ITTDAF：不依赖父权限信息传递的去中心化授权框架

罗期丰[1], 石瑞生[1]

[1]北京邮电大学 网络空间安全学院 北京 中国 100876

摘要 为了解决现有去中心化授权协议在支持传递权限时需要传递父权限信息从而容易导致权限信息泄露的问题以及单个用户信息泄露会威胁到其他用户权限的机密性的问题, 本文提出了基于检索树结构和可信平台模块的去中心化授权框架 ITTDAF, 其核心思想是用户在授予其他用户权限时, 需要将授权信息告知提供相关资源的实体, 由资源实体基于授权信息生成检索树结构, 得知权限的传递关系。当用户在向资源实体请求资源时只提供自己拥有的权限信息即可证明权限有效性, 并不需要用户得知父权限的相关信息。避免了用户的权限信息泄露对其他用户的权限信息机密性的破坏, 同时降低了权限验证所需传输的数据量并减少权限验证所需要的时间。所有信息通过可信平台模块进行签名, 以保证数据的来源的唯一性并实现权限与设备的绑定, 使得权限信息不会在非用户设备上得到执行。相较于比对方案, 在相同条件下本文所提出的方案在描述权限所需数据量上缩小 44.2%, 权限验证所需时间减少 51.2%, 在拥有更高安全性的同时, 也有着更好的可用性。

关键词 加密与解密; 访问控制; 去中心化授权; 可传递授权; 可信平台模块(TPM)

中图法分类号 TP309.7 DOI 号 10.19363/J.cnki.cn10-1380/tn.2022.03.11

ITTDAF: Decentralized Authorization Framework That Does Not Rely on The Transmission of Parent Permission Information

LUO Qifeng[1], SHI Ruisheng[1]

[1]School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract To solve the problem that the existing decentralized authorization protocols need to transmit the parent permission information when one user transmits permissions to the other user, which is easy to cause the permission information leakage and threaten the confidentiality of other users' permission information, this paper proposes a decentralized authorization framework ITTDAF(Index Tree & TPM based Decentralized Authorization Framework)based on index tree structure and trusted platform module. The core idea is that when one user authorizes permission to other users, the authorizing user needs to send the authorization information to the entities which providing relevant resources. The resource entity generates an index tree structure based on the authorization information sent by authorization user to know the transmission relationship of permissions between users. When one user requests resources from a resource entity, the user only needs to provide his own permission information to the resource entity to prove the validity of its permission and does not need to know any of the parent permission information. The permission information does not contain the relevant information of the parent permission, so as to avoid the damage of the permission information leakage to the confidentiality of the permission information of other users. This decreases the amount of data that needs to be transmitted and time consume of permission validation made by resource entity. All information is signed by the user device's trusted platform module to ensure the source of data is from user and realize the binding between permission and device to let the permission can't be execute on other user's device. Compared with the comparison scheme, under the same conditions, the scheme proposed in this paper reduces the amount of data required to describe permissions by 44.2% and the time required for permission verification by 51.2%. It not only has higher security, but also has better availability.

Key words encryption and decryption; access control; decentralized authorization; delegate authorization; trusted platform module(TPM)

通讯作者： 石瑞生, Email: shiruisheng@bupt.edu.cn

本课题得到国家重点研发计划项目(No. 2020YFB1005500), 北京市自然科学基金项目(No. M21037), 广东省重点研发计划项目(No. 2019B010137003), 北京市自然科学基金项目(No. M21034)资助。

收稿日期: 2021-10-26; 修改日期: 2022-01-08; 定稿日期: 2022-01-06

1 前言

传统的权限管理基于单一的可信中心服务, 其易于部署, 便于统一管理。但依赖可信中心服务面临单点故障的问题: 一旦该可信中心服务瘫痪整个访问控制系统都会受到影响。因此权限的管理需要去中心化, 并通过区块链、智能合约等防篡改技术进行去中心化的存储以及权限管理。

近几年已有多个与去中心化技术相结合的访问控制方案被提出[1-16]。但是部分方案[1-12]只做到了权限的去中心化存储。权限的分发依然是由单个或多个授权中心进行分发, 依然面临单点故障的问题。其余方案的授权操作不完全依赖中心授权实体: 用户在接收到授权中心的权限信息后可以将拥有的部分权限传递给其他用户。但是 BlendCAC[13] 和 CapBAC[14]未考虑权限的机密性, 使得攻击者在获得权限明文后可根据需要选择性的对用户发起攻击。DCACI[15]和 WAVE[16]保护了授权信息的机密性, 其共同点是让资源实体解密多个经过加密的证书, 从而验证用户拥有的权限的来源及有效性。其中 DCACI 依赖 IOTA 的限制模式以及单一主密钥及主密钥的 SHA256 值生成的侧密钥。WAVE 则是将父权限的解密密钥包含于子权限信息当中。权限拥有者可解密该权限颁发者的权限信息, 并通过这一操作证明权限拥有者的权限来源及有效性。

已有的去中心化且支持传递授权的授权协议在设计上存在如下问题, 包括: ①单个用户的密钥泄露会威胁到使用授权协议的所有用户的权限机密性。如 DCACI 的侧密钥泄露或 WAVE 的密钥泄露。②权限机密性保护不足。其体现在合法用户依然会了解父权限的部分信息。如使用 WAVE 授权协议的用户可通过权限信息中的解密密钥了解其他用户的权限信息。权限机密性对于授权协议来说是相当重要的。如果无法保护权限的机密性, 攻击者便可对拥有攻击者需要的权限且更易攻破的用户发起攻击。同时在一些保密项目中, 用户所拥有的权限与其所负责的项目相关。因此权限信息即使由合法用户而非攻击者得知也是不合适的。③设备绑定: 上述方案中的授权信息只与用户信息绑定, 但是未与设备绑定。攻击者通过钓鱼链接、暴力猜测等方式在获取到用户认证信息后, 假冒用户的身份在攻击者的设备上执行恶意操作。

为了解决这些问题, 本文基于 WAVE 的部分思想提出基于检索树和可信平台模块的去中心化授权框架: ITTDAF(Index Tree & TPM based Decentralized Authorization Framework)。相较于其他授权协议, 本文所提出的授权协议要求用户在授予另一个用户权限时需要将权限信息发送到与权限相关的资源实体处。资源实体通过收到的授权信息生成一个可检索权限来源的树结构。通过存储在资源实体的该结构, ITTDAF 不依赖父权限信息的传递, 即可允许用户证明权限的来源及有效性。相较于其他协议, 该授权协议拥有较低的密钥维护难度, 用户只需要维护自己的加密密钥和签名密钥即可。当用户的密钥泄露时除自己的权限外不会破坏其他用户的权限的机密性。在该协议中, 除资源实体外用户只会知道其所收到的权限以及其所下发的权限, 而不会获得其他用户的权限信息。通过可信平台模块(Trusted Platform Module, TPM), 该授权协议可保证权限信息的来源并保证权限只能在指定的设备上执行。

