

# 密码学原理与实践实验报告 (认证服务系统)

韦昆杰

1181000420

2021 年 1 月 7 日

## 目录

<b>1 背景与意义</b>	<b>2</b>
1.1 项目开发意义 . . . . .	2
1.2 国内外现状及技术综述 . . . . .	3
<b>2 需求分析</b>	<b>4</b>
2.1 总体需求 . . . . .	4
2.2 功能需求 . . . . .	4
2.3 性能需求 . . . . .	5
<b>3 概要设计</b>	<b>5</b>
3.1 接收验证用户数字证书的申请 . . . . .	5
3.2 生成证书 . . . . .	5
3.3 存储证书 . . . . .	6
3.4 向申请者颁发 (或拒绝颁发) 数字证书 . . . . .	6
3.5 接收用户数字证书的查询、撤销 . . . . .	6
3.6 产生和发布证书的有效期 . . . . .	6
3.7 数字证书的归档 . . . . .	6
3.8 密钥归档 . . . . .	6
<b>4 详细设计</b>	<b>7</b>
4.1 主页面 . . . . .	7
4.2 注册页面 . . . . .	7

1 背景与意义	2
4.3 登录页面	8
4.4 验证证书页面	9
4.5 个人页面	10
4.6 申请证书	11
4.7 下载证书	12
4.8 撤销证书	12
4.9 查看证书	13
4.10 个人信息	13
4.11 生成口令	14
5 实现与测试	14
5.1 数字证书	14
5.2 认证安全	17
5.3 存储安全	17
5.4 传输安全	18
5.5 注册测试	18
5.6 登录测试	20
5.7 申请证书测试	21
5.8 验证码	21
5.9 口令	22
5.10 密码隐藏显示	23
6 结束语	23
7 参考文献	23

## 1 背景与意义

### 1.1 项目开发意义

数字证书认证机构（英语：Certificate Authority，缩写为 CA），也称为电子商务认证中心、电子商务认证授权机构，是负责发放和管理数字证书的权威机构，并作为电子商务交易中受信任的第三方，承担公钥体系中公钥的合法性检验的责任。所有通过 CA 的信息传输方，都要无条件的信任 CA

的公正性,在消息传输的过程中,CA 为信息传输的双方提供公私钥加密环境,提供身份认证、安全传输、不可否认性和数据完整性等功能。

公钥基础设施 PKI (英语: Public Key Infrastructure, 缩写: PKI), 是一组由硬件、软件、参与者、管理政策与流程组成的基础架构,其目的在于创造、管理、分配、使用、存储以及撤销数字证书。PKI 技术采用证书管理公钥,通过第三方的可信任机构—认证中心 CA(Certificate Authority),把用户的公钥和用户的其他标识信息(如名称、E-mail、身份证号等)捆绑在一起,在 Internet 上验证用户的身份(其中认证机构 CA 是 PKI 系统的核心部分)。目前,通用的办法是采用建立在 PKI 基础之上的数字证书,通过把要传输的数字信息进行加密和签名,保证信息传输的机密性、真实性、完整性和不可否认性,从而保证信息的安全传输。

## 1.2 国内外现状及技术综述

美国的 PKI 建设过程经历了 1996 年之前的无序、1996—2002 年间以 FBCA 为核心的体系搭建、2003 之后策略管理和体系建设并举的三个阶段。1996 年以前,很多政府部门自建 PKI 系统,例如美国邮政服务部门、社会安全部门、美国国防部、能源部、美国商标与知识产权局等。1996 年美国提出联邦桥接计划,2001 年正式公布,计划最终建立一个覆盖美国 80 个机构、19 个部的 PKI 以保护电子政府的通信安全。

美国联邦 PKI 体系主要由联邦的桥认证机构 (Federal Bridge CA, FBCA)、首级认证机构 (Principal CA, PCA) 和次级认证机构 (Subordinate CA, SCA) 等组成。联邦 PKI 的体系结构中没有采用根 CA,而采用了首级 CA。

这是因为在美国,信任域的结构是多种多样的,美国联邦 PKI 体系结构可以支持分级(树状)维构、网状结构和信任列表等。联邦的桥 CA 是联邦 PKI 体系中能核心组织,是不同信任域之间能桥梁,主要负责为不同信任域能首级 CA 颁发交叉认证的证书,建立各个信任域的担保等级与联邦 CA 的担保等级之间的映射关系,更新交叉认证证书,发布交叉认证证书注销黑名单。但是联邦的桥 CA 不要求一个机构在与另一个机构发生信任关系时必须述循联邦 PKI 所确定的这种映射关系,而是可以采用它认为合适的映射关系确定彼此之间的信任。

