控制 LCDs 库
SD	读写 SD 卡
SoftwareSerial	I/O 串行通信接口
WIFI	WiFi 通信接口
Wire	TWI/I2C 总线接口

通过灵活利用上述表中函数库中的函数可以通过 Arduino 单片机实现各种功能。

5.3.2 匿名认证管理模块

在本模块中，房屋智能锁对租赁者的身份进行匿名认证，房屋智能锁首先对租赁者

提交的中间代理的身份进行验证，然后验证租赁者的生成的知识签名，从而确认该租赁者是由合法的中间代理授权的合法用户，智能锁验证租赁者身份的具体操作的伪代码定义如下：

```
//智能锁验证租赁者提交的中间代理身份信息
  bool ProxyUserIdentityVerify(char* puserID, char* psignature,char* proxyPubKey)
    {
       try{
            Signature psign;//用 DSA 签名算法实例化一个代理签名
            //根据代理签名密钥对生成代理签名的公钥
            char* proxyPK = GenProxyKey(proxyPubKey);
            psign.initVerify(proxyPK); //初始化代理签名公钥
            psign.update(puserID);//更新要验证的中间代理的身份信息
            //代理签名验证，确认中间代理身份合法性
            result = psign.verify(psignature);
            }catch(exception e){
                cout<<"Authentication failed!"<< endl;
                return false;
                }
                return result;
      }
    class Signature   //定义数字签名类
    {
       public:
            char GenProxyKey(ProxyPubKey); //生成验证代理签名的公钥
            char GetInstance(SIGNATURE_ALGORITHM);//  指定数字签名的算法
            void initVerify(PublicKey); //用证书的公钥初始化验证的对象；
            void initSign(PrivateKey); //初始化用于签名的对象
            char sign(); //进行数字签名
            bool verify(signature);//验证数字签名
            void update(); //更新要被签名或验证的数据
            ……
       }
    //验证租赁者的知识签名
    bool KnowledgeSignVerify(char* message, char* ksignature)
    {
       char* hash = hashcode(message); //设备根据收到的消息计算知识签名
       if (hash == ksignature){
```

```
            cout <<"Success!"<< endl;
            return true;
            }
            return false;
    }
```

在完成了对租赁者的匿名身份认证后，租赁者和智能锁进行会话密钥的协商，建立安全的通信信道，本文基于 Diffie－Hellman 算法进行密钥协商，具体算法如下所示：

```
class keyPairGenerator //定义密钥对生成器类
{
    public:
        char* keyGetInstance(ALGORITHM);//指定密钥生成算法
        void initialize(KEY_SIZE);//初始化密钥对生成器
        char* generateKeyPair( );//生成密钥对
    }
class keyPair{//定义密钥对类
    public:
        char* getPublicKey(); //获得公钥
        char* getPrivateKey(); //获得私钥
    }
class dhParamSpec{ //定义生成密钥的材料类
    public:
        char* getParam( ); //获得构造公钥的参数
    }
class KeyAgreement { //会话密钥协商类
    public:
        char* kaInstance(ALGORITHM);
        void init( );//私钥初始化
        doPhase( );//结合对方公钥进行运算
        genSessionKey(SELECT_ALGORITHM);//生成本地会话密钥
    }
int DHSKNegotiation(){
    KeyPairGenerator DKPGen ;//实例化智能锁的密钥对生成器
    DKPGen.keyGetInstance("DH");
    DKPGen.initialize(KEY_SIZE);
    KeyPair dkeypair = DKPGen. generateKeyPair();//实例化一个密钥对
    char dpubkey = keypair. getPublicKey(); //获得智能锁公钥
    char dprikey = keypair. getPrivateKey(); //获得智能锁私钥
```

```
//智能锁通过公钥证书的形式将公钥发送给租赁者，租赁者根据设备的公钥生成密钥对，再通过公钥证书的形式将公钥给智能锁，同时根据智能锁公钥和其私钥生成会话密钥
    dhParamSpec dhParam = dpubkey.getParam();
    KeyPairGenerator UKPGen ;//实例化租赁者的密钥对生成器
    UKPGen.keyGetInstance(“DH”);
    UKPGen. initialize(KEY_SIZE);
    KeyPair ukeypair = UKPGen.generateKeyPair();//实例化租赁者的密钥对
    KeyAgreement uKeyAgree = KeyAgreement. kaInstance ("DH");//实例化密钥协商类
    uKeyAgree.init(ukeypair.getPrivateKey());
    uKeyAgree.doPhase(dpubkey,true);
    uSKey = uKeyAgree.genSessionKey(SELECT_ALGORITHM);
    dKeyAgree.doPhase(upubkey,true);
    dSKey = dKeyAgree.genSessionKey(SELECT_ALGORITHM);
    if(dSKey == uSKey)   //验证租赁者与智能锁生成的会话密钥是否一致
        cout <<”会话密钥协商成功!”<< endl;
        else
            cout <<”会话密钥协商失败!”<< endl;
    }
```