相较于现有支持传递授权的访问控制方案, 本论文所提出的方案重点在于将“证明权限来源”这一操作由用户实体证明转换为资源实体自证明。若由用户对权限来源进行证明, 用户就需要获得其他用户的权限的相关信息, 如解密密钥、颁发者的签名等信息。而这会面临信息泄露的风险进而威胁其他实体的权限机密性。权限的颁发者对其拥有的权限与颁发的权限均有知情能力。因此权限颁发者可将其颁发的权限信息与之前接收到的权限信息均告知资源实体。资源实体可对这两个权限信息的继承关系进行验证并存储。当用户发起操作请求时, 用户只需提供其拥有的权限信息, 而资源实体可通过已存储的权限继承关系对权限的来源及有效性完成验证。

在权限机密性的保护上, 本论文采用 RSA[17] 非对称加密算法加密权限的具体内容, 保证权限信息存储在远程数据库时只能由权限拥有者对权限进行解密。为了避免资源实体存储的权限信息数据体积过大, 资源实体生成的检索树的节点中仅存储通过 SHA256 算法生成的权限信息摘要值与用于远程检索权限信息验证权限有效性的哈希值。在保证数据来源唯一性上, 本论文使用 TPM 生成的不可迁移密钥生成的数据完整性签名保证数据来源唯一性。

2 可信平台模块

可信平台模块(Trusted Platform Module, TPM)是一个含有密码运算部件和存储部件的小型片上系统, 可包含于传统计算机以及手机、平板等可移动设备,

对外可提供非对称密钥生成运算、非对称算法加密解密运算、数字签名运算和随机数产生运算。TPM平台可提供完整性度量、存储和报告, 远程证明, 数据保护和密钥管理 4 大功能。密钥管理功能中 TPM 安全管理 7 类密钥的生成、使用和存储, 7 类密钥中的签名密钥用于对应用数据和信息签名。签名密钥可以是迁移或不可迁移的。可迁移密钥能够在 TPM 之间传递, 而不可迁移密钥则与 TPM 芯片绑定[18]。

任何能使用 TPM 创建密钥的用户都可以创建一个受限的签名密钥, 密钥创建者可以要求一个可信第三方, 如证书颁发机构为该密钥提供一个证书, 证书颁发机构可以向调用者请求一些关于该密钥的证据, 用以表明被认证的密钥确实来自 TPM。通常提供的证据为使用同一个 TPM 上已经被证明过的一个其他密钥, 如背书密钥的证书或者平台证书。背书密钥是永远不会暴露在 TPM 外部的非迁移解密密钥。对于 TPM 或一个平台而言, 背书密钥是唯一的, 也因此由该 TPM 产生的背书密钥签名的不可迁移的签名密钥也是唯一的。该签名密钥生成的数据的签名可用于证明数据的真实来源。

本论文中, 可信平台模块用于保证操作请求来源的唯一性, 即用户的操作请求能且只能在该用户拥有的设备上产生。用户在执行操作前需要拥有自己的帐户。用户在注册帐户信息时, 除用户名和密码等基本认证信息外, 本论文要求用户提交由设备的可信平台模块生成的不可迁移签名密钥的公钥信息。当用户执行任意操作时均需提供由该签名密钥生成的签名信息用于证明操作请求的来源。这样即使用户的身份数据和权限数据遭到泄露, 攻击者在未得到用户设备的情况下也无法假冒用户执行恶意操作。即使用户设备丢失, 用户也可通过挂失设备的方式将设备与用户信息解绑, 保证权限不会被获得设备的攻击者恶意使用。然而可信平台模块生成的签名仅能保证权限信息的唯一性(即来自特定的用户), 而不能保证权限信息的有效性。因为权限信息中并不包含与其相关的权限(如父权限)的相关信息, 因此依然需要资源实体自己完成权限来源的验证。如何判断密钥来自可信平台模块而不是由任意软件程序产生不在本论文的讨论范围之内。

3 ITTDAF 的设计与实现

3.1 核心思想

在一个可传递权限的去中心化授权协议中, 当用户向资源实体发起权限验证时, 资源实体需要溯源权限的源头以确定权限的有效性。例如资源由 A 管理, A 向 B 授予了访问资源的权利, B 又授予 C 访问资源的权力。当 C 访问资源时, 资源服务器需要知道 C 的权限是来自 B, 而 B 的权限是来自根权限 A。因此, 当权限分发实体分发权限信息时, 其可以将自己拥有的权限信息与将要发送的权限信息一起发送到资源实体。资源实体接收到两个权限信息后可生成用于检索权限来源的树结构(后文统称为检索树), 如图 1 所示。

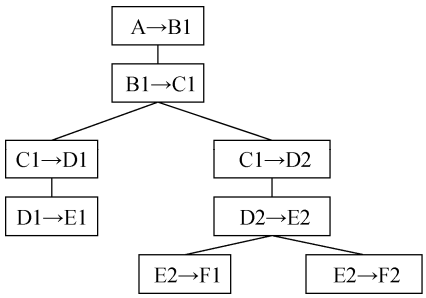


图 1 资源实体内的检索树结构

Figure 1 Index tree structure in resource entity

通过该结构, 资源实体可获悉有哪些用户需要访问该资源以及这些实体的权限是由谁发送的。当一个有相关权限的用户访问资源实体时, 资源实体首先需要判断该用户所提供的权限信息是否记录于检索树结构的某个节点中。若存在再判断从存储该权限信息的节点开始到检索树结构的根节点结束这一路径上所涉及的所有权限是否均有效。若均有效, 如果用户的权限允许执行请求的操作则资源实体执行该操作, 反之则拒绝。

在 ITTDAF 中, 每一个实体所下发的权限信息以证书的形式来描述, 证书的具体格式如图 2 所示。

明文信息头
1. 撤销口令revoke的哈希值
2. 一次性私钥生成的签名
3. 一次性公钥
4. 证书过期时间

权限声明 (使用证书接收者的加密公钥加密)
1. 权限集合 = $\{p_1, p_2, p_3, p_4, \cdots, p_n\}$
p_i = {操作对象，操作行为，是否可传递}
2. 证书接收者ID
3. 证书发布者ID
4. 对一次性公钥生成的签名