欧洲在 PKI 基础建设方面也成绩显著。已颁布了 93/1999EC 法规,强调技术中立、隐私权保护、国内与国外相互认证以及无歧视等原则。为了解

决各国 PKI 之间的协同工作问题，它采取了一系列措施：积极资助相关研究所、大学和企业研究 PKI 相关技术；资助 PKI 互操作性相关技术研究，并建立 CA 网络及其顶级 CA。并于 2000 年 10 月成立了欧洲桥 CA 指导委员会，于 2001 年 3 月 23 日成立了欧洲桥 CA。

我国的 PKI 技术从 1998 年开始起步，政府和各有关部门对 PKI 产业的发展给予了高度重视。2001 年 PKI 技术被列为“十五”863 计划信息安全主题重大项目，并于同年 10 月成立了国家 863 计划信息安全基础设施研究中心。国家电子政务工程中明确提出了要构建 PKI 体系。我国已全面推动 PKI 技术研究与应用。2004 年 8 月 28 日，十届全国人大常委会第十一次会议 28 日表决通过了电子签名法，规定电子签名与手写签名或者盖章具有同等的法律效力。这部法律的诞生极大地推动了我国的 PKI 建设。

1998 年国内第一家以实体形式运营的上海 CA 中心 (SHECA) 成立，此后，PKI 技术在我国商业银行、政府采购以及网上购物中得到了广泛应用。国内的 CA 机构大致可分为区域型、行业型、商业型和企业型四类，并出现了得安科技、创原世纪、国创科技、吉大正元、国瑞数码等一批 PKI 服务提供商。

## 2 需求分析

### 2.1 总体需求

设计实现一个认证服务系统，在消息传输的过程中，CA 为信息传输的双方提供公私钥加密环境，提供身份认证、安全传输、不可否认性和数据完整性等功能。

### 2.2 功能需求

- 接收验证用户数字证书的申请
- 生成证书
- 存储证书
- 向申请者颁发（或拒绝颁发）数字证书
- 接收用户数字证书的查询、撤销

- 产生和发布证书的有效期
- 数字证书的归档
- 密钥归档

## 2.3 性能需求

技术栈：reactjs+material-ui+mysql

本实验选用 react 作为前端框架，react 通过引入虚拟 DOM 的方式显著减少了每次更新 UI 所进行的 DOM 操作，提高了效率。

通过采用客户端渲染 (Client-side Rendering)，相较服务器渲染 (Server-side Rendering) 每次都要向服务器请求 HTML 页面性能显著提升。

最后，每次前端向后端 api 接口请求数据，并根据请求数据的结果更新 DOM，实现了前后端分离的同时也避免了不必要的页面更新。

# 3 概要设计

围绕着需求分析，将展开如下设计：

## 3.1 接收验证用户数字证书的申请

1. 用户首先登录 CA 认证系统 (无账号需要先注册)
2. 用户进入申请证书页面，输入域名、国家、省份、城市、组织、组织单位等信息，在“1 年”、“2 年”、“3 年”中选择一项作为证书的有效期，点击提交按钮。
3. 后台处理用户的申请证书信息，将这些信息归档后，生成证书供用户下载。
4. 用户下载证书到本地

## 3.2 生成证书

后端根据用户申请证书输入的信息生成相应证书

### 3.3 存储证书

后端生成证书后将证书序列号、开始时间、截至时间、证书、公私钥、证书对应用户存储在数据库中 (私钥用 AES 加密后存储)

### 3.4 向申请者颁发（或拒绝颁发）数字证书

如果申请者为已经注册证书的域名申请证书，会拒绝颁发，否则向申请者颁发数字证书

### 3.5 接收用户数字证书的查询、撤销

- 查询：用户进入下载证书页面，输入需要查询的证书序列号，后台会查找数据库寻找到相应证书供用户下载
- 撤销：用户进入撤销证书页面，输入需要撤销的证书序列号和登录密码，如果用户是证书所有者后台会查找数据库寻找到相应证书并且撤销，否则拒绝请求。