5.3.3 访问控制管理模块

租赁者在完成了身份认证和密钥协商后将他的权限凭证发送给智能锁，权限凭证中包含了每一级用户对下一级用户的信任度值。智能锁首先对权限凭证进行解析，得到租赁者的角色和每一级的信任度，然后将信任度相乘获得租赁者的综合信任度，具体操作代码如下所示：

```
int UserTrustCompute(AuthorityCredential *usercertificate)
{
    t = 1;
    for(int i=0;i<numcredential;i++)
    {
        k = usercertificate[i].getTrustValue(); //得到每个权限凭证中的信任度值
        t = t·k;   //将每个信任度值相乘
        }
        return t;
    }
//权限凭证解析类
```

```
class AuthorityCredential()
{
    public:
        char getRole();
        int getTrustValue();
    }
```

智能锁计算出租赁者的综合信任度后，检索它存储的访问控制列表，寻找用户的角色以及其对应的权限和该权限的信任度阈值，若租赁者的信任度大于权限的信任度阈值，则释放相应的权限，否则拒绝该租赁者的访问，实现了对租赁者的访问控制。本系统中的访问控制列表以 XML（Extensible Markup Language）文件形式存储。

XML 称为可扩展标记语言，它是一种结构性的数据存储语言。XML 文件由两大部分组成：文件头和文件体，其中文件体是 XML 文件的主要内容，XML 文件体的基本单元是 XML 元素。如下所示是 XML 元素的基本格式：

```
<标记名称 属性名 1 =“属性值 1”属性名 1 =“属性值 1”……>内容</标记名称>
```

XML 文件中所有的内容都一定要有标记的开始和结束，同时每个标记都在另外的标记的开始和结束标记中，形成嵌套式的标记分布，最外层的标记又称为根元素不被其他的标记包含。设备的访问控制表 3.2 可以用 XML 语言表示如下：

```
<?xml version =“1.0”encoding =“UTF-8”？ >
<Aclist>
    <guests>
        <guest id=”1”>
            <permission>p_garage</permission>
            <threshold>0.3</ threshold >
        </guest>
    </guests>
    <generals>
        <general id=”2”>
            <permission>p_garage</permission>
            <threshold>0.21</ threshold >
      </general>
       <general id=”3”>
         <permission>p_livingroom</permission>
         <threshold>0.6</ threshold >
       </general>
      ……
   </generals>
    ……
```

```
</Aclist >
```

目前有两种主流的解析 XML 文件的方式，DOM 解析（Document Object Model，文件对象模型）和 SAX 解析（Simple APIs for XML，XML 简单应用程序接口）。基于 DOM 的 XML 解析器会将 XML 文件转换成一个对象模型的集合，也通常被称作 DOM 树。DOM 接口通过分层对象模型的方式来访问 XML 文件信息，分层对象模型会依据 XML 文件的结构形成节点树。SAX 解析器则是一种基于事件的模型，当通过 SAX 解析器对 XML 文件进行解析时，它会触发一系列的事件，同时激活对应的事件处理函数，应用程序会通过事件处理函数访问 XML 文件。SAX 解析器的编码比较复杂，同时访问 XML 文件中的不同数据是很困难的。DOM 解析器生成树结构，更方便代码的编写和修改。本文采用 DOM 解析器对访问控制表进行解析。通常采用 CMark 类解析 XML 文件，其具体的方法如下所示：

```
class CMarkup
{
    public:
        bool load(MCD_CSTR fileName); //加载需要解析的 XML 文件
        bool FindElem(MCD_CSTR filename = NULL); //遍历查找的 XML 标签
        bool FindChildElem(MCD_CSTR filename = NULL);//查找子标签
        bool IntoElem();        //进入该当前标签
        bool OutOfElem();    //退出该标签
        MCD_STR GetData();    //获得当前标签对应的值
}
```