图 2 权限证书的结构

Figure 2 Structure of permission certificate

证书包含明文信息头和权限声明两个部分。明文信息头以明文形式存储, 用于验证权限证书是否被撤销或者过期, 权限声明部分使用证书接收者的加密密钥进行加密, 以保护证书接收者权限的机密

性。相较于其他方案, ITTDAF 的权限证书不包含父权限的信息或者权限颁发实体的除 ID 外的相关信息。这使得即使该证书的数据被攻击者获取, 攻击者也无法获得其他用户的权限信息。同时这一设计降低了证书在加密设计上的复杂度并降低了数据占用空间。

明文信息头部分中, 撤销口令 *revoke* 为基于证书信息所生成的签名的哈希值, 其生成方式如公式 1 所示:

$$revoke = Hash(Sign(sk_{TPM}; subID, issID, policy\_set, exp\_time)) \tag{1}$$

式(1)中: sk_{TPM} 为用户设备的 TPM 中存储的不可迁移签名密钥, 其与证书发布者相对应。*subID* 为证书接收者的身份标识, 用于唯一标识证书接收者。*issID* 为证书发布者的身份标识, 用于唯一标识证书发布者。*policy_set* 为权限集合, *exp_time* 为该权限证书的过期时间。撤销口令存储于证书发布者所拥有的设备中。当需要撤销证书时, 证书发布者需要提交撤销口令到证书存储服务器中。当资源实体远程检索证书检索到该证书的撤销口令 *revoke* 值时, 若该撤销口令 *revoke* 值的哈希值与证书中存储的撤销口令的哈希值一致, 则认为该证书无效。

一次性公钥由证书发布者在发布证书时随机生成。一次性公钥对生成的证书签名方式为:

$$sign_{att} = Sign(sk_{OTP}; plainHeader \mid sign_{att}, cipher\_auth) \tag{2}$$

式(2)中: sk_{OTP} 为一次性(One Time)公钥对的私钥。不直接使用权限颁发者的私钥对证书签名是为了避免攻击者通过暴力测试的方式通过公钥获悉证书的真实颁发者。“$plainHeader \mid sign_{att}$”代表除签名外的明文信息头, *cipher_auth* 则代表加密后的权限声明。

“权限集合”由一到多个权限细则组成。每个权限细则主要包含操作对象、操作行为、是否可下发 3 个参数。操作对象为具体的被操作主体, 如数据库、数据表等。操作行为指对被操作对象所执行的操作, 如创建数据表、向表中插入数据等。是否可下发则决定该权限是否可由一个用户传递给另一个用户。

“证书接收者 ID”和“证书发布者 ID”为一长字符串, 用于唯一标识证书接收者和证书发布者。

资源实体可通过权限证书判断用户所下发的权限与用户所拥有的权限是否具有继承关系。权限信息之间具有继承关系是指父权限证书的拥有者是子权限证书的颁发者, 同时子权限证书的权限集合是父权限证书的权限集合的子集。

ITTDAF 授权协议包含 4 个方法: 权限分发方法、权限验证方法、权限更新方法和权限撤销方法。其中权限分发方法用于权限拥有者向其他用户分发权限证书进而授予该用户证书所描述的权限。权限验证方法指用户携带描述权限的证书以及身份凭证等信息向资源实体发起操作请求时, 资源实体如何验证用户所提供的信息并判断用户是否有权执行请求的操作。权限更新方法指权限拥有者如何对所下发的权限进行更新。权限撤销方法指权限拥有者如何对所下发的权限进行撤销操作。协议中所涉及的变量及符号说明如表 1 所示。

表 1 ITTDAF 符号说明

Table 1 Inform of symbol in ITTDAF

符号	说明
$entity_{grant}$	发送权限证书的用户实体, 该实体可根据其所拥有的权限证书向其他普通用户实体发送权限证书;
$entity_{subject}$	接收权限证书的用户实体;
$entity_{req}$	请求资源的实体, 该实体向资源实体提交权限证书与具体操作请求以获取所需求的资源;
$entity_{admin}$	资源管理实体, 生成访问资源实体的根权限证书以及检索树结构的根节点;
$entity_{res}$	资源实体, 由资源管理实体直接管理, 并向有合法权限的资源请求实体提供其所需的资源;
att_{send}	实体所颁发的权限证书, 该证书的权限声明部分未被加密;
$att_{receive}$	实体所拥有的权限证书, 该证书的权限声明部分未被加密;
$att_{A\_TO\_B}$	由 $entity_A$ 向 $entity_B$ 所颁发的证书;
$update\_att_{A\_TO\_B}$	由 $entity_A$ 在撤销 $att_{A\_TO\_B}$ 后向 $entity_B$ 所颁发的证书;
$enc\_update\_att_i$	被加密的 $update\_att_i$。

续表

符号	说明
enc_att_{send}	实体所颁发的权限证书, 该证书的权限声明部分通过该证书的接收者的公钥加密;
enc_att_{send}`	资源实体通过 att_{send} 的接收者的加密公钥对 att_{send} 的权限声明部分进行加密所生成的权限证书;
$enc\_att_{receive}$	实体所拥有的权限证书, 该证书的权限声明通过该证书的接收者的公钥加密;
$hash_i$	用于检索 enc_att_i 的哈希值;
$update\_hash_i$	用于检索 $update\_att_i$ 的哈希值;
$sign_{entity}$	由 *entity* 所生成的签名, 签名均由 TPM 模块的非迁移密钥生成;
$SHA256_i$	通过 att_i 生成的 SHA256 值;
$UPDATE\_SHA256_i$	通过 $update\_att_i$ 生成的 SHA256 值;
$DB_{证书}$	权限证书存储服务器;
set_{hash}	通过远程检索证书用的哈希值生成的集合;
exp_{set}	集合的最终有效时间;
timestamp	当前时间戳, 防止重放攻击。

3.2 初始环境部署

本文假设各实体通信采用安全传输层协议连接(TLS)等方式以保证数据传输的安全性并不会被第三方监听。各实体的身份凭证数据、公钥数据等信息已存储于数据库中, TPM 公钥数据与用户信息数据已绑定, 如图 3 所示

