北京大学

硕士研究生学位论文

题目：基于区块链的 PKI 实体鉴权方案设计与实现

姓　　名：李安然

学　　号：1501214389

院　　系：信息科学技术学院

专　　业：计算机软件与理论

研究方向：网络与信息安全

导　　师：关志副研究员

X 年 X 月

版权声明

摘要

pkuthss 文档模版最常见问题：

`\cite`、`\parencite` 和 `\supercite` 三个命令分别产生未格式化的、带方括号的和上标且带方括号的引用标记：1，[4]、[1,4]。

若要避免章末空白页，请在调用 pkuthss 文档类时加入 openany 选项。

如果编译时不出参考文献，请参考 `texdoc pkuthss`“问题及其解决”一章“其它可能存在的问题”一节中关于 biber 的说明。

关键词：其一，其二

Test Document

Anran Li (Software and Theory of Computer)
Directed by Prof. ZhiĠuan

ABSTRACT

Test of the English abstract.

KEYWORDS: First, Second

目录

序言

pkuthss 文档模版最常见问题：

\cite、\parencite 和 \supercite 三个命令分别产生未格式化的、带方括号的和上标且带方括号的引用标记：1，[4]、[1,4]。

若要避免章末空白页，请在调用 pkuthss 文档类时加入 openany 选项。

如果编译时不出参考文献，请参考 texdoc pkuthss“问题及其解决”一章“其它可能存在的问题”一节中关于 biber 的说明。

第一章　绪论

1.1　背景知识和研究意义

1.2　国内外研究现状

1.2.1　区块链技术的发展

1.2.2　PKI 的研究现状

1.3　本文研究工作和章节安排

第二章　区块链技术

2.1　比特币和以太坊

2.2　共识机制

2.3　merkle 树

2.4　智能合约

pkuthss 文档模版最常见问题：

`\cite`、`\parencite` 和 `\supercite` 三个命令分别产生未格式化的、带方括号的和上标且带方括号的引用标记：1，[4]、[1,4]。

若要避免章末空白页，请在调用 pkuthss 文档类时加入 `openany` 选项。

如果编译时不出参考文献，请参考 `texdoc pkuthss`“问题及其解决”一章“其它可能存在的问题”一节中关于 biber 的说明。

第三章 PKI

本章将对 PKI 系统进行详细介绍，首先从系统架构对 PKI 进行解析，阐述系统中包含的组成部分和各自的功能；其后将对证书进行介绍，包括证书的结构以及证书的生命周期；最后对本系统存在的问题以及相应的解决方案进行简要叙述。

3.1 PKI 系统

公钥基础设施 PKI（Public Key Infrastructure) 作为一种遵循标准的密码管理平台，其可以为网络应用提供密钥和证书管理服务，使用户可以在多种应用环境下方便的使用加密和数字签名技术，从而保证数据的机密性、完整性、有效性和抗抵赖性。

本小节将从框架和其包含的相关组件对 PKI 进行介绍。

3.1.1 PKI 基本框架

PKI 框架中包括安全和操作策略，安全服务以及支持公钥密钥和证书管理的交互式协议。公钥和对应证书的生成、分发和管理将通过授权机构（CAs）、注册机构（RAs）和目录服务来完成[3]，它们将会建立等级信任或者说信任链。以上提到 CAs、RAs 和目录服务可以将数字证书用于鉴定不同实体的身份，而 PKI 拥有如此架构的目的是为了能支持并完成数据、凭证在各种不安全环境下的安全交换。

3.1.2 PKI 的系统构成

在 PKI 系统可能会包含一下组成部分：

- 注册机构 (RA)

  RA 的作用是执行身份验证和处理新的数字证书请求、更新数字证书请求和吊销数字证书请求。
- 授权机构 (CA)

  CA 将创建和发布数字证书以及证书吊销列表（CRLs)，其颁发的数字证书将对主体名称（如用户表示）和绑定的公钥进行签名。
- 验证结构 (VA)

  VA 是一个 PKI 的管理实体，可以用来检查数字证书的有效性。当证书的签发者和证书的状态管理服务有不同的服务提供者提供时，将使用到 VA。
- 用户

PKI 面对的用户是证书持有者或者密钥持有者，通过遵从认证运作规范（CPS）和证书策略（CP) 获得证书，完成证书和密钥对的绑定。PKI 系统中的用户可以是个体、组织或者非个人实体，有责任保存好自己的私钥不被泄露。

- 依赖方

  依赖方在 PKI 系统中接收、验证和接受数字证书。

- 证书策略 (CP)

  证书策略是一组安全规则要求，适用于一类应用系统的共同安全需求。

- 认证运作规范 (CPS)