### 3.6 产生和发布证书的有效期

证书的有效期为从证书制作时间起，1 年/2 年/3 年止，时限由用户在申请证书时选择。

### 3.7 数字证书的归档

数字证书的归档一律以文件的形式存储在本地路径中，同时详细信息以字段形式保存在数据库中。

### 3.8 密钥归档

密钥的归档一律以文件的形式存储在本地路径中，同时密钥以字段形式保存在数据库中 (公钥直接保存，私钥用用户输入口令的 AES 加密后保存)。

## 4 详细设计

### 4.1 主页面

# Welcome to HIT数字证书认证中心

登录 →

登录已有账号

注册 →

注册新账号

验证证书 →

上传证书并进行验证

个人页面 →

登录后直接进入个人页面

网站的主页面，用户可以选择登录、注册、验证证书或进入个人页面

### 4.2 注册页面



The registration form is titled "HIT数字证书认证中心" (HIT Digital Certificate Authentication Center). It features a red lock icon and the word "注册" (Register) in red. The form includes three input fields: "用户名\*" (Username), "密码\*" (Password), and "确认密码\*" (Confirm Password). Below these fields are two checkboxes: "Remember me" and "进行人机身份验证" (Human-machine identity verification). The "进行人机身份验证" checkbox is accompanied by a reCAPTCHA logo and the text "reCAPTCHA 数秘院 · 使用策略". At the bottom of the form is a blue "SIGN UP" button. A link "Already have an account? Sign in" is located at the bottom right of the form.

用户注册需要输入用户名和密码，需要满足以下几个条件：

- 用户名必须为字母、数字、下划线、减号的组合，长度为 6 16 位

- 密码至少包括数字和字母，长度至少为 6 位，至多 20 位。

最后用户还需要确认密码以及进行人机身份验证，一切无误后点击注册按钮即可成功注册，相关信息（密码用 bcrypt 哈希）会存入到数据库的 user 表中

	username	password
1	bank123	\$2a\$10\$k0kT1gN605JSDSdBZih2x.www074QCn9SBC075B2s97yPyHV6QTHy
2	shop123	\$2a\$10\$G2VUP0Actz0UZYI/0izV/e0u5.I/oRCjPojkJv.jf02rD9cFnzDuy
3	test123	\$2a\$10\$NI2laemxM24UGPXv2uZXfuSNpc7tB8tJhZVv9p7YZqs6SSqlv5Zy2
4	user123	\$2a\$10\$.H8LBQBosCY50bvBxygEwug/yR8USe0kxKrPKfJ66GUqQPcXLqF0e

### 4.3 登录页面



The image shows a login page for the HIT Digital Certificate Authentication Center. The page has a light gray background with a subtle geometric pattern. At the top, the title 'HIT数字证书认证中心' is displayed in a white box. Below the title, there is a red circular icon with a white padlock, and the word '登录' (Login) is written in black. The login form consists of two input fields: '用户名 \*' (Username \*) and '密码 \*' (Password \*). Below these fields, there is a checkbox labeled 'Remember me'. Further down, there is a checkbox labeled '进行人机身份验证' (Perform human-machine identity verification) next to a reCAPTCHA logo. At the bottom of the form, there is a blue button labeled 'SIGN IN'. Below the button, there are two links: 'Forgot password?' and 'Don't have an account? Sign Up'.

用户登录需要输入用户名和密码，需要满足以下几个条件（和用户注册一样）：

- 用户名必须为字母、数字、下划线、减号的组合，长度为 6 16 位
- 密码至少包括数字和字母，长度至少为 6 位，至多 20 位。

用户登录需要输入用户名、密码，并通过人机身份验证。后台将从数据库中找到用户名对应的密码，对用户输入的密码用 bcrypt 哈希后与数据库中的密码比对，如果相同，则登陆成功。



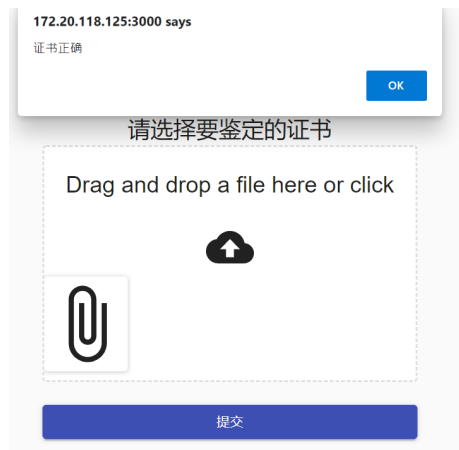
```
//比较用户登录密码是否与数据库密码相同(Hash)
bcrypt.compare(password, result[0].password, function (error, response) {
```

#### 4.4 验证证书页面



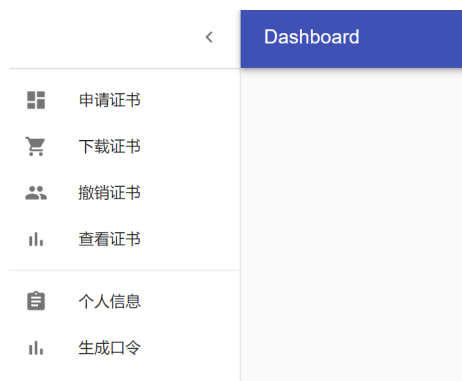
The image shows a web interface for certificate verification. At the top, there is a red circular icon with a white document symbol. Below it, the text "请选择要鉴定的证书" (Please select the certificate to be verified) is displayed. A dashed-line box contains the text "Drag and drop a file here or click" and a cloud icon with an upward arrow. Below this box is a blue button labeled "提交" (Submit). At the bottom, there is a small copyright notice: "Copyright © Your Website 2021."