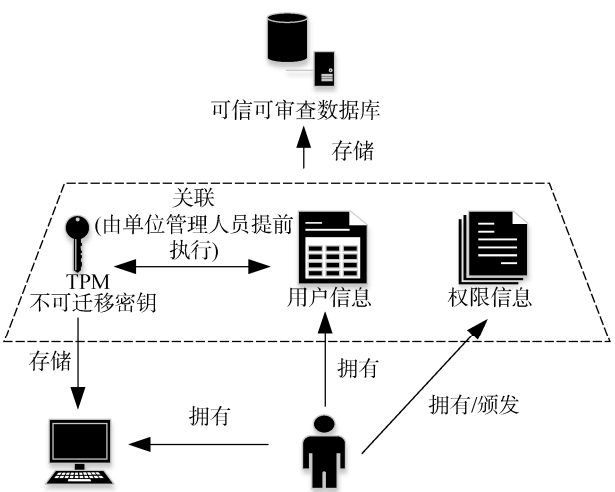


图 3 用户信息与 TPM 密钥存储于可信数据库中
Figure 3 User information and TPM key are stored in a trusted database

ITTDAF 不依赖特定的去中心化存储方案, 本文假设证书存储服务器中的数据是可信可审查且不可被篡改的。

3.3 权限分发

权限分发方法分为“资源管理员权限分发”与“普通用户权限分发”两种。其中“资源管理员权限分发”指资源实体的管理员向用户分发权限, 允许用户访问相关资源。资源管理员在发放权限时不需要提交其已拥有的权限信息到资源实体。“普通用户权限分发”指已被分配权限的用户向未被分配权限的用户颁发权限以允许后者访问相关资源。两种权限分发方法主要包括五步, 如图 4 所示。

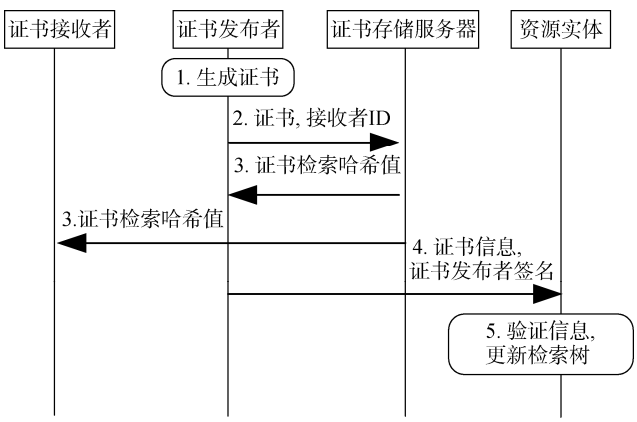


图 4 权限分发方法
Figure 4 Method of delegating permissions

与“资源管理员权限分发”不同, “普通用户权限分发”需要用户同时提交已拥有的权限信息与将要发送的权限信息到资源实体用于证明用户确实有权利下发权限给其他用户。“资源管理员权限分发”的具体方法如 **过程 1** 所示。

过程 1. 资源管理员权限分发

1 $entity_{grant}$ 生成其所欲授予的权限生成证书: att_{send}, 使用证书接收者的公钥对权限声明部分进行加密生成加密证书: enc_att_{send}

2 $entity_{grant}$ 将 enc_att_{send} 和 $entity_{subject}$ 的 ID 发

送到 $DB_{证书}$

3 $DB_{证书}$对 enc_att_{send} 进行存储, 生成检索该加密证书的哈希值: $hash_{send}$, 并将 $hash_{send}$ 发送到该证书的 $entity_{grant}$ 与 $entity_{subject}$

4 $entity_{grant}$ 将 att_{send}, $hash_{send}$, *timestamp* 以及 $sign_{admin}$ 发送到 $entity_{res}$

5 $entity_{res}$ 验证 *timestamp* 和 $sign_{admin}$ 是否有效

6 IF 验证未通过

7 拒绝执行

8 ELSE

9 生成检索树结构的根节点, 该节点存储通过 att_{send} 生成的 $SHA256_{send}$ 以及用于检索 $DB_{证书}$获得证书的 $hash_{send}$

"普通用户权限分发"的具体方法如 **过程 2** 所示。由于资源管理实体与普通用户产生权限证书的方法相同, 因此**过程 2** 仅描述资源实体如何验证用户产生的权限证书的有效性并将证书存储于检索树中。

过程 2. 普通用户权限分发

输入: att_{send}, $att_{receive}$, $hash_{send}$, *timestamp*, $sign_{grant}$

输出: 权限添加结果: *result*

1 检索 $entity_{grant}$ 的 TPM 公钥

2 验证 *timestamp* 和 $sign_{grant}$ 是否有效

3 IF 验证未通过

4 *result*←"拒绝执行"

5 ELSE

6 通过 $att_{receive}$ 生成 $SHA256_{receive}$ 搜索检索树, 获取与该权限证书相关的所有权限证书的远程检索哈希值

7 通过 $hash_{receive}$ 及上游证书的远程检索哈希值检索 $DB_{证书}$

8 IF 检索到的证书被撤销或证书不存在

9 *result*←"拒绝执行"

10 ELSE

11 通过 $hash_{send}$ 检索 $DB_{证书}$获得 enc_att_{send} 并模拟发布者加密 att_{send} 生成 enc_att_{send}`

12 IF enc_att_{send}` 与 enc_att_{send} 不相同

13 *result*←"拒绝执行"

14 ELSE

15 IF att_{send} 与 $att_{receive}$ 没有继承关系

16 *result* ←"拒绝执行"

17 ELSE

18 通过 att_{send} 生成 $SHA256_{send,}$ 将其与 $hash_{send}$ 作为存储 $SHA256_{receive}$ 的节点的子节点

19 *result* ←"权限添加成功"

20 RETURN *result*

其中模拟加密用于资源实体确认其所添加的权限信息是真实存在且有效的。同时避免权限颁发者存储到证书存储仓库的权限信息与资源实体所添加的权限信息不一致。

3.4 权限验证

权限验证方法分为标准权限验证与快速权限验证两个方法。两个方法的区别在于资源实体是否需要通过查询检索树结构来验证用户所提供的证权限书是否有效。

3.4.1 标准权限验证

标准权限验证方法的主要步骤如图 5 所示。

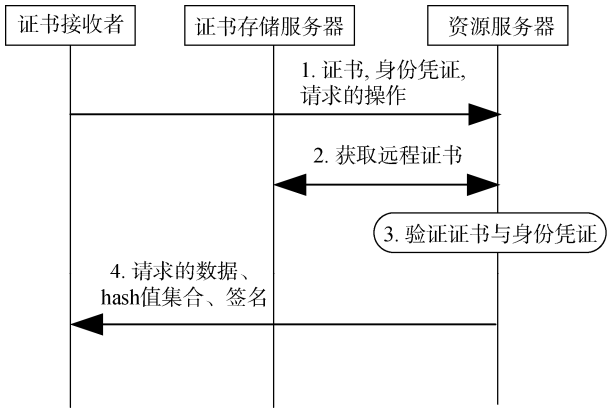


图 5 标准权限验证流程

Figure 5 Standard permission verification

标准权限验证的具体过程如**过程 3** 所示

过程 3. 标准权限验证

输入: $att_{receive}$, *身份凭证*, *timestamp*, $sign_{req}$, 操作请求: *req*

输出: 请求的资源: *res*, set_{hash}, exp_{set}, $sign_{res}$

1 $entity_{req}$ 提交自己的*身份凭证*, $att_{receive}$, *timestamp, req* 以及通过这些信息所生成的 $sign_{req}$ 到 $entity_{res}$