  CPS 描述了 CA 提供数字证书服务的规则和处理方式，其中可能会包括提供服务描述，证书生命周期的管理细则、业务信息、法律义务和金融责任等。

图3.1将给出以上所提到组件之间的关系，图中的剪头将表示数字证书和证书状态信息的传递。

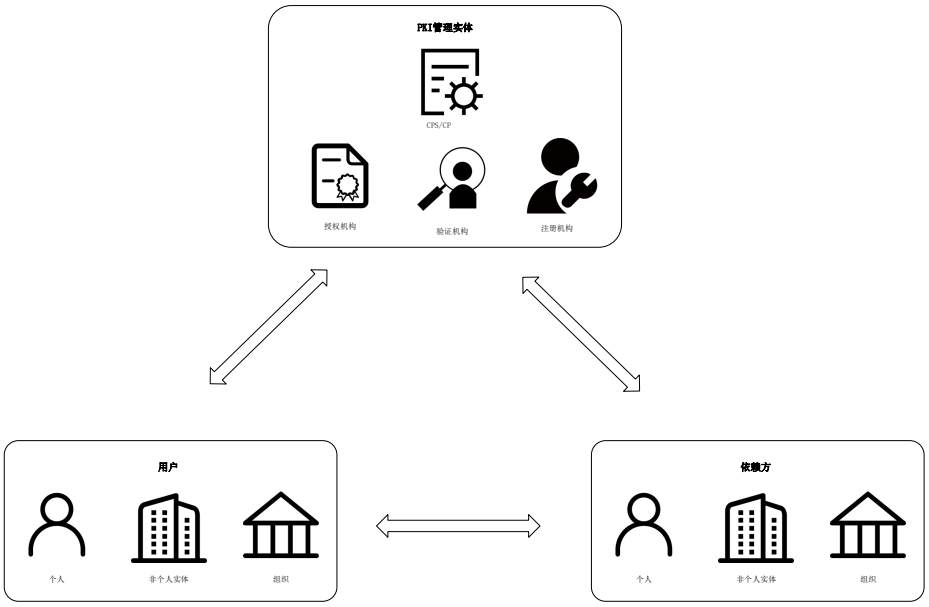

图 3.1　PKI 组成部分

3.2　证书

3.2.1　证书的结构

描述证书的结构。

3.2.2　证书的生命周期

证书作为包含公钥、数字签名以及一些其它附带信息的数字文档，在 PKI 系统中充当着公钥交换、存储和使用的介质。了解证书的申请、签发和使用，可以明白 PKI 系统的是如何运作，并从中发现可能存在的问题。

证书的声明周期从用户提交准备的证书签发请求 (CSR) 并提交给其选择的 CA 开始。CSR 中包含了用户的公钥和相应的信息，并通过签名的方式表明对相应私钥的所有权。同时，CSR 可以携带额外的信息元，但在实际使用过程中并没有全部使用。CA 可以对 CSR 中的内容进行重写，放置一些其它的信息在证书中。

其后 CA 通过遵循验证流程，对用户进行身份验证。待成功完成验证之后，CA 将签发证书，同时提供验证至根证书的所有中间证书。

得到证书后，用户就可在证书过期之前使用证书。如果证书对应的私钥泄露，证书将可以被吊销，该过程和证书签发的过程类似。

对于 Internet PKI 系统而言，根据以上证书流转流程，证书生命周期如图3.2所示.

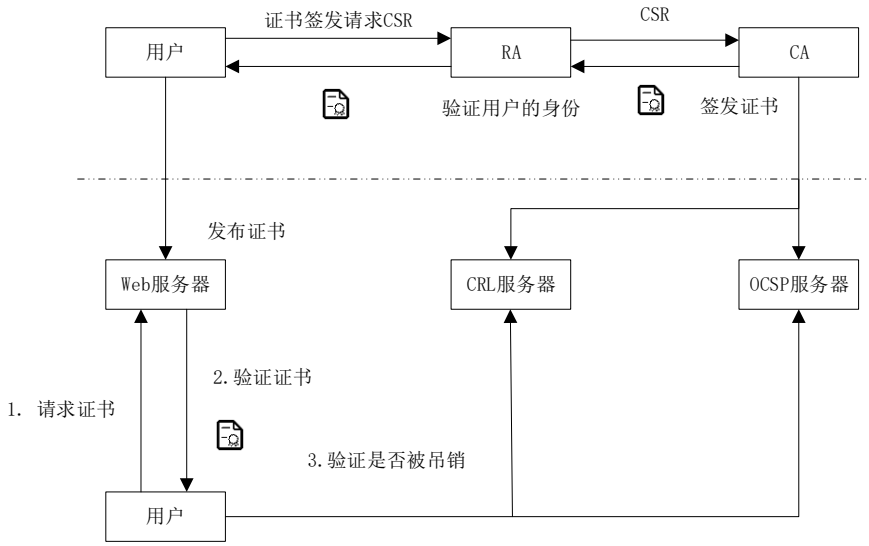

图 3.2　证书的生命周期

3.3　系统中存在的问题

从安全的角度来看现有 Internet PKI 系统，其中存在着大大小小的各种问题，在本小节中，将对这些存在的问题进行简要概述。

当 Web 安全在 1995 年刚开始被谈及时，当时的互联网和现在是具有很大差异的，

它的重要性并没有现在这么强。随着互联网的蓬勃发展，加密安全成为了商业中必备的一部分，关乎这一个企业的生死。现有的 PKI 系统和当初设计的目的是一致的，即为上电子商务操作提供足够的安全保障，更进一步可以说 PKI 系统希望提供商务安全。这种安全可以通较少的资金、更快的网页获取速度、可接受的不安全操作和不对用户做过多的限制来完成。本系统由 CAs、寻利的商业实体和希望扩大市场份额的游览器供应商一起来组成。

CA 在本系统中充当着信任的基础，其每年签发了数百万计的证书，这些证书将在互联网中流通并使用。虽然证书的安全性并没有想象中那么好，但是这个系统任然在正常的运转下去。同时，在那么不完美的情况下，用户还需要对申请的证书进行付费，然而并不是所有用户都愿意为之付出代价的，很简单的一个原因就是他们在付费之后希望得到完美的安全性保障。