用户可以点击上传证书文件 (pem 格式), 点击提交, 后端验证证书, 如果证书未过期且由本 CA 签发就会提示证书正确, 否则警告证书错误



The image shows the same certificate verification page as before, but with a success message overlay. The message box at the top left says "172.20.118.125:3000 says" and "证书正确" (Certificate is correct), with an "OK" button. The main interface remains the same, with the "提交" (Submit) button at the bottom.

### 4.5 个人页面



只有用户登录后才能进入对应的个人页面，未登录用户选择进入个人页面会直接跳转到登录页面。

进入个人页面后，可以选择相应的功能页面，主操作面板中一共包括 6 个功能：

- 申请证书
- 下载证书
- 撤销证书
- 查看证书
- 个人信息 (输入口令可以查看私钥)
- 生成口令

## 4.6 申请证书

A vertical form for applying for a certificate. It consists of eight input fields stacked vertically, each with a label and an asterisk indicating it is required. The labels are: '域名 \*', '国家 \*', '省份 \*', '城市 \*', '组织 \*', '组织单位 \*', '有效期 \*', and '口令 \*'. The '有效期 \*' field is a dropdown menu. At the bottom of the form is a blue button with the text '提交' (Submit).

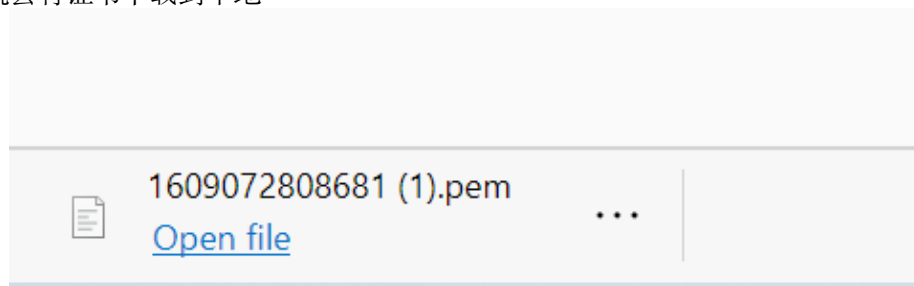
申请证书页面包括一个 container，其中有用户申请证书需要提交的表单。该表单包括：

- 域名
- 国家
- 省份
- 城市
- 组织
- 组织单位
- 有效期（1 年/2 年/3 年）
- 口令

其中域名即是申请的网站网址，有效期只能三选一，口令用于对私钥加密（可以从生成口令页面中随机生成）

## 4.7 下载证书

用户输入想要下载的证书序列号，点击提交按钮，如果该证书在数据库中就会将证书下载到本地



点击查看：

```
C: > Users > 34897 > Downloads > 1609072808681 (1).pem
1  |-----BEGIN  CERTIFICATE-----
2  MIICEjCCAXugAwIBAgIHAWCQcoCGgTANBgqhkiG9w0BAQUFADA5MQswCQYDVQQD
3  EwJDQTEdMBsGA1UEChMUQ2VydG1maWNhdGVbdXR0b3JpdHkxCzAJBgNVBAYTAkNO
4  MB4XDTEwMTIyNzEyNDAwOFoXDTEwMTIyNzEyNDAwOFowXzETMBEGA1UEAxMKaG10
5  LmVkdS5jbjELMAkGA1UEBhMCY24xDjAMBGNVBAgTBWh1bmFuMQ8wDQYDVQQHEwZh
6  bn1hbmcyDDAKBgNVBAoTA2hpdDEMMAoGA1UECjMDaG10MIGfMA0GCSqGSIb3DQEB
7  AQUAA4GNADCBiQKBgQC689aTjfpvP9DqOI2ihsJ0/HNnsstepE1tWlQBvBRRd8Nn
8  9WoD1GIQgO3EQBV1/SWy3QH2MAxdCq12TCA1S8RY60fmdkT5QEv3Yiqz14Hur0
9  DXC5noDdPxYlmsSzoNKdBtZAn00C4m9CetwzLr3F3xu50066XLRJm3jPm1Y1NQID
10 AQABMA0GCSqGSIb3DQEBBQUAA4GBAG0vtDOe94Hx78JLi8uqkVeoQfjSqaxUT+/T
11 VrtOTjMc70v3S/W+zpNSMbSgz4Q3JujBari6EKRj07VpscfeYNGeH0zI9YbH/gP
12 j8Eyp0841yZknvNhAwXf+cCq3nFPsWlF0rp53h/HJGFNGgJv6tjTdoXnS0EPIX4
13 4k29C0Ar
14 -----END  CERTIFICATE-----
15
```