智能锁对访问控制列表的解析过程如图 5-11 所示：

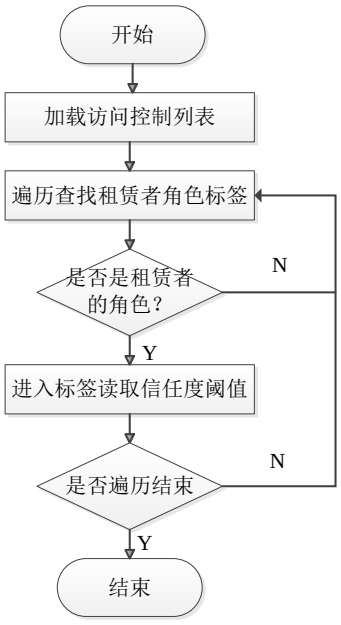


图 5-11 访问控制列表解析流程

智能锁首先加载访问控制列表文件，遍历查找租赁者被分配的角色标签，进入该角色标签，由于一个角色可能对应不同的权限，进而继续遍历该标签下的每个标签，将每

个标签下的子标签 threshold 中的值与租赁者的信任度值相比较，判断是否释放权限，打开对应的门。具体的操作代码如下所示：

```
//访问控制表的解析函数
ACLIST_Parser(FileName){
  CMark xmlAcl；  //用 CMarkup 类实例化一个 CMarkup 对象
  //加载智能锁的访问控制列表
  if (xmlAcl.Load(FileName))
  {
    //查找租赁者被分配的角色标签，如果找到就进入标签，否则就退出
    if (xmlAcl.FindElem(ROLE_TAG))
    {
       xmlAcl.IntoElem( ); //进入该角色标签
      //通过循环查找租赁者角色标签下的信息，获取对应的权限和信任度阈值
      while(xmlAcl.FindElem(subROLE_TAG))
      {
          xmlAcl.FindChildElem(“permission”); //获得子标签的元素值
          CSring pname = xmlAcl.GetChildData();
          xmlAcl.FindChildElem(“threshold”);
          CSring value = xmlAcl.GetChildData();
          int lock = pname;
          void setup( )
          {//初始化 Arduino 的变量、设置引脚的输出/输入类型等
              pinMode(lock,OUTPUT); //将连接该权限锁的引脚设置为 OUTPUT，表示由该引脚向外传输数据
              }
          if (t > value) //判断租赁者的信任度值是否大于信任度阈值
          {
              Serial.println(“Open the lock!”);
              Serial.flush( );//刷新串口
              void loop
              {
                 digitalWrite(lock,HIGH);  //将数字输入输出接口的数值设置为高电平，打开该权限对应的门
                 delay(3000);
                 }
              }else
              {
```

```
                    Serial.printl("No permission to open the lock!"); //从串口输出数据
                }
            }
        }
    }
```

5.4 系统测试与分析

本章主要内容是对整个系统的功能进行测试，房屋所有者将授权能力委托给中间代理，中间代理在获得所有者委托后向其社交范围内的用户进行授权，中间代理根据用户的访问请求授予其相应的权限。

5.4.1 测试准备

整个系统的设计包括两部分：Android 手机端的 APP 设计和智能锁端的设计。本系统中在智能锁中采用的单片机是 Arduino 单片机，通过对 Arduino 单片机进行编程实现对智能锁的控制。如图 5-12 所示是智能锁系统图，其中包括房屋锁控制器和房屋锁，其中 Arduino 单片机在控制器中。

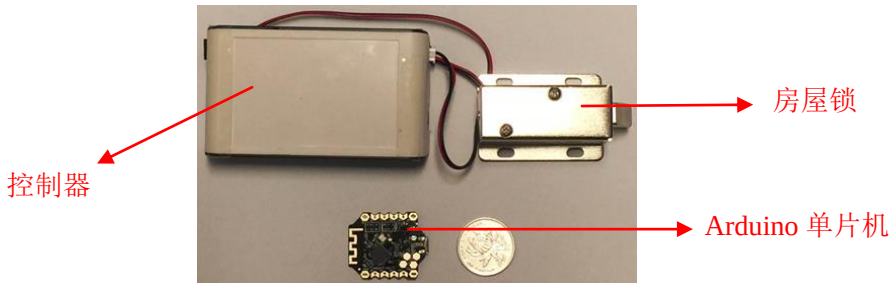
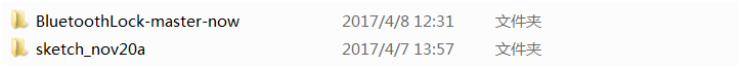


图 5-12 智能锁系统

图 5-13 是房屋租赁共享原型系统的程序列表。



(a)系统程序

(b)手机端程序

图 5-13 智能锁系统程序图

在图 5-13(a)是整个原型系统的程序，其中有两个文件，第一个文件

BluetoothLock-master-now 是 Android 手机 APP 程序；第二个文件 sketch_nov20a 是智能锁系统里 Arduino 单片机程序，采用 Arduino 提供的官方编译环境进行编译。在图 5-13(b)是 Android 手机 APP 程序，其中可执行代码在 bin 目录下。