在该系统中，存在着一下几个方面的问题：

- 域名权利过弱

  在 PKI 系统中最大的问题是任意 CA 在未经域名同意的情况下就可以对其签发证书，导致这个问题的主要原因是没有相关的技术策略去避免 CA 的疏漏和安全隐患，而该系统需要对 CA 给予相当高的信任。当 CA 数量比较少的时候，这个问题并没有那么严重，但是下周乃有数以百计的 CA 存在，很难保证每一个都不会存在行为过失或者不当的安全配置。一个系统的安全性取决于该系统最薄弱的环节，而 PKI 系统中存在着各种潜在的薄弱环节。所有的 CA 都需要被审计，但是审计的质量缺是不确定的。例如在 2011 年 DigiNotar 由于自身安全性问题就被黑客攻陷，最后导致自身倒闭。

  同时，另外一个存在的问题是 CA 是否可以被信任，它们能否在不需要监督的情况下为了公众的利益去做好自己的本质工作。这些被信任的 CA 可能在面对商业利益的时候放弃我们需要的安全。例如在 2012 年 Trustwave 承认其签发了低级别的假冒证书用于流量检查。虽然 Trustwave 是唯一公开承认自己做过类似事情的 CA，大家相信这样的事情肯定大量存在。

  政府也可能会滥用 PKI 系统签发虚假证书，完成对任意域名的假冒。公众无法确保 CA 不会作为证书的前线，即使不是也无法保证这些 CA 不会被强迫的签发虚假证书。

- 没有信任灵活度

  另外一个重要的问题是本系统缺乏信任的灵活度。依赖方将会存储一些列信任的根证书，一个 CA 只存在信任与否，并不存在中间地带。理论上，依赖方可以移除任何存储的 CA，实际上这种情况只会在 CA 很小或者其已经被攻陷的时候发

生。一旦一个 CA 签发了大量的证书，其将由于自身的大体量被不会被撤销。

一些小的改善任然在被提出，例如对具有过失行为的 CA 不在信任打，是其之前的证书任然可以被使用。

- 域名验证过于简单

  DV 证书的签发是基于域名的 WHOIS 协议查询域名拥有者信息来完成的，也就是说大部分验证是通过邮件来完成的，而其本身的安全性就存在问题。如果域名被黑掉或者相应的邮箱密码被获取，那么就可以得到给域名的 DV 证书。同时通过拦截 CA 端验证信息也可以对发起攻击。

- 吊销不工作

  在一般情况下，对吊销证书的检查并没有那么严格，大多数情况都不能正常的工作。在 2011 年中有很多这样的例子，依赖方不得不将被泄露的证书通过特殊通信方式下载并存储在黑名单中，来保证吊销查询是可靠的。

  这样做的原因主要包括两个，首先将吊销信息发送各个系统需要一定的延时，在基准规则中允许 CRL 和 OCSP 的信息在 10 天是有效的，也就是说至少需要 10 天才能保证吊销信息被完全扩散出去；其次软失败机制在所有的游览器中被使用，当他们在查询吊销信息时会忽略掉所有失败的情况，一个主动的网络攻击者将可以轻易的拦截 OCSP 的请求，保证他们使用虚假的证书可以完美的被使用。

  由于以上的原因，Chrome 开发者取消去证书吊销的检查，除非是 EV 类型的证书。对于重要的证书，例如中间证书，其间依赖于 CRL 信息的吊销通道查询相关内容。一种可能的解决方案是使用 Must-stale 的方案来保证证书的有效性。

3.4 已有的改善措施

为了解决 X.509 PKI 中的安全问题并削弱对 CAs 的信任，一系列的方案被提出来，本小节将对这些方案进行分类并简要叙述。

根据 PKI 中存在的主要三方实体域名、CA 和客户端，可以对现有的方案进行分类，如图3.3所示。

以客户端为中心的方案

这类方案希望在客户端接受证书之前，可以更加准确的验证证书的有效性。Policy engine 的方案允许客户端在本地制定信任决策，比如支持的密码算法，证书的一致性性等。

有一些其它的方案希望构建一个公共的域名证书存储厂库，使得客户端可以在接受到域名证书之后与厂库中的证书进行对比，确定证书的正确性，Perspectives 和 Con-

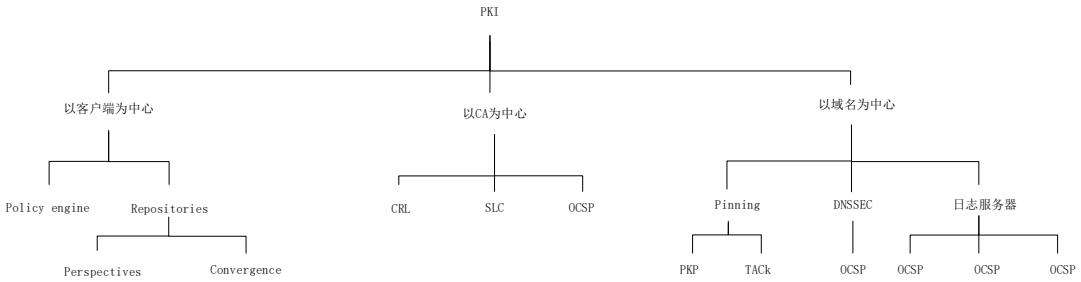

图 3.3　PKI 方案分类

vergence 就输入这类方案。

以客户端为中心的方案并不需要对服务端进行的改动，但是其需要客户端建立额外的连接去查询资源厂库，这将在一定程度上降低建立 HTTPS 的速度。

以 CA 为中心的方案

X.509 PKI 包含了证书吊销列表（CRL) 标准，希望放置客户端与一个使用已经被吊销证书的域名之间发起建立 TLS 连接。但是这需要保证客户端需要能随时的访问到 CRLs。为了进一步解决在线验证的问题，在线证书状态协议（OCSP）允许客户端通过 CA 的 OCSP 服务器去检查域名证书的状态。但是 OCSP 也拥有如安全、隐私、性能方案的顾虑。另外一个解决方案是短期证书（SLC），该方案希望签发短声明周期的证书，让域名定期的更换证书。SLC 希望提带来和 OCSP 类似的好处，但是不需要在线验证。以证书为中心的方案严重依赖于游览器去检测和拉黑被攻陷 CA 签发的证书，这是这类方案的最大缺陷。