## 4.8 撤销证书

用户输入要撤销的证书序列号 and 用户密码，如果该证书为用户申请并且密码正确，后台验证就可以成功撤销该证书。

其实大型公证 CA 系统的证书的申请和撤销都是比较严谨的工作，通常需要多方核实，并且在一到三个工作日内给出证书的申请和撤销反馈。本次实验中，对于证书的申请没有加权限（任何登录用户都能申请证书），证书的撤销也只加了登录密码的权限限定（允许用户撤销自己申请的证书）。

## 4.9 查看证书

证书序列号 \*

提交

用户输入要查看的证书序列号，如果数据库中有该证书，就可以查看到相应证书的信息

[illegible]

## 4.10 个人信息

```
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAK8gDQlMmNnwU7eNoX3s20iBNtt/hw7LaKr/J2nsHvprvwwNuz15pzs
0aAimB9Y7OowJcLLH0skqBuYmZvudhrCxZ0wA9qGNeVbUGHCe5FKtN6Pn8DS+73D
htS0x1BHK/Y1k2kyOpL/9RJHS/DB108NDIMGFLiQfQ+T8115B4otbEn1wIDAQAB
AoGBAKUYZ6hAzMkNqHxKwB6eq31CM1JzsB28d1WY68EaPvOS5M2auUS0m8z11Pt
Zd8Iwqr/qR20Q1Hyb0VqEY2ORjGzL5Q2iHdAOpmCv6mD1gZk1UbMeAlat+Lien
bd2F+VK+ZSofFmhjs5cibt9eKe2dYcPUWv2YLH/+C3DGTlABAKEA84VpC/sSczO3
kX0dvQcgN8K9QRzt3iHlMPGXaoMwV10d5YMqW715yVA9V1/GRG0AkkYrMv24XfXD
RUVbhl0AQJBAOGXUmeUfGRg/zku11SQZF9F9AvJEPxLU/jSv/aesrLtL1qvuaA0
9hSnhIq8mTCF+Pgu3S06CEB13T054xqc25cQ0EmVZ2+KV0HeP7DnqQf0rmqK+prh
PFmNgNptDQLPRGEQDAec+eVTRR5S6kACNjYvX8BNC/vbHk494wp983LJO+AECQDP
rkqjYEIrdk/+qGuPPFXhUXIuac80dvv+L6vgdpckhJZ9ozYjklzAj74Vd8AmJT2
hM33kc7Tas48gOKpIsrJAKAoLTiZ6HqLUCHQtpHfERLlie9w/XdwnFnI0YHxxa
wNfNxf0rQyPwWPhEjBfEaY5maSFH7LgrqPdut2Mrqk6
-----END RSA PRIVATE KEY-----
```

用户可以查看当前登录用户名，并且输入申请证书时的口令就可以查

看对应私钥

### 4.11 生成口令



每次点击生成口令就可以随机生成一个由 4 或 5 个英语单词组成的口令 (passphrase)，由于口令由英语单词组成，因此相较复杂无规则的口令更容易记忆，同时其安全性甚至更好

安全性分析：从 65536 个单词的列表中随机选取单词，每个单词的熵值是 16bit，因此 5 个单词的熵值就是 80bit，作为口令很难依靠穷举破解（假设敌手每秒尝试 1,000,000,000,000 次，尝试一半的密钥空间成功破解，其需要的时间大约是 1 年）

## 5 实现与测试

### 5.1 数字证书

X.509 是密码学里公钥证书的格式标准。X.509 证书已应用在包括 TLS/SSL 在内的众多网络协议里，同时它也用在很多非在线应用场景里，比如电子签名服务。X.509 证书里含有公钥、身份信息（比如网络主机名，组织的名称或个体名称等）和签名信息（可以是证书签发机构 CA 的签名，也可以是自签名）。对于一份经由可信的证书签发机构签名或者可以通过其它方式验证的证书，证书的拥有者就可以用证书及相应的私钥来创建安全的通信，对文档进行数字签名。