2 $entity_{res}$ 验证*身份凭证*并通过后, 验证 $sign_{req}$ 的生成设备所对应的用户实体与 att_{send} 的接收者是否为同一用户

3 IF 非同一实体或验证失败: 拒绝执行

4 ELSE

5 $entity_{res}$ 通过 $att_{receive}$ 生成 $SHA256_{receive}$ 查找检索树中是否有节点包含该值

6 IF 不存在

7 拒绝执行

8 ELSE

9 $entity_{res}$ 获取从该节点到根节点的路径上的所有用于远程证书检索的哈希值。

10 通过获得的哈希值从 $DB_{证书}$获取所有证书并判断各证书是否被撤销。

11 IF 有证书被撤销或证书不存在

12 拒绝执行

13 ELSE

14 IF $att_{receive}$ 描述的权限满足请求的操作

15 $entity_{res}$ 执行 $entity_{req}$ 所请求的操作, 并通过此次权限验证所涉及的所有远程检索证书用哈希值生成一哈希值有序集合 set_{hash}, $entity_{res}$ 对 set_{hash}、$SHA256_{receive}$、哈希值集合有效期 exp_{set} 进行签名生成 $sign_{res}$

16 ELSE

17 拒绝执行

18 RETURN set_{hash}, $sign_{res}$, exp_{set}, res

set_{hash} 具有时效性, 在如下情况下失效:

(1) 资源实体的签名私钥过期或被盗。此时拒绝 $entity_{req}$ 的快速权限验证并在 $entity_{req}$ 执行完标准权限验证后生成新的 hash 集合以及 $sign_{res}$。

(2) set_{hash} 中的任意一个哈希值所对应的证书因密钥、权限修改等原因被撤销, 证书链失效。

(3) set_{hash} 过期, 需要生成新的哈希值集合。

3.4.2 快速权限验证

为了减少搜索检索树时产生的性能损耗, 当用户非首次访问资源实体时, 其可执行快速权限验证。当 $entity_{req}$ 再次请求资源时, 则发送 set_{hash}、exp_{set}、$sign_{res}$、$att_{receive}$ 到 $entity_{res}$。set_{hash} 为首次访问资源时由资源实体生成的用户所拥有权限的上游权限证书的哈希检索值。$entity_{res}$ 验证签名并通过后, 通过 set_{hash} 检索 $DB_{证书}$并判断各证书是否被撤销。若未被撤销, 且 $att_{receive}$ 所包含的权限允许用户执行操作则返回请求的资源, 反之则拒绝。

3.5 权限撤销

权限撤销与 WAVE 方案类似。证书颁发者发送证书的检索值 $hash_{send}$ 和撤销口令 *revoke* 到证书存储仓库。证书存储仓库通过 $hash_{send}$ 检索到加密的证书后通过撤销口令 *revoke* 生成 $hash_{revoke}$ 并比对该值和证书中存储的撤销口令的哈希值是否相同。如果相同则存储该撤销口令 *revoke*。与 WAVE 不同的是资源实体检索证书时如果同时收到了证书和撤销口令 *revoke*, 且通过撤销口令 *revoke* 生成的 $hash_{revoke}$ 和加密证书存储的撤销口令的哈希值一致, 则认为该证书被撤销。若检索到证书被撤销, 资源实体会通过 $hash_{send}$ 查找检索树, 并将包括存储 $hash_{send}$ 的节点向下的所有节点均标记为不可用。如图 6 所示, 深色节点代表因权限证书被撤销而被禁用的节点。斜线图案节点代表因上游节点(即深色节点)被禁用而被标记为不可用的节点。

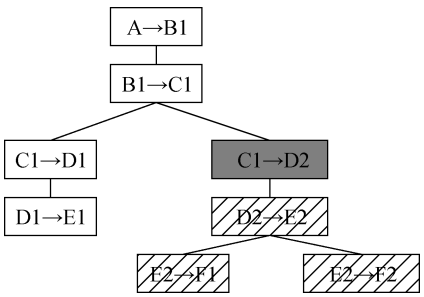


图 6 撤销权限及权限的所有子权限

Figure 6 Revoke permissions and all child permissions

3.6 权限更新

当用户需要更新其他用户的权限证书时, 更新权限的用户需要先发布新的权限证书并获得检索新权限证书的检索用哈希值。在发送撤销口令撤销旧证书的同时还需要提供给证书存储服务器新证书的检索用哈希值、以及由撤销口令和新证书检索哈希值共同生成的 SHA256 值到证书存储服务器。

用户在更新证书存储服务器内的信息的同时, 还需要更新相关资源实体的检索树结构信息。权限信息更新主要包含两个方法: 权限信息替换与权限信息恢复。权限信息替换指用户将自己已发送的权限信息替换为新的权限信息。权限信息恢复指资源实体基于新收到的权限信息与已发送的权限的继承关系恢复已发送的权限信息的可用性。当用户已发送的权限与其接收到的新权限信息依然具有继承关系时, 资源实体可将包括存储已发送的权限信息在内的所有子节点重新标记为可用。若实体已发送的权限与其接收到的新权限信息不具有继承关系时, 实体需要基于新收到的权限信息执行权限信息替换操作。如图 7 所示, 用户 D2 给 E2 发送了新的权限信息 $update\_att_{D2\_TO\_E2}$(U_D2→E2), 即渐变深色节点。该权限信息向下的所有权限信息均被暂时标记为不可用, 即斜线图案的节点。假设权限信息 $att_{E2\_TO\_F1}$(E2→F1)与 $update\_att_{D2\_TO\_E2}$ 依然具有继承关系, 资源实体便可恢复包括 $att_{E2\_TO\_F1}$ 在内向下的所有权限信息的可用性。如图 8 节点颜色恢复为白