以域名为中心的方案

第三类方案允许域名拥有者积极控制和保护他们的证书而不被 CA 端的潜在问题所威胁。这些方案又可以分为以下三类：pining、DNSSEC、日志服务器。

1. Pinning-based（公钥固定）

如 Public Key Pining(PKP) 和 Trust Assertions for Certificate Keys(TACK) 等 Pinning 的方案希望域名宣称自己使用的密钥，以便客户端收到的密钥是否正确。但是这类方法有相应的安全缺陷，比如在第一次访问域名的时候无法给予安全保护。

2. DNSSEC-based

这类方案的全称叫基于 DNS 的命名实体鉴权（DANE），希望域名的所有者可以在 DNSSEC 实体上方式证书相关的特殊声明，例如可以为其签发证书的 CA 名单、声

明接受的证书或者是声明验证证书有效性的节点，但是 DANE 的安全性严重的依赖于 DNS 操作的安全性行。

3. 日志服务器

另外一种使用比较多的方法是通过日志服务器的方案来记录 CA 的行为，为域名拥有者提供公共的、可审计的 CA 行为日志，监督 CA 的行为。例如 Sovereign Keys (SK) 要求域名拥有者生成一个主密钥对来完成对 TLS 公钥的签名，并且将其主公钥以只读和只可追加的方式记录在时间服务器上。不行的是，SK 需要客户端查询服务器，增加了延迟并牺牲了隐私。

证书透明（CT）的方案允许每个域名将自己的证书注册记录到一个公共日志服务器上，该服务器使用默克尔哈希树的结构来存储这些证书，并保证只能追加的性质。该服务器将返回一个不可否认的证书审计证明给域名，域名使用证书和该证明给客户端来完成 TLS 连接的建立。但是，本方案并不能防止当一个攻击者攻破了 CA 并创建并注册一个虚假的证书的情况，CT 并不能阻止客户端接受这类证书。由于证书透明在设计过程中并没有强调证书的吊销，相应的证书吊销透明（RT）也被提出。为了提高 CT 对证书吊销的效率，证书签发吊销透明（CIRT）方案被提出。

另外一种基于日志服务器的方案称作可审计密钥设施（AKI），该方案希望保护域名和客户端遭受到单点失败攻击，比如某个 CA 的根密钥被泄露，通过制衡该系统中各个实体，AKI 在保持高效处理证书操作的情况下，成功的将信任分散到了多个实体，并可以检测出实体的恶意行为。

第四章　基于区块链的命名实体鉴权方案设计与实现

本章节将阐述基于区块链的命名实体鉴权方案，首先从系统设计的整体架构出发，描述本系统中包含的系统角色和架构；然后对鉴权方案进行详细阐述，并理论论证鉴权方案的有效性和安全性；最后对系统实现的各个模块进行描述。

4.1　系统设计

本部分将从涉及的角色和结构对本系统的整体架构进行描述。

4.1.1　系统角色

针对证书申请和使用流程，在 PKI 系统中涉及的主要角色包含以下三类：

- 授权机构（CA）：授权机构是证书的签发者，在 PKI 系统充当证书签发和管理的角色。
- 证书申请者：证书申请者是 PKI 系统中用户，作为证书的持有者，在初始过程中需要向 CA 提出证书申请，在完成身份验证之后即可获得相应的证书，并在使用过程中提供证书以证明自己的身份，在本场景中，证书的申请者指域名或者站点。
- 证书验证者：证书验证者在 PKI 系统中是依赖方，是证书的受用者，在通信过程中需要认证对方身份的时候，获得证书并完成有效性验证。

不同的角色在 PKI 证书的流转过程中对证书的控制权不尽相同，授权机构作为证书的签发方，对其是否签发证书具有绝对的主动权，甚至可以在未经证书申请者同意的情况下签发恶意证书；证书申请者则只能发起证书申请，不能对证书是否被签发起到决定权；证书验证者则只能验证证书是否合法，对于恶意 CA 私自签发的虚假证书，并没有辨别真伪的能力。

可以看出，在传统的 PKI 系统中，证书申请者和证书授权者之间的地位是不对等的，这是由于证书授权方对证书的拥有绝对管理权，所以在 PKI 系统中，需要所有的角色对中心化的 CA 绝对性信任。为了使得证书申请者和证书授权者之间的权利更加均衡，需要提高证书申请方对证书签发的控制权，使用本文中涉及的命名实体鉴权方案，保证未被允许的证书签发签发机构无法对证书进行私自签发。

在本方案中，以上提到的三类角色具有以下特性：

- 授权机构（CA）：在本方案中，其角色和功能保持不变，负责对发起证书申请的实体进行证书颁发。

- 证书申请者：作为 PKI 系统中的用户，在本方案中其将利用区块链完成身份的认证，并将自己信任的 CA 列表发布在区块链上。
- 证书验证者：作为 PKI 系统中的依赖方，在本方案中验证者可以通过区块链查询通信实体的信任 CA 列表，保证接受到的证书是通信实体信任 CA 签发的。
- 验证节点：在本方案中，验证节点作为证书申请者的身份验证者（域名），通过这些节点完成对区块链上证书申请者的身份认证。