5.4.2 系统界面测试

本文基于 Android 智能手机进行房屋共享租赁系统应用的开发，整个系统的界面结构如图 5-14 所示。整个系统包括房屋所有者操作界面、中间代理操作界面和租赁者操作界面，房屋所有者负责中间代理的管理、授权管理和时间管理，中间代理负责代理申请、用户管理、授权管理和时间管理，租赁者负责访问权限申请和扫描智能锁建立通信。

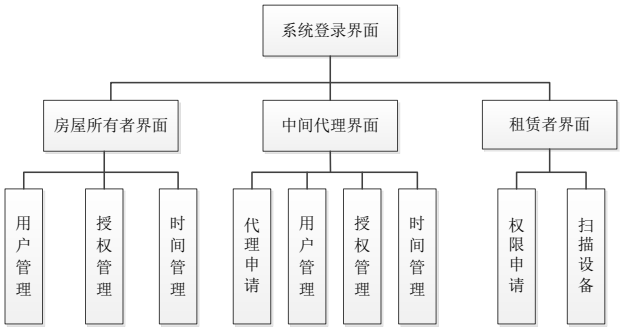


图 5-14 系统界面结构图

接下通过系统的操作界面对房屋共享租赁系统的工作流程进行简要介绍。

(a)系统登录界面图　　(b)房屋所有者操作界面　　(c)中间代理操作界面

图 5-15 系统界面图 1

基于 Android 手机的房屋共享租赁系统的登录界面如图 5-15(a)所示。该图是用户注册或登录系统的界面，新用户点击“注册”登记自己的个人信息，包括用户名、密码、真实姓名、证件号等，已注册的用户直接输入用户名和密码登录系统。房屋所有者、中间代理和租赁者的登录界面相同，系统通过用户名对登录系统的用户身份进行识别判断。

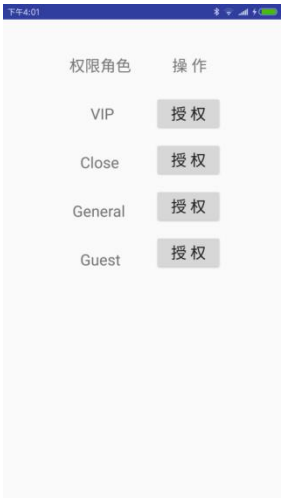
图 5-15(b)是房屋所有者的操作界面，房屋所有者的操作包括用户管理和授权管理。“用户管理”负责对注册和申请代理权的用户进行管理，“授权管理”负责向中间代理进行授权。

图 5-15(c)是中间代理的操作界面，中间代理的操作包括代理权申请、用户管理和授权管理。“代理申请”负责向所有者发出代理权限申请，“用户管理”对申请权限的租赁

者进行管理，“授权管理”负责向租赁者授权。中间代理点击“代理申请”键，在后台向所有者发送代理申请信息，其中包括个人身份信息和所有联系人手机号等信息，房屋所有者在后台根据中间代理的代理申请信息计算对中间代理的信任度，根据表 5.1 当信任度值大于 0.96 授予角色 VIP 的权限，大于 0.9 小于 0.96 授予角色 Close 和角色 General 的权限，大于 0.8 小于 0.9 授予只授予角色 General 的权限。

(a)授权管理界面

(b)时间管理界面



(c)房屋所有者委托授权界面

图 5-16 系统界面图 2

如图 5-16(a)所示是房屋所有者的授权管理界面，房屋所有者点击“添加用户”键，可以添加新的中间代理；所有者点击“修改”键可以对中间代理的有效代理时间进行设置或修改，图 5-16(b)所示是时间管理界面。房屋所有者点击授权管理界面中的“授权”键对中间代理进行角色的委托授权，如图 5-16(c)所示是房屋所有者的委托授权界面，包括角色 VIP、Close、General 和 Guest。设备所有者根据中间代理的信任度进行角色的授予，其中角色 Guest 表示该用户没有获得代理权限。

图 5-16(a)也是中间代理的授权管理界面，中间代理也可以对申请房屋权限的租赁者的访问时间进行设定和修改，中间代理点击“授权”键，出现中间代理的授权界面，如图 5-17(a)所示。在本次测试中中间代理获得的角色是 Close 和 General。