色的节点所示。假设权限信息 $att_{E2\_TO\_F2}$(E2→F2)与 $update\_att_{D2\_TO\_E2}$ 不存在继承关系, E2 需要基于 $update\_att_{D2\_TO\_E2}$ 更新旧的权限信息为 $update\_att_{E2\_TO\_F2}$(U_E2→F2), 如图 9 所示。$update\_att_{E2\_TO\_F2}$ 向下的权限信息需要根据继承关系来执行权限信息恢复或权限信息替换。

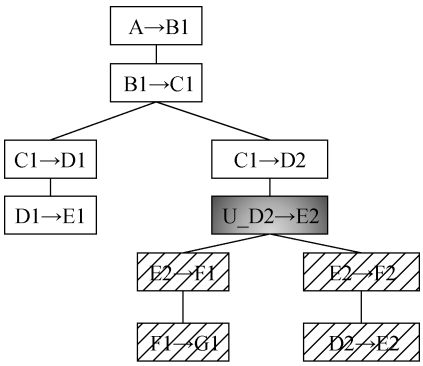


图 7 单节点被更新, 其下游节点均标记为不可用

Figure 7 A single node is updated and its downstream nodes are marked as unavailable

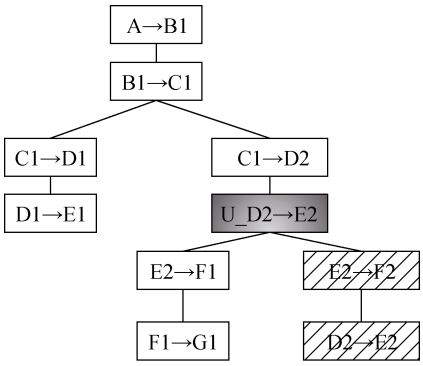


图 8 未受影响的权限可以被恢复

Figure 8 Unaffected permissions can be restored

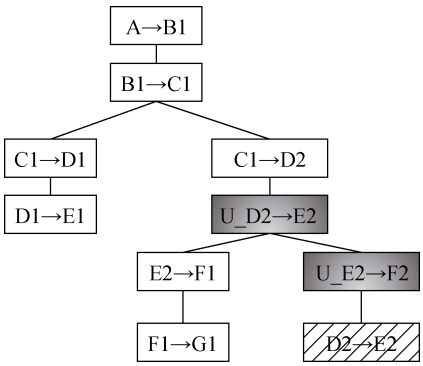


图 9 受到影响的权限得到更新, 该权限的下游权限未被恢复

Figure 9 The affected permission is updated, and the downstream permission of the permission is not restored

4 安全性分析

4.1 权限颁发安全性分析

当证书颁发用户向证书接收用户颁发权限证书时, 在证书存储的安全性上存储在证书存储服务器的证书由证书接收用户的加密公钥加密, 而解密用的私钥可由 TPM 保护, 因此攻击者只有盗取用户设备并对其进行必要的物理攻击才能获得解密私钥。作为公司或组织的设备管理员, 在统一部署用户设备时, 管理员可基于 TPM 搭建可信软件平台, 以防止 TPM 中的私钥由于恶意程序对系统内程序的篡改而造成密钥的泄露或被用于签发、解密数据。用户将证书存储到证书存储服务器时, 其还需要将权限信息发送到与证书所描述权限相关的资源实体以用于资源实体创建检索树。在创建过程中, 资源实体需要验证用户是否有权颁发权限。验证方式分为两种: 一种是该权限证书由资源实体的管理员直接颁发, 另一种基于颁发证书的用户其拥有的证书和资源实体创建的检索树。对于第一种验证方式, 管理员颁发的证书同样需要存储到证书存储服务器用于证书的持久性存储和对管理员行为进行审计。因此管理员自身无法在不被察觉的情况下进行恶意证书颁发, 而证书的颁发依赖设备的 TPM 模块, 因此除非对设备进行物理攻击, 否则不会产生恶意且具有合法性的权限证书。对于第二种验证方式, 颁发实体在颁发证书时需要提供自己拥有的证书, 该证书需要资源实体验证证书存储服务器以验证证书的有效性并通过章节三中描述的验证检索树验证该权限证书的来源及有效性。最后资源实体可验证用户拥有的证书与将要发送的证书是否具有继承关系, 因此证书颁发用户不能颁发其没有的权限和不能传递的权限。

4.2 权限验证安全性分析

在权限验证方面, 相较于其他授权协议, 资源请求实体只发送自己所拥有的权限证书及身份凭证信息到资源实体, 而不发送授权路径(即实体拥有的权限的来源), 权限的授权路径已经在各实体颁发权限时由资源实体生成并签名, 因此攻击者无法通过伪造或篡改该授权路径以获取权限, 只能通过攻击目标设备获取目标设备所拥有的权限。而目标设备包含 TPM 模块, 降低了设备受到攻击并造成私钥泄露或被恶意使用的可能性。

4.3 检索树结构安全性分析

检索树结构只包含用于远程检索权限证书的哈希值和验证证书完整性的 SHA256 值。通过哈希值

只能获取加密后的证书, 而存储在远程服务器的证书的明文部分不包含接收者信息、发送者信息以及权限信息。因此攻击者即使获取到检索树也无法获取各实体的授权路径以及权限的详细信息。

4.4 权限信息与设备的绑定

在本文的使用场景中用户的用户信息与该用户的设备的 TPM 不可迁移密钥相绑定。权限证书中包含权限的颁发者与权限的接收者的唯一标识, 因此资源实体可通过 TPM 模块生成的签名与权限信息中的颁发者与接收的唯一标识一起判断权限信息来源的有效性用以决定是否添加新的权限信息或者执行用户所请求的操作。

5 性能分析

本文对 ITTDAF 的检索树创建所需时间, 权限证书创建所需时间、权限证书验证所需时间、权限证书传输所需数据大小进行了测试, 并与 WAVE 协议进行了对比。与其他支持传递授权的访问控制方案相比, WAVE 包含如下两个主要特性且这些特性与本文所提出的方案在所欲达到的目标上相一致:

(1) 不依赖中心可信性: 用户不依赖公钥基础设施生成用于权限加密与权限验证的密钥, 而是通过用户自己生成密钥并提交公钥信息到数据库。

(2) 相较其他访问控制方案, WAVE 在传输权限和存储权限时均可保证权限的机密性, 使权限信息不会被非相关用户得知。

(3) 方案已开源并在现实环境中测试使用长达 2 年。因此相较其他访问控制方案, 本论文选择 WAVE 方案作为比对对象并认为与该方案相比较所获得的实验数据更加具有参考价值。