4.1.2 系统架构

本系统依托于现有的 PKI 体系，并将区块链作为存储和身份鉴权平台，在其上进行证书申请者信任 CA 列表的存储，系统中的各个角色都要通过本系统中的区块链进行数据交流。更为重要的是，域名站点需要在此基础上进行身份认证，作为信任建立的根基。系统的总体设计如图4.1所示。

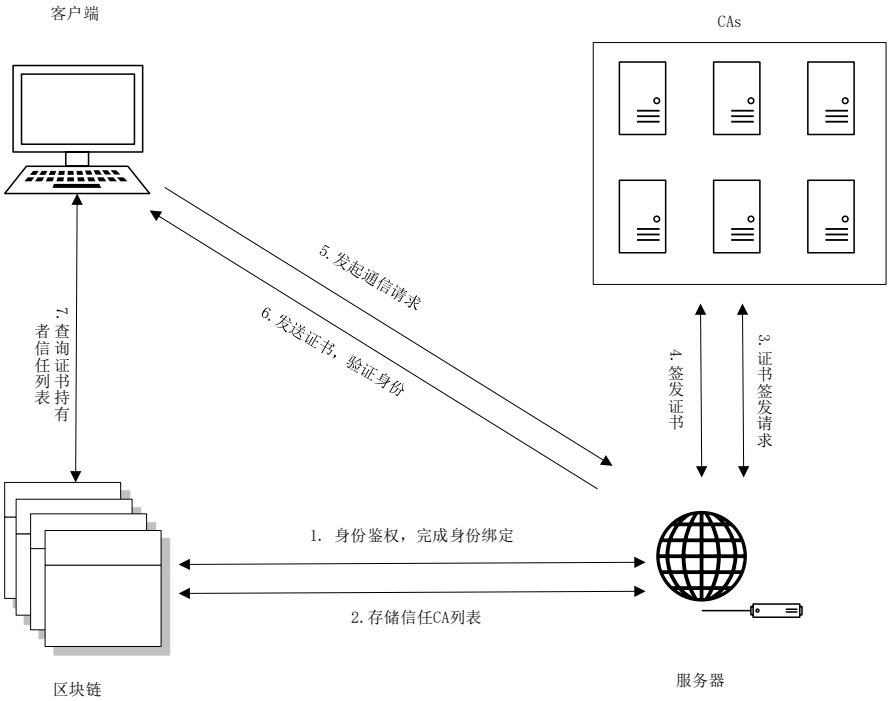

图 4.1 总体架构

在本系统中，域名站点作为证书申请者，向区块链发起身份认证请求，当交易被记录到区块链上之后，域名站点根据请求交易提供验证信息，供验证节点对其身份进行验证；完成验证之后，域名即可将自己信任的 CA 列表发布到区块链上，并向自己信任的 CA 进行证书申请；完成申请后，即可将证书用在与用户通信的过程中，进行身份

证明，此时证书验证者将通过区块链进行信任列表查询，判断该证书是否是被授权签发的。

4.2　身份绑定方案

在本方案中，域名需要将自己信任的 CA 列表记录到区块链上，但是区块链具有去中心化和匿名的特性，域名直接将自己的信任 CA 放置到区块链上是不具有可信性度，也就是说任何人都可以对任何域名进行声明。因此，之前需要完成区块链地址（或者公钥）和域名本身的绑定，实现地址与域名之间的实体认证。本文将提供基于验证时间和基于验证次数的身份绑定方案。

身份绑定过程大致分为 3 个步骤：

1. 域名发送认证请求到区块链上；
2. 待交易确认后，根据交易和区块在服务端放置验证；
3. 验证节点对其进行验证，达到验证期限或者次数后，完成身份绑定。

4.2.1　基于验证时间的身份绑定方案

基于验证时间的方案旨在让绑定者提供验证信息持续一段时间，以供所有节点对其进行验证，当提供的验证达到指定时间后，即认为其对该域名服务器的所有权，完成指定地址和域名的绑定。

假设

假设绑定者 A 拥有公钥 Pk_A、私钥 Sk_A 以及域名 $example.com$，其需要完成 $(Pk_A, example.com)$ 的绑定；某一验证者 B 拥有公钥 Pk_B 和私钥 Sk_B。

交易类型

在本方案中，包含以下两种类型的交易：

1. 绑定请求交易 Tx_{req}：交易请求者发布绑定请求，包含了自己的公钥 Pk_A 和域名 $examle.com$。
2. 检举交易 Tx_{rep}：验证节点发现绑定者的验证信息放置不准确，可以提出检举交易，驳回身份绑定。

流程

绑定的具体流程包含以下四步：

1. A 发布绑定交易 $Tx_{req}(Pk_A, example.com)$ 到区块链上;
2. 根据需要的绑定信息 $(Pk_A, example.com)$，计算 $(Path, Chal) = F(Pk_A, example.com)$ 放置在自己控制的域名下，即访问 $example.com/Path$ 即可获取到 $Vcode = Sign_{Sk_A}(Chal)$ 值；

3. 验证者 B 访问域名验证 A 是否正确操作，如果不，发送检举交易 $Tx_{rep}(Pk_A, example.com)$ 到区块链;
4. A 保持验证时长 T 后，可停止该验证服务，完成绑定。

在该绑定过程中，首先需要绑定者将绑定信息自己的 Pk_A 和自己的域名 $example.com$ 发布到区块链上，将绑定身份信息展现给所有的验证节点，并根据函数 F 生成相关验证内容。我们选择哈希函数 $sha256$ 作为函数原型，将 Pk_A 和 $example.com$ 直接拼接后作为输入，将计算得到的 256 位哈希值前 128 位赋值给 $Path$、后 128 位赋值给 $Chal$，并且使用自己的私钥对 $Chal$ 进行签名：

$$Path_{128}||Chal_{128} = sha256(Pk_A||example.com) \tag{4.1}$$

$$Vcode = Sign_{Sk_A}(Chal) \tag{4.2}$$

绑定者将以上得到的 $Vcode$ 值放置在网站的 $example.com/Path$ 目录下，作为身份验证的内容，供所有验证节点验证。如果验证节点在验证过程中发现其并没有正确的放置验证信息，将可以通过发送检举信息 Tx_{rep} 对身份绑定消息进行驳回。