(a)中间代理授权界面

(b) 租赁者权限申请界面

(c)用户管理界面

图 5-17 系统界面图 3

图 5-17(b)是房屋租赁者权限申请界面，设备有 6 种权限，车库锁、客厅锁、卫生间锁、客房锁、书房锁和主卧锁，每个锁都对应一个“开启”操作。租赁者点击他想要打开的锁对应的“开启”键，在后台向中间代理发送他申请的权限、身份信息以及通讯录中联系人的手机号，中间代理获得租赁者的信息并判断租赁者申请的权限是否在自己角色的授权范围内，然后根据租赁者联系人的手机号计算租赁者的信任度，并将权限凭证发送给租赁者。租赁者得到授权凭证后点击“设备扫描”键，通过蓝牙扫描智能锁，并将权限凭证发送给智能锁，智能锁根据他的访问控制表决定是否对该租赁者释放权限。在本次测试中，租赁者只能申请车库锁、客厅锁、卫生间锁和客房锁。图 5-17(c)是房屋所有者和中间代理的用户管理界面，房屋所有者可以根据智能锁的反馈撤销行为表现不好的中间代理的权限，并且中间代理也可以根据反馈撤销行为表现不好的普通用户的权限。

5.4.3 系统延时性能测试

在房屋共享租赁原型系统中，租赁者得到权限打开智能锁的时间是影响租赁者用户体验的一个重要因素。在整个系统中，租赁者与智能锁间的匿名身份认证的交互过程复杂、计算量较大，是决定整个系统延时的关键。本节对系统中租赁者得到中间代理的授权后打开智能锁的时间进行测试，并与使用 Yang 和 Liu 的匿名认证方案打开智能锁的时间进行比较，如图 5-18 所示。

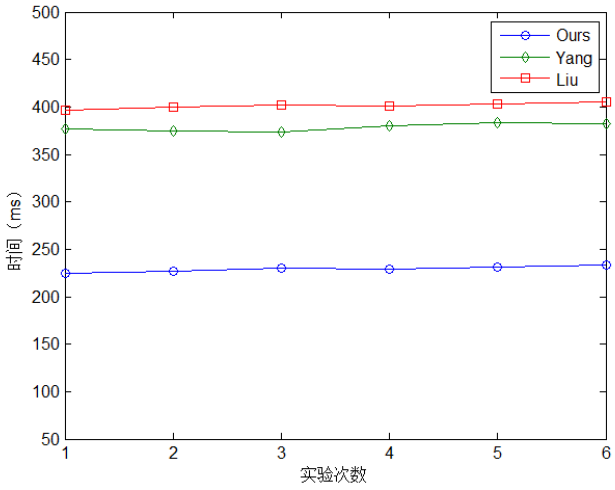


图 5-18 系统时延测试结果图

本文分别对三种认证方案打开智能锁的时间进行了 6 次测试实验，本方案、Yang 和 Liu 的开锁时间均值分别是 229.2ms，378.3ms 和 401.3ms。测试结果显示，与其他两种方案相比，本方案打开锁的时间明显减小了，有效的提高了用户体验效果，可应用于计算能力有限的物联网设备中。本次测试只是针对租赁者认证阶段的延时进行统计，测试中存在一些不可避免的测量误差。整个系统的时延存在仍然存在着很大的改进空间，这就需要对系统进一步的研究并完善相关的密码学算法，从而在保证系统安全性的条件下，提升系统效率。但是总体而言，本方案满足了无线环境下物联网设备共享服务中的匿名认证要求。

5.5 本章小结

本章主要设计并实现了一个可委托授权的可保护隐私的房屋租赁共享原型系统，首先介绍了系统的整体架构，由房屋所有者、中间代理、房屋租赁者以及房屋智能锁系统构成，可实现房屋租赁的委托授权、用户匿名身份认证和基于社交信任度的房屋访问控制，所有的交互流程通过移动智能终端进行。移动智能终端程序基于 Android 平台开发，房屋智能锁基于 Arduino 平台开发。最后对系统的功能和认证性能进行测试，测试结果表明该系统可以有效的实现远程的房屋租赁共享服务，并且减小了系统的认证延时，改善了用户体验效果和提高了系统效率。

第六章 总结与展望

6.1 全文工作总结

本文的主要工作是对物联网设备共享系统中的权限传递和隐私保护技术进行研究，在分析当前设备共享系统及其存在的问题的基础上，提出了一种可保护隐私的委托授权方案。本文的主要工作可以总结为如下几点:

本文正文部分一共分为五章内容，第一章主要介绍了本文的研究背景，首先介绍了物联网设备共享模式和物联网设备共享安全，其中对可委托授权的物联网设备共享场景下存在的安全问题进行分析；结合国内外物联网设备权限管理和基于隐私保护的身份认证方案的研究，提出了本文主要研究内容和意义。