测试环境方面, 使用处理器为 Intel 10870HQ, 操作系统为 Windows 10, 包含 64GB 内存的主机作为测试环境。在该主机上同时运行 3 台虚拟机, 每台虚拟机分配 8 个处理核心和 16GB 内存。3 台虚拟机分别用于表示用户设备、证书存储服务器与资源服务器。仿真环境均采用 JAVA 编程语言进行编写, 以保证测试数据不会受到语言特性的影响。

由于检索树本质上是一个 N 叉树结构, 但该树结构有多少层, 每层有多少节点无法预先得知。因此本文采用二叉树来表示 N 叉树, 如图 10 所示。

检索树生成时间如图 11 所示。当每层每个节点的子节点数目一定的情况下, 随着层数的增加, 虽然树的生成时间会以较快的速度增长, 但检索树结构仅在有新的权限被插入或者旧的权限需要修改时才会被访问, 在通常权限验证, 即 ITTDAF 的快速权限验证操作中不会涉及。因此检索树生成时间较长对于 ITTDAF 的运行效率影响有限。在后续测试中, 实体在通过 ITTDAF 进行权限验证时均采用快速权限验证。

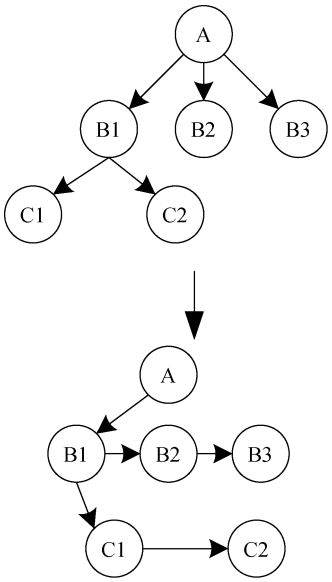


图 10 N 叉树转换为 2 叉树

Figure 10 Convert n-ary tree to 2-ary tree

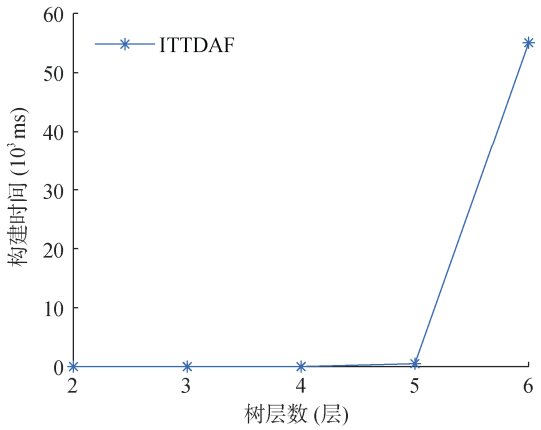


图 11 不同层数对检索树生成时间的影响

Figure 11 Effects of different layers on the generation time of index tree

ITTDAF 的证书创建时间、证书体积大小、单个证书验证所需时间与 WAVE 的对比如表 2 所示。由于 ITTDAF 的权限证书创建不需要多层加密, ITTDAF 单个证书需要加密的数据也少于 WAVE, 因此在证书创建效率和证书数据体积上 ITTDAF 优于 WAVE。在验证权限时, 由于 ITTDAF 只需要验证实体所拥有的证书的有效性和该证书的父权限证书是否被撤销, 而不需要解密并验证所有父权限证书, 因此权限验证所需时间 ITTDAF 也优于 WAVE。更

短的权限验证时间允许资源实体在相同时间内验证更多用户的权限验证请求, 提高了并发性。而更短的证书创建时间和更小的权限证书体积使得用户设备不需要拥有过高的性能即可执行该框架的权限管理操作。因此相较于 WAVE, ITTDAF 拥有更高的可用性。

表 2　ITTDAF 与 WAVE 性能对比

Table 2　Performance comparison between ITTDAF and WAVE

测试项目	协议名称	
	WAVE	ITTDAF
创建证书耗时/ms	172.8	16.7
证书体积/byte	11766.78	6569.5
验证所需时间/ms	13.0	6.2

证书链长度(即实体所拥有的权限到根权限所涉及的权限证书数量)对授权时间的影响如图 12 所示。本文令用户设备通过访问资源实体 1000 次测得图 12 中数据。因为是单一用户重复执行权限验证, 所以单个证书验证所需的平均时间略低于表 2 中的证书验证时间。由于 ITTDAF 不需要对上游证书进行解密, 仅需要验证上游证书是否被撤销。因此当证书链长度一定时, 相较于 WAVE, ITTDAF 允许资源实体在更短的时间内完成权限验证操作。

较长的证书链可实行更细粒度的权限管理与维护, 如用户的电脑设备拥有最高权限, 该设备管理的智能家居网关拥有中级权限, 连接该网关的智能家居设备拥有低级权限。在证书链长度一定的情况下, 更短的权限验证时间允许更多低性能的设备可以使用 ITTDAF 实现访问控制。而资源实体也不需要花费过多的时间在权限验证上, 拥有更多自由的时间处理其他操作请求。

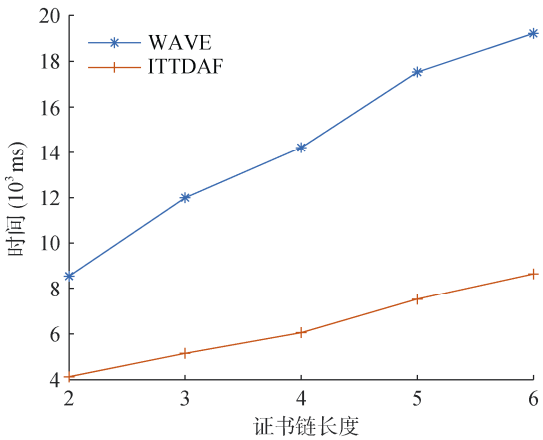


图 12　证书链长度对验证时间的影响

Figure 12　Effect of certificate chain length on verification time

证书链长度对数据传输量的影响如图 13 所示。本文令一个实体通过访问资源实体 1000 次测得图 13 中数据。由于 ITTDAF 的权限证书不包含上游权限信息, 并且 ITTDAF 只需要验证父权限是否被撤销而不需要接收并验证完整的父权限证书信息, 因此在相同测试条件下 ITTDAF 的权限传输数据占用低于 WAVE 的数据占用。