4.2.2 基于验证次数的身份绑定方案

基于验证次数的方案需要在绑定者发布绑定信息后，选择出合适的验证者，对绑定者的验证信息进行确认，当验证的次数达到规定次数后即可完成身份绑定。

假设

假设绑定者 A 拥有公钥 Pk_A，私钥 Sk_A 以及域名 $example.com$，其需要完成 $(Pk_A, example.com)$ 的绑定；某一验证者 B 拥有公钥 Pk_B 和私钥 Sk_B。

交易类型

在本方案由于需要特定验证者完成验证信息的确认，相比于基于时间的验证方案，增加了验证交易类型，包含一下三类交易类型：

1. 绑定请求交易 Tx_{req}：交易请求者发布绑定请求，包含了自己的公钥 Pk_A 和域名 $examle.com$。
2. 检举交易 Tx_{rep}：验证节点发现绑定者的验证信息放置不准确，可以提出检举交易，驳回身份绑定。
3. 验证交易 Tx_{vfy}：验证者在完成验证信息的对比后，在验证通过的情况下发送验证通过的交易。

流程

基于验证次数的身份绑定方案相比于基于验证时间的方案需要验证者更多的和区块链进行交互，在验证过程中需要将验证是否通过的信息提交到区块链上去，具体流程如下：

1. A 发布绑定交易 $Tx_{req}(Pk_A, example.com)$ 到区块链上;
2. A 获取 Tx_{req} 所在位置区块信息 $Info_{block}$，并根据其计算值 $(Path, Chal) = F(Info_{block})$ 放置在自己控制的域名下，及访问 $example.com/Path$ 即可获取到 $Vcode = Sign_{Sk_A}(Chal)$ 值；
3. 根据交易 Tx_{req} 计算符合该条请求的验证者：$v_1, v_2, ..., v_k$，假设 B 为其中一个验证者
4. B 从 $example.com/Path$ 获取验证内容，提交交易 $Tx_{vfy}(Pk_B, Pk_A, C)$ 完成验证。
5. 在经过 K 个验证过后，即完成绑定。

整体流程和基于时间的方案类似，需要绑定者将自己的公钥 Pk_A 和自己的域名 $example.com$ 发布到区块链上，然后通过公式函数获得(4.1)和(4.2)计算得到 $Path$ 和 $Vcode$ 的值放置到自己的服务器上；验证者通过(4.1)计算获得 $Path$、$Chal$ 和 Pk_A 对验证信息进行确认，确认通过后发送验证交易 Tx_{vry} 到区块链上完成一次验证，待达到验证次数 K 后，即完成身份绑定。

验证者的选取

在本方案中，需要一种选取选取验证者的规则，避免节点为了自己的利益而不纳入或者滞后其它节点的验证交易；更为重要的是，如果验证节点不是经过一定规则筛选得出的，而是每个节点都可以对绑定身份进行验证的话，在本方案中恶意的绑定者可以使用不同的公钥作为验证节点，然后对恶意绑定进行确认；同时，正常的验证者为了获得更多的奖励，也会申请尽量多的账户对交易进行验证。

为了避免以上的情况出现，本方案中设计了依据绑定信息随机选择验证节点的方法，保证验证可以有效的进行。当绑定信息被发布到区块链上后，根据绑定信息 Pk_A 和 $example.com$，将其装换为基准公钥 Pk_{cmp}:

$$Pk_{cmp} = sha_{512}(Pk_A || example.com) \tag{4.3}$$

在得到基准公钥后，验证节点计算公钥 Pk_v 与基准公钥 Pk_{cmp} 之间的海明距离，如果其距离小于某一个值 d，就具有验证该次绑定的权力，可以向区块链提交验证结果。为使得能够有住够多的验证者，让身份绑定过程可以更加高效的进行，随着时间的推移，对海明距离的约束 d 将会逐渐变大。

[需要展开一下]

4.2.3 检举的工作机制

在一个 p2p 的网络中或者一个区块链的网络中，每个用户使用都是一个公钥（即地址），和自身的身份是没有任何关系的，这就是为什么需要设计公钥与域名绑定方案的原因。但是在这样的网络中，任何人都可以发起对任意域名的绑定，为了排除一些错误的绑定或者假冒的绑定，需要设置检举的机制来完成这一目标。

当一个诚实的验证节点 C 发现一个虚假的绑定时，他就可以向区块链发起检举交易，其中包括自身的公钥 Pk_C 和绑定者公钥 Pk_A 和域名 $examle.com$。但是对于一个恶意的验证节点，可能也会对一个真实有效的绑定发起检举交易。为了确认一条检举交易的有效性，需要区块链上验证节点对该条交易给出判断，和4.2.2中选择验证节点的方式类似，这里使用（Pk_C，Pk_A，$examle.com$）作为输入，得到一个基准地址，然后挑选出 n 个节点对该检举交易进行投票，判断该次检举的有效性。