证书组成结构标准用 ASN.1（一种标准的语言）来进行描述，X.509 v3 数字证书结构如下：

- 证书

- 版本号
  - 序列号
  - 签名算法
  - 颁发者
  - 证书有效期
    - \* 此日期前无效
    - \* 此日期后无效
  - 主题
  - 主题公钥信息
    - \* 公钥算法
    - \* 主题公钥
  - 颁发者唯一身份信息（可选项）
  - 主题唯一身份信息（可选项）
  - 扩展信息（可选项）
    - \* ...
- 证书签名算法
  - 数字签名

数字证书广泛应用在 https 等网络协议上,我们访问学校官网 <https://www.hit.edu.cn/>, 查看网站的证书



我们可以看到版本号、序列号、签名算法等证书信息

参考 X.509 证书结构，本实验设计实现根证书，将根证书导入到证书列表查看其信息



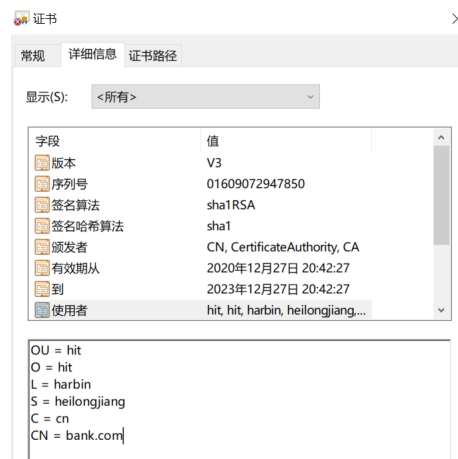
本实验需要用户输入域名、国家、省份、城市、组织、组织单位、有效期（1 年/2 年/3 年），然后生成相应的证书。

我们可以将根证书签名的用户证书同样导入到证书列表查看信息，此处以银行申请的证书为例：

颁发给: bank.com

颁发者: CA

有效期从 2020/12/27 到 2023/12/27





## 5.2 认证安全

本 CA 系统的认证通过 jwt(JSON Web Token) 实现，当用户成功登录后，后台会签名一个 token 并存储在 localStorage 中。对于非公开页面(例如申请证书、撤销证书等个人页面)，前端会通过 localStorage 检查 (verify)token，如果不存在或是错误会跳转到登陆页面，这样保证了个人页面只有登录才能查看。最后，本实验设置 token 时间为 1 小时，过期后用户需要重新登录。

```
//比较用户登录密码是否与数据库密码相同(Hash)
bcrypt.compare(password, result[0].password, function (error, response) {
  if (error) {
    console.log(error);
  }
  //如果相同，设置token
  if (response) {
    const token = jwt.sign({ iss: "wkj", "username": username }, secret, { expiresIn: '1h' });
```

## 5.3 存储安全

本 CA 系统的存储安全只作用在用户密码和证书私钥，由于证书信息是公开的因此没有进行加密。

- 对于用户密码，通过 bcrypt 哈希后存储到数据库中
  1. bcrypt 是一个由 Niels Provos 以及 David Mazières 根据 Blowfish 加密算法所设计的密码散列函数，于 1999 年在 USENIX 中展示。实现中 bcrypt 会使用一个加盐的流程以防御彩虹表攻击，同时 bcrypt 还是适应性函数，它可以借由增加迭代之次数来抵御日益增进的电脑运算能力透过暴力法破解。
  2. 由 bcrypt 加密的文件可在所有支持的操作系统和处理器上进行转移。它的口令必须是 8 至 56 个字符，并将在内部被转化为 448 位的密钥。然而，所提供的所有字符都具有十分重要的意义。密码越强大，您的数据就越安全。相较普通的哈希函数加盐 (HMAC) 方法，bcrypt 一次哈希所需要的时间大大增加，因此敌手依靠穷举破解的办法很难实现。
  3. 例子：在一个普通笔记本电脑，将密码 yaaa 通过 bcrypt 哈希需要 0.3 秒，而用 MD5 哈希该密码只需要不到一微秒)

- 对于证书私钥，通过用户输入的口令 (passphrase) 对私钥 (message) 进行 AES 加密。由于口令是安全的 (难以通过暴力破解)，AES 加密后的密文也是安全的

```
function encrypt(message, passphrase) {  
  const encrypted = CryptoJS.AES.encrypt(JSON.stringify(message), passphrase).toString();  
  return encrypted;  
};
```