第二章主要对在物联网设备共享中实现可保护隐私的委托授权方案的相关技术进行介绍。首先介绍了委托授权的基本概念、代理签名技术以及经典的 M-U-O 代理签名方案；其次介绍了隐私保护方面的相关技术，包括零知识证明，直接匿名认证方案和基于嵌入式设备的直接匿名认证方案，并对这些协议的优缺点进行分析；然后介绍基于信任度的访问控制的相关知识，包括基于信任度的访问控制模型、信任的概念以及计算信任度的形式化方法；最后介绍了分析协议安全性的安全模型，并主要介绍了 CK 模型。

第三章介绍了基于信任度的委托授权和访问控制。本章首先根据物联网设备共享场景中设备所有者委托授权时权限敏感度保护的问题，提出了一种基于信任度的可控的部分权限委托授权机制，介绍了该方案的具体流程以及实现原理，该机制基于 Mambo-Usauda-Okamoto 代理签名方案的基础上进行了改进，中间代理使用的代理签名私钥由其与设备所有者共同生成，最后文中给出了相应安全性分析；然后根据用户访问物联网设备时中间代理过度授权的问题，提出了一种基于信任度的访问控制机制，该机制采用基于角色和信任度的访问控制模型，根据用户动态社交关系所生成的信任度和设备所有者设定的信任度阈值确定是否授权；最后，提出了一种基于动态社交关系的信任度生成方案，该方案根据用户间联系人、社交联系地址和使用的手机软件的相似性实时的生成用户当前的信任度。

第四章介绍了用户与物联网设备间的可保护的隐私的身份认证方案。本方案主要是基于代理签名和知识证明技术，采用信任委托的思想，通过代理签名技术实现了设备对中间代理的身份验证，通过验证用户提交的知识签名在不需要用户身份信息的前提下验证用户是否是中间代理授权的用户，并且在认证过程中进行会话密钥的协商，增强了匿名认证方案的安全性。最后采用 Canetti-Krawczyk 模型对该方案的认证安全性进行了形式化分析和证明。

第五章设计并实现了一个可控委托授权的隐私保护房屋租赁共享原型系统，该系统由房屋所有者、中间代理、房屋租赁者以及房屋智能锁构成，整个系统功能包括委托授权、匿名身份认证和访问控制。随后分别对移动终端和物联网设备端功能的具体实现进

行介绍，移动终端设计包括身份管理模块、信任度管理模块和授权授权证书管理模块；物联网设备端的设计包括匿名认证管理模块和访问控制管理模块。最后对整个系统进行功能性测试和性能测试。

第六章对全文工作进行总结，指出研究工作中存在的不足，并对将来物联网设备共享系统中的研究方向进行展望。

6.2 进一步研究方向

1) 本文对移动智能终端应用软件 APP 的设计主要是基于 Android 平台，并没有对其他比如 iOS、BlackBerry OS 等系统的 APP 应用进行开发。因此下一步 APP 应用的开发也会基于其他手机操作系统研究。
2) 在此次的研究中，设备所有者的访问控制列表还是由人为指定，具有一定的主观性，针对这一问题，设计出更加智能、客观的信任度阈值是下一步研究的重点。
3) 在系统设计中，中间代理还只是向其直接用户进行授权，还没有实现设备所有者权限的逐级向下传递。所以针对这个问题，下一步系统设计的重点是实现设备使用权限的多次传递，扩大可使用设备的用户范围。
4) 本文设计的系统在计算用户的信任度时，只是基于用户间联系人的相似度进行信任度的计算，所以在未来的研究中加入更多反映用户间亲密程度的变化因素，使得信任度的计算更加的完善。

共享经济是当今社会最受瞩目的热点问题之一，人们可以将闲置的设备共享给别人使用，从而可以提高社会资源的利用率，并可以从中获得回报。但是在设备分享过程中不可避免的出现的权限管理和安全隐私问题，将会阻碍设备共享的应用。因此如何更好的解决物联网环境中设备共享的安全和隐私问题将是未来研究的重点。

参考文献

[1] Jin J, Gubbi J, Marusic S, et al. An information framework for creating a smart city through internet of things[J]. IEEE Internet of Things Journal, 2014, 1(2): 112-121.

[2] B ¢issent J. Getting clever about smart cities: New opportunities require new business models[J]. 2010.

[3] 李德仁,姚远,邵振峰. 智慧城市中的大数据[J]. 武汉大学学报(信息科学版),2014,(06):631-640.