4.2.4 奖励惩罚机制

为了促使本方案有足够多的验证节点加入，保证身份绑定能够完整有效的进行，需要对验证节点给予一定的奖励。在验证过程中，如果发现存在未合理放置验证信息的域名，将可以发起检举交易到区块链上，待交易确认后，其可以得到相应的奖励; 为保证本系统中的奖励平衡，发起绑定者需要付出一定的代价，而验证者发现错误时，将可以获取得到相应的奖励。所以涉及到的奖励和惩罚机制包括以下几点:

- 在发起绑定交易时，绑定者需要付出一定的代价
- 在完成验证之后，验证者将会获得相应的奖励（无论是基于验证次数方案中的验证节点或是对检举交易验证的节点）
- 在发起检举交易时，需要对该交易的发起付出一定担保，在检举交易确认后将退回担保
- 在检举交易被认可后，检举发起者和相应的验证者将会获得奖励；在检举交易被否决时，相应的验证者可以获得检举交易的担保费用

第一条规则的设计让身份绑定者需要付出代价，而不是无限制的随意的在本方案中发起身份绑定操作，从而避免了恶意的破坏者无限度的去发起无效的身份绑定，从而影响本系统中正常身份绑定者的顺利进行；第二条规则是对付出验证操作的节点给予奖励，作为吸引更多节点加入的激励机制；第三条规则是为了防止网络中的验证节点随意的发起检举交易来扰乱正常绑定的进行；第四条规则和第二条规则类似，属于激励规则，初始更多的节点加入到本系统中。同时，由于验证者在完成验证或者发现验证不通过的时候，将会获得奖励，而这些奖励不是凭空产生的，而是验证发起者所付出的相应费用。

4.2.5　安全性分析

站点对域名服务器的控制权

为了保证绑定者对域名服务器拥有控制权，在以上方案中，使用提交的 Tx_{req} 来生成验证内容 $Path$ 和 $Chal$，保证了验证内容和提交请求的相关性；同时在(4.1)中使用 $sha256$ 作为随机函数，将提交内容转换为验证内容，该过程是一个不可逆的过程，从验证内容到提交内容的生成是很难完成的，保证了很难通过验证内容去构造提交内容。在放置的验证内容中，$Vcode$ 是使用绑定公钥对应私钥对 $Chal$ 进行签名得到的，保证了绑定者对私钥的拥有权。

验证交易真实有效

基于时间的验证方案中不需要提交者发起验证提交，在发现错误的时候菜需要发送检举交易；对于基于验证次数的方案而言，需要提交验证内容即 $Vcode$，因为有签名的存在，验证节点不能凭空的发送验证交易，需要通过服务器提供的验证信息才能正确的完成验证。

绑定者如果未能正确的放置验证内容，或者通过其它方式公布验证内容，导致验证过程依旧执行，那么其他验证节点可以通过检举交易对其发起检举。

验证时间的选择

在区块链上有确认交易的概念，以保证交易被正确的收录到了区块链中，并在很大程度上达到不可篡改的安全级别[2]。假设区块链上的交易确认块数为 Δ，出块的平均时间为 T_{avg}，在一个 $\Delta * T_{avg}$ 的时间之后，交易将被确认有效，在此期间，所有的节点都可以对该条交易的验证内容进行确认，一旦发现不符合的现象，即可发起检举交易，完成检举。

验证次数的选择

4.3　模块实现

本文以以太坊为例作为区块链底层架构，利用其智能合约完成以上方案的逻辑实现，PKI 各个实体通过与智能合约之间交互，存储并获取相关数据，完成对信任 CA 列表的读写，并最终完成对证书有效性的进一步确认。

根据上面的设计，以以太坊作为区块链基础，本系统如4.2图分为四个模块进行实现：智能合约、域名客户端、验证节点客户端、游览器验证插件。

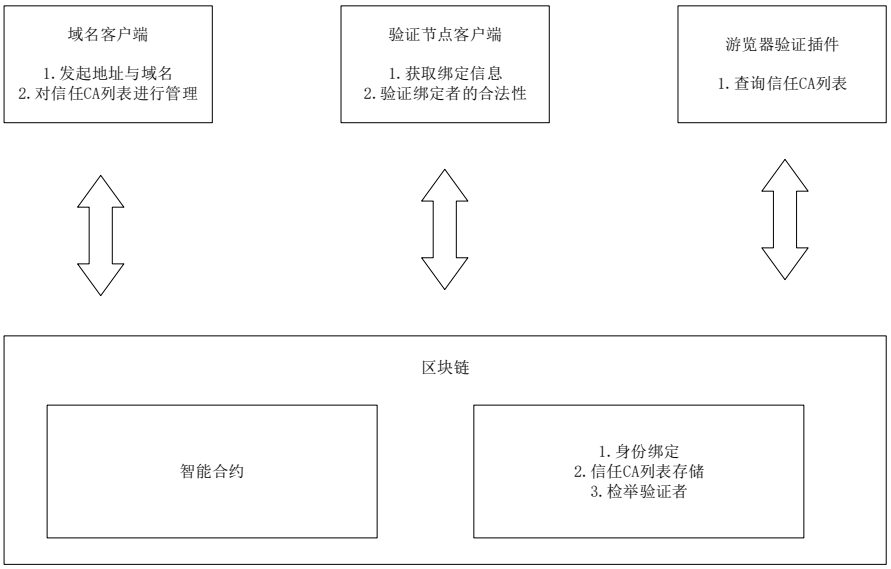

图 4.2　系统模块

4.3.1　智能合约

智能合约作为可以在区块链上实现其他逻辑操作的一种工具，在本部分，将给予以太坊作为底层区块链基础，在其上开发具有本文所述方案的智能合约，提供其它模块访问区块链的相关接口和功能，完成本文所述方案中的逻辑操作。