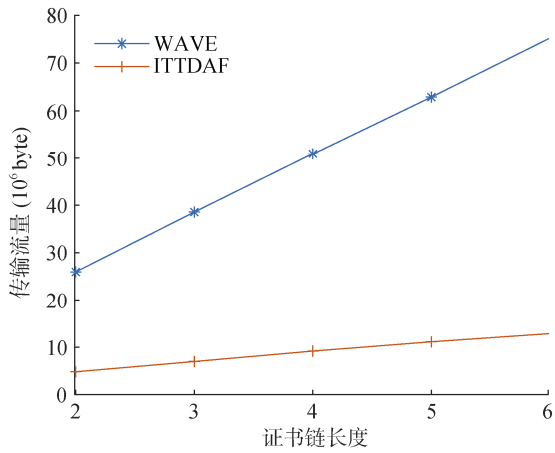


图 13　证书链长度对传输数据量的影响

Figure 13　Effect of certificate chain length on the amount of data transmitted

更低的传输数据量需求允许 ITTDAF 在低网络带宽和较差的网络环境下使用。在网络流量付费的情况下节约用户使用成本。如大量的物联网信息采集节点如果需要访问控制的支持, 相较于 WAVE, ITTDAF 能提供更好的性能与更低的流量成本。而网络带宽一定时允许资源实体可接收到并处理更多的访问请求。

6　总结与展望

本文提出了基于索引树和可信平台模块的去中心化授权框架 ITTDAF。通过实验和安全性分析, 其在支持可传递授权的情况下具有不需要传递父权限信息的特性, 拥有更低的密钥维护难度。降低了验证权限时的时间消耗与流量消耗。同时增强了对用户的权限的机密性的保护, 避免了用户的密钥泄露或权限信息泄露时对其他用户权限机密性的破坏。通过 TPM 模块的引入, ITTDAF 保证了权限与设备的绑定, 使得用户的权限能且只能在用户设备上得到执行。但是 ITTDAF 局限于检索树性能上的不足, 该部分将是本文在未来改进的方向。

参考文献

[1] Nakanishi R, Zhang Y Y, Sasabe M, et al. IOTA-Based Access Control Framework for the Internet of Things[C]. *2020 2nd Conference on Blockchain Research & Applications for Innovative Networks and Services* (*BRAINS*), 2020: 87-95.

[2] Siris V A, Dimopoulos D, Fotiou N, et al. Interledger Smart Contracts for Decentralized Authorization to Constrained Things[C]. *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops*, 2019: 336-341.

[3] Dramé-Maigné S, Laurent M, Castillo L. Distributed Access Control Solution for the IoT Based on Multi-Endorsed Attributes and Smart Contracts[C]. *2019 15th International Wireless Communications & Mobile Computing Conference*, 2019: 1582-1587.

[4] Andersen M P, Kolb J, Chen K F, et al. Democratizing Authority in the Built Environment[J]. *ACM Transactions on Sensor Networks*, 2018, 14(3/4): 1-26.

[5] Nakanishi R, Zhang Y Y, Sasabe M, et al. Combining IOTA and Attribute-Based Encryption for Access Control in the Internet of Things[J]. *Sensors*, 2021, 21(15): 5053.

[6] Saini A, Zhu Q Y, Singh N, et al. A Smart-Contract-Based Access Control Framework for Cloud Smart Healthcare System[J]. *IEEE Internet of Things Journal*, 2021, 8(7): 5914-5925.

[7] Tang B, Kang H J, Fan J W, et al. IoT Passport: A Blockchain-Based Trust Framework for Collaborative Internet-of-Things[C]. *The 24th ACM Symposium on Access Control Models and Technologies*, 2019: 83-92.

[8] Ullah I, de Roode G, Meratnia N, et al. Threat Modeling-how to Visualize Attacks on IOTA? [J]. *Sensors* (*Basel*, *Switzerland*), 2021, 21(5): 1834.

[9] Ouaddah A, Bellaj B. FairAccess2.0: A Smart Contract-Based Authorisation Framework for Enabling Granular Access Control in IoT[J]. *International Journal of Information and Computer Security*, 2021, 15(1): 18.

[10] Ghaffari F, Bertin E, Crespi N, et al. A Novel Access Control Method via Smart Contracts for Internet-Based Service Provisioning[J]. *IEEE Access*, 2021, 9: 81253-81273.

[11] Wang S P, Wang X, Zhang Y L. A Secure Cloud Storage Framework with Access Control Based on Blockchain[J]. *IEEE Access*, 2019, 7: 112713-112725.

[12] Zhang Y, Li B, Liu B, et al. An Attribute-Based Collaborative Access Control Scheme Using Blockchain for IoT Devices[J]. *Electronics*, 2020, 9(2): 285.

[13] Xu R H, Chen Y, Blasch E, et al. BlendCAC: A BLockchain-Enabled Decentralized Capability-Based Access Control for IoTs[C]. *2018 IEEE International Conference on Internet of Things and IEEE Green Computing and Communications and IEEE Cyber, Physical and Social Computing and IEEE Smart Data*, 2018: 1027-1034.

[14] Nakamura Y, Zhang Y Y, Sasabe M, et al. Exploiting Smart Contracts for Capability-Based Access Control in the Internet of Things[J]. *Sensors* (*Basel*, *Switzerland*), 2020, 20(6): 1793.

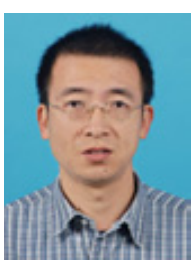
[15] Pinjala S K, Sivalingam K M. DCACI: A Decentralized Lightweight Capability Based Access Control Framework Using IOTA for Internet of Things[C]. *2019 IEEE 5th World Forum on Internet of Things*, 2019: 13-18.

[16] Andersen M P, Kumar S, AbdelBaky M, et al. {WAVE}: A decentralized authorization framework with transitive delegation[C]. *28th {USENIX} Security Symposium*. 2019: 1375-1392.

[17] Rivest R L, Shamir A, Adleman L. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems[J]. *Communications of the ACM*, 1978, 21(2): 120-126.

[18] Li X Y. Theory and Technology of Trusted Computing[M]. Beijing: Beijing University of Posts and Telecommunications Press, 2018: 10-17.
(李小勇. 可信计算理论与技术[M]. 北京: 北京邮电大学出版社, 2018: 10-17.)

罗期丰 于 2019 年在北京邮电大学物联网工程专业获得学士学位。现在北京邮电大学网络空间安全专业攻读硕士学位。研究领域为网络安全, 研究兴趣包括访问控制、发布订阅系统等。Email: luoqifeng@bupt.edu.cn



石瑞生 博士, 北京邮电大学副教授、硕士生导师。研究领域为网络安全、区块链安全监管、物联网、服务计算等。Email: shiruisheng@bupt.edu.cn