[4] 陈龙彪,李石坚,潘纲. 智能手机:普适感知与应用[J]. 计算机学报,2015,(02):423-438.

[5] 周礼艳. 基于 O2O 的共享经济商业模式分析及构建[J]. 商业时代, 2016(22):69-71.

[6] Li S, Da Xu L, Zhao S. The internet of things: a survey[J]. Information Systems Frontiers, 2015, 17(2): 243-259.

[7] 杰里米·里夫金. 走向物联网和共享经济[J]. 企业研究, 2015(2):14-21.

[8] Sandhu R S, Coyne E J, Feinstein H L, et al. Role-based access control models[J]. Computer, 1996, 29(2): 38-47.

[9] Zhang G, Tian J. An extended role based access control model for the Internet of Things[C]//Information Networking and Automation (ICINA), 2010 International Conference on. IEEE, 2010, 1: V1-319-V1-323.

[10] Fu Y, Ye C. Using XACML to define access control policy in information system[C]//Wireless, Mobile and Sensor Networks, 2007.(CCWMSN07). IET Conference on. IET, 2007: 676-679.

[11] Li N, Winsborough W H, Mitchell J C. Distributed credential chain discovery in trust management: extended abstract[C]// CCS 2001, Proceedings of the, ACM Conference on Computer and Communications Security, Philadelphia, Pennsylvania, Usa, November. DBLP, 2001:156-165.

[12] Kolev A, Čobanov, S. Trustbac—Integrating Trust Relationships Into The Rbac Model For Access Control In Open Systems[C]// SACMAT 2006, ACM Symposium on Access Control MODELS and Technologies, Lake Tahoe, California, Usa, June 7-9, 2006, Proceedings. DBLP, 2006:49-58.

[13] 翟征德,冯登国,徐震. 细粒度的基于信任度的可控委托授权模型[J]. 软件学报,2007,(08):2002-2015.

[14] Juels A, Wattenberg M. A fuzzy commitment scheme[C]//Proceedings of the 6th ACM conference on Computer and communications security. ACM, 1999: 28-36.

[15] Yu J, Wang G, Mu Y, et al. An efficient generic framework for three-factor authentication with provably secure instantiation[J]. IEEE Transactions on Information Forensics and Security, 2014, 9(12): 2302-2313.

[16] Xue K, Ma C, Hong P, et al. A temporal-credential-based mutual authentication and key agreement scheme for wireless sensor networks[J]. Journal of Network and Computer Applications, 2013, 36(1): 316-323.

[17] Jiang Q, Ma J, Lu X, et al. An efficient two-factor user authentication scheme with unlinkability for

wireless sensor networks[J]. Peer-to-Peer Networking and Applications, 2015, 8(6): 1070-1081.

[18] Lysyanskaya A, Rivest R L, Sahai A, et al. Pseudonym systems[C]//International Workshop on Selected Areas in Cryptography. Springer Berlin Heidelberg, 1999: 184-199.

[19] Brickell E, Camenisch J, Chen L. Direct anonymous attestation[C]//Proceedings of the 11th ACM conference on Computer and communications security. ACM, 2004: 132-145.

[20] 甄鸿鹄. 可信匿名认证的研究与应用[D].解放军信息工程大学,2009.

[21] Derler D, Potzmader K, Winter J, et al. Anonymous Ticketing for NFC-Enabled Mobile Phones[J]. 2011.

[22] Yang L, Ma J, Lou W, et al. A delegation based cross trusted domain direct anonymous attestation scheme[J]. Computer Networks, 2015, 81: 245-257.

[23] Mubarak M F, Ab Manan J L, Yahya S. An Implementation of a Unified Security, Trust and Privacy (STP) Framework for Future Integrated RFID System[M]//Future Data and Security Engineering. Springer International Publishing, 2014: 122-135.

[24] Gope P, Hwang T. A realistic lightweight authentication protocol preserving strong anonymity for securing RFID system[J]. Computers & Security, 2015, 55: 271-280.

[25] 陆颖颖. 物联网隐私保护策略的研究与应用[D].南京邮电大学,2012.

[26] 张俊松. 物联网环境下的安全与隐私保护关键问题研究[D].北京邮电大学,2014.

[27] 张磊. 车载自组织网络安全认证与隐私保护的研究和实现[D].安徽大学,2016.

[28] 姜顺荣. 物联网中信息共享的安全和隐私保护的研究[D].西安电子科技大学,2016.

[29] 肖玥. 委托授权模型的研究[D].中国人民解放军信息工程大学,2005.

[30] Mambo M, Usuda K, Okamoto E. Proxy signatures: Delegation of the power to sign messages[J]. IEICE transactions on fundamentals of electronics, communications and computer sciences, 1996, 79(9): 1338-1354.