在智能合约中需要提供身份绑定接口、信任列表修改接口、信任列表存储接口以及检举接口。

身份绑定接口

身份绑定接口主要提供给域名客户端和验证节点客户端使用，针对不同的客户端有两种不同的的操作：身份绑定请求和身份绑定确认。

身份绑定请求操作是由域名通过域名客户端发起，发送需要绑定的域名和公钥到智能合约中进行登记，并放置合法的验证内容到自己的服务器上，供验证节点对其进行验证。

在身份绑定请求发送成功之后，验证节点客户端可以获得最新的待验证身份绑定信息，并对身份绑定的信息进行验证，如果验证通过，则会通过智能合约调用身份绑定确认操作，完成身份绑定验证。

信任列表修改接口

域名在完成身份绑定之后，即拥有了与域名相关联的公私钥对，通过对公钥对应内容的控制调整自己信任的 CA 列表。其通过调用信任 CA 列表对信任 CA 进行修改操作，并存储在区块链上供其它人查阅。

检举接口

当验证者发现带确认的域名并未正确放置验证内容时，即可调用检举接口，完成对公钥对本次域名绑定的检举。

4.3.2　域名客户端

域名作为证书的申请者，需要将自己的可信任 CA 列表存储到区块链上，以供证书受用者在使用证书的过程中对证书的真伪进行进一步检查。

域名客户端需要与区块链进行交互，实现域名和公钥的绑定以及对自己可信 CA 列表的控制，主要包括以下操作：

- 发起身份绑定请求
  通过调用智能合约身份绑定接口，发送需要绑定的域名和公钥，完成身份绑定请求操作。
- 修改信任 CA 列表
  在完成身份绑定之后，域名客户端将可以控制公钥和域名所对应的信任 CA 列表，通过调用智能合约中的信任列表修改接口，完成对自身信任 CA 列表的管控。

4.3.3　验证节点客户端

验证节点是本系统中身份鉴权的重要组成部分，是身份绑定有效性的保证，只有住够多验证节点存在的情况下，才可以保证身份确认的有效。

验证节点的主要工作是实时的查询验证请求并对验证请求进行验证，并根据其验证请求计算自己是否符合验证节点的要求，如果符合的话将进行验证确认操作，如果不是，也可以进行验证，对不符的验证提交检举交易。

4.3.4　游览器验证插件

游览器 Web PKI 中证书受用者的客户端，在原有体系结构中充当着检查证书真伪的角色。在本方案中，为了保证被恶意 CA 私自签发的证书可以被识别出来，需要在客户端进行对证书进行额外的检查，也就是需要游览器需要有区块链进行交互，查询收到的证书是否由证书申请者信任的 CA 所签发。

第五章　系统测试？

pkuthss 文档模版最常见问题：

\cite、\parencite 和 \supercite 三个命令分别产生未格式化的、带方括号的和上标且带方括号的引用标记：1，[4]、[1,4]。

若要避免章末空白页，请在调用 pkuthss 文档类时加入 openany 选项。

如果编译时不出参考文献，请参考 texdoc pkuthss“问题及其解决”一章“其它可能存在的问题”一节中关于 biber 的说明。

结论

pkuthss 文档模版最常见问题：

\cite、\parencite 和 \supercite 三个命令分别产生未格式化的、带方括号的和上标且带方括号的引用标记：1，[4]、[1,4]。

若要避免章末空白页，请在调用 pkuthss 文档类时加入 openany 选项。

如果编译时不出参考文献，请参考 texdoc pkuthss“问题及其解决”一章“其它可能存在的问题”一节中关于 biber 的说明。

参考文献

[1] Author. “*Title*” [J]. *Journal*, 2014-04-01.

[2] Satoshi Nakamoto. “*Bitcoin: A peer-to-peer electronic cash system*”. **2008**.

[3] Joel Weise. “*Public key infrastructure overview*”. *Sun BluePrints OnLine, August*, **2001**: 1–27.

[4] 作者。“标题”[J]。期刊，2014-04-01。

附录 A　附件

pkuthss 文档模版最常见问题：

\cite、\parencite 和 \supercite 三个命令分别产生未格式化的、带方括号的和上标且带方括号的引用标记：1，[4]、[1,4]。

若要避免章末空白页，请在调用 pkuthss 文档类时加入 openany 选项。

如果编译时不出参考文献，请参考 texdoc pkuthss“问题及其解决”一章“其它可能存在的问题”一节中关于 biber 的说明。

致谢

pkuthss 文档模版最常见问题：

\cite、\parencite 和 \supercite 三个命令分别产生未格式化的、带方括号的和上标且带方括号的引用标记：1，[4]、[1,4]。

若要避免章末空白页，请在调用 pkuthss 文档类时加入 openany 选项。

如果编译时不出参考文献，请参考 texdoc pkuthss“问题及其解决”一章“其它可能存在的问题”一节中关于 biber 的说明。

北京大学学位论文原创性声明和使用授权说明

原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品或成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律结果由本人承担。

论文作者签名：　　日期：　　年　　月　　日

学位论文使用授权说明

（必须装订在提交学校图书馆的印刷本）

本人完全了解北京大学关于收集、保存、使用学位论文的规定，即：

- 按照学校要求提交学位论文的印刷本和电子版本；
- 学校有权保存学位论文的印刷本和电子版，并提供目录检索与阅览服务，在校园网上提供服务；
- 学校可以采用影印、缩印、数字化或其它复制手段保存论文；
- 因某种特殊原因需要延迟发布学位论文电子版，授权学校在□一年/□两年/□三年以后在校园网上全文发布。

（保密论文在解密后遵守此规定）

论文作者签名：　　导师签名：　　日期：　　年　　月　　日