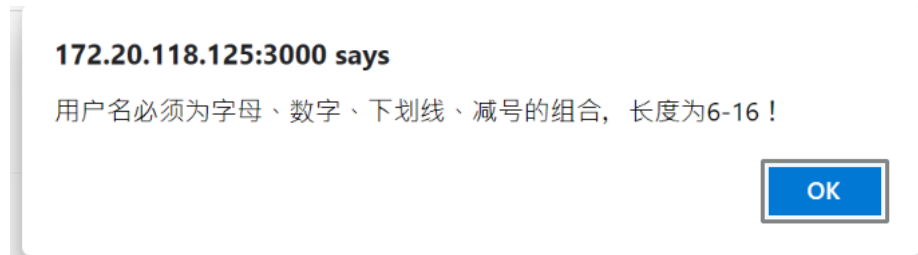
## 5.4 传输安全

本小组通过电子信封和双重签名等实现了银行、顾客、电商三方交易信息传输的安全性。借鉴 SET 协议，银行电商交换证书并通过 CA 验证证书真伪实现了相互认证 (由 CA 负责为通信双方提供信用担保)，如下是验证证书真伪的函数：

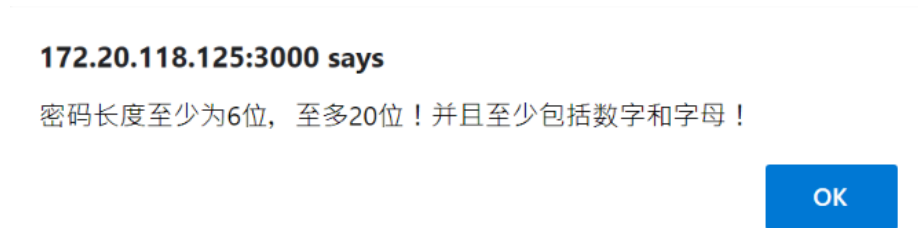
```
export default function verify(pem) {  
  try {  
    const root_cert = forge.pki.certificateFromPem(root_certificate);  
    const cert = forge.pki.certificateFromPem(pem);  
    if (!root_cert.verify(cert)) {  
      return false;  
    }  
    const now = new Date().getTime();  
    const from = cert.validity.notBefore.getTime();  
    const to = cert.validity.notAfter.getTime();  
    if (now < from || now > to) {  
      return false;  
    }  
    return true;  
  } catch (err) {  
    console.log(err);  
    return false;  
  }  
}
```

## 5.5 注册测试

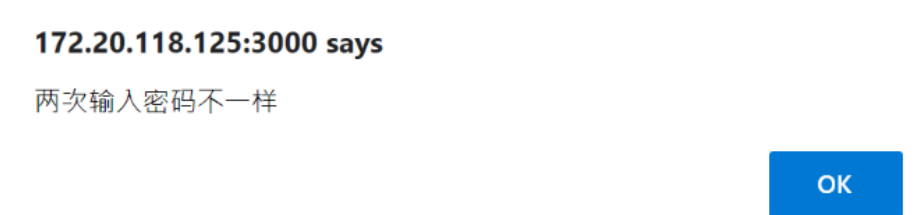
我们输入用户名、密码、确认密码，人机验证通过后点击注册按钮就可以成功注册用户名不合法：