[31] Lee B, Kim H, Kim K. Strong proxy signature and its applications[C]//Proceedings of SCIS. 2001, 2001: 603-608.

[32] Chaum D, Van Heyst E. Group signatures[C]//Workshop on the Theory and Application of of Cryptographic Techniques. Springer, Berlin, Heidelberg, 1991: 257-265.

[33] Camenisch J, Michels M. A group signature scheme with improved efficiency[C]//International Conference on the Theory and Application of Cryptology and Information Security. Springer Berlin Heidelberg, 1998: 160-174.

[34] Goldwasser S, Micali S, Rackoff C. The knowledge complexity of interactive proof systems[J]. SIAM Journal on computing, 1989, 18(1): 186-208.

[35] Alliance T C P. TCPA main specification v. 1.1 b[J]. 2005.

[36] Alliance T C P. TCPA main specification v. 1.2[J]. 2006.

[37] Camenisch J, Groth J. Group signatures: Better efficiency and new theoretical aspects[C]//International Conference on Security in Communication Networks. Springer Berlin Heidelberg, 2004: 120-133.

[38] Ge H, Tate S R. A direct anonymous attestation scheme for embedded devices[C]//International Workshop on Public Key Cryptography. Springer Berlin Heidelberg, 2007: 16-30.

[39] 刘武，段海新，张洪,等. TRBAC:基于信任的访问控制模型[J]. 计算机研究与发展，2011, 48(8):1414-1420.

[40] 高尚,胡爱群,石乐,陈先棒. 安全协议形式化分析研究[J]. 密码学报,2014,(05):504-512.

[41] 鲁来凤. 安全协议形式化分析理论与应用研究[D]. 西安电子科技大学, 2012.

[42] Canetti R, Krawczyk H. Analysis of key-exchange protocols and their use for building secure channels[C]//International Conference on the Theory and Applications of Cryptographic Techniques. Springer Berlin Heidelberg, 2001: 453-474.

[43] 殷安生. 可信网络中信任评估机制若干关键技术研究[D].南京邮电大学,2015.

[44] Chen R, Guo J, Bao F. Trust management for soa-based iot and its application to service composition[J]. IEEE Transactions on Services Computing, 2016, 9(3): 482-495.

[45] 杨颖涛，王跃钢，邓卫强,等. 基于共轭先验分布的贝叶斯网络分类模型[J]. 控制与决策，2012, 27(9):1393-1396.

[46] 钟小伟,傅鸿源. 基于β分布的区间估计量化方法[J]. 数学的实践与认识,2011,(17):90-95.

[47] Jing-sen L, Guan-zhong D, Yu L. A TPM authentication scheme for mobile IP[C]//Computational Intelligence and Security Workshops, 2007. CISW 2007. International Conference on. IEEE, 2007: 721-724.

[48] YANG Y, CAO L, LI Z. A novel Direct Anonymous Attestation protocol based on zero knowledge proof for different trusted domains[J]. China Communications, 2010, 7(4): 172-175.

[49] 周彦伟,杨波,吴振强,何聚厚,李骏. 基于身份的跨域直接匿名认证机制[J]. 中国科学:信息科学,2014,(09):1102-1120.

[50] Boyar J, Friedl K, Lund C. Practical zero-knowledge proofs: Giving hints and using deficiencies[J]. Journal of cryptology, 1991, 4(3): 185-206.

[51] Van De Graaf J, Peralta R. A simple and secure way to show the validity of your public key[C]//Conference on the Theory and Application of Cryptographic Techniques. Springer Berlin Heidelberg, 1987: 128-134.

[52] Bellare M, Canetti R, Krawczyk H. A modular approach to the design and analysis of authentication and key exchange protocols[C]//Proceedings of the thirtieth annual ACM symposium on Theory of computing. ACM, 1998: 419-428.

[53] 罗建利. Android 手机数据取证在案件侦破中的应用研究[D].华南理工大学,2016.

[54] Arduino.[EB/OL].http://baike.baidu.com/link?url=Zd2Ow1vQmL1TmQ7766otEWCcqEaiiRGaE_Of9WqM5yJ097JZi32P-HQzNvHt8NvgDlZ8EW71Ftlb_W4P0pQOi_,2017.

[55] Libraries.[EB/OL]. https://www.arduino.cc/en/Reference/Libraries,2017.

[56] Li N, Winsborough W H, Mitchell J C. Distributed credential chain discovery in trust management[J]. Journal of Computer Security, 2003, 11(1): 35-86.