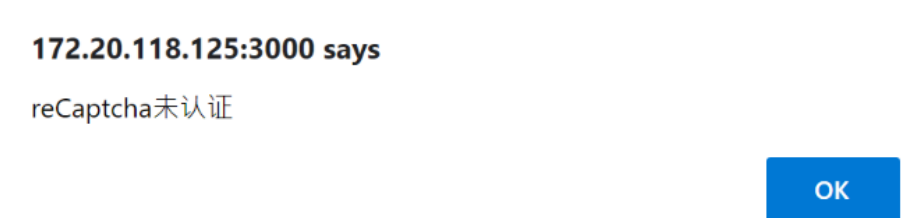
密码不合法：



两次输入密码不一样：



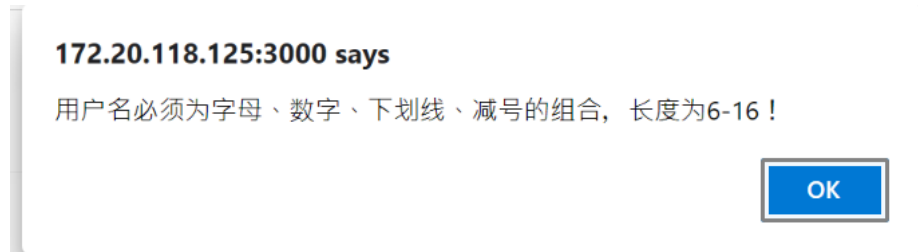
未进行人机身份验证：



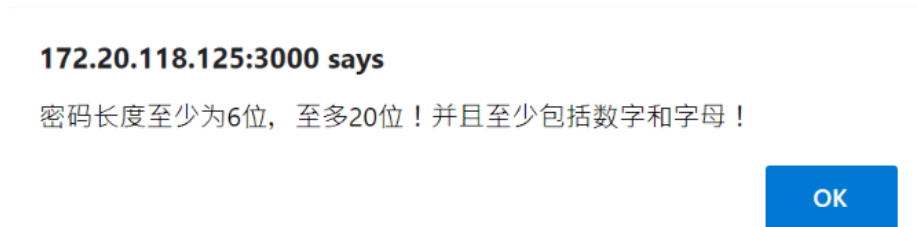
注册成功直接跳转到登录界面

## 5.6 登录测试

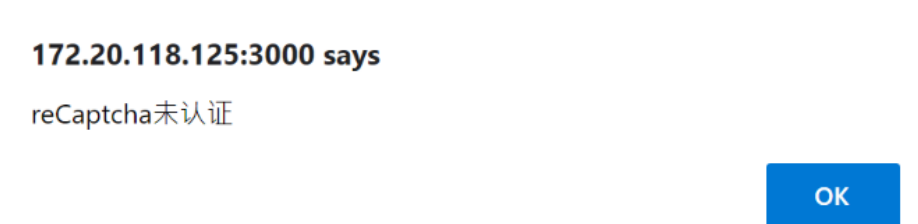
在登录界面我们输出注册的用户名和密码，人机验证通过后点击登录按钮，后台进行验证，通过后就会进入用户功能页面用户名不合法：



密码不合法：



未进行人机身份验证：



用户名/密码错误：当用户的用户名或密码输入错误时，不应该显式地指出到底是用户名错误还是密码错误，防止攻击者对用户名或密码针对性的攻击。

**172.20.118.125:3000 says**

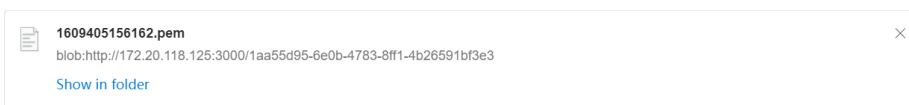
用户名不存在或密码错误

OK

登录成功后跳转到用户界面

## 5.7 申请证书测试

用户输入证书相关信息和私钥口令，点击提交按钮，申请的证书就会下载到本地



点击查看：

```
1  -----BEGIN  CERTIFICATE-----
2  MIICEDCCAXmgAwIBAgIHAWCUBRVhYjANBgkqhkiG9w0BAQUFADA5MQswCQYDVQQD
3  EwJDQTEdMBsGA1UEChMUQ2VydG1maWNhdGVDbXRob3JpdHkxCzAJBgNVBAYTAkNO
4  MB4XDTIwMTIzMTA4NTkxN1oXDTIxMTIzMTA4NTkxN1owXTERMA8GA1UEAxMIdGVz
5  dC5jb20xCzAJBgNVBAYTAjNlMQ4wDAYDVQQIEwVoZW5hbGJEPMA0GA1UEBxMGYW55
6  YW5nMQwwCgYDVQQKEwNoaXQxDDAKBgNVBAsTA2hpdDCBnzANBgkqhkiG9w0BAQEF
7  AAOBjQAwgYkCgYEA1pgzZ8F03jaF97NjogTbbf4Vuy2iq/ydp7B76a78MDbs5eac
8  7DmgIpgfW0zqMCXCyxzrKpMgbjNr7nYawsWTsAPahjXlW1BhwhORSrTej5/A0vu9
9  w4U7NMdQRyv2NZNpMjqS//USR0vwwZTtDQ5TBhSyKhUPk/NZedQeKLWxJ5cCAwEA
10 ATANBgkqhkiG9w0BAQUFAAOBgQCRirioGghutFrbsR15LxpQuw/BIcBtkHNptXe
11 PvqhbTpeq6iClR1Z2OuAcikwcibr2d4iEFeiWwgYaZpN2j9IR0f/L8Zbm7RsCSE8
12 09P7Vd/qLg1wr-fIHxU0iXB/t7iJEIXHku4LX12GQ00XheUxGGcGpMBTwpv6EqkR/
13 7sTwGQ==
14  -----END  CERTIFICATE-----
15  |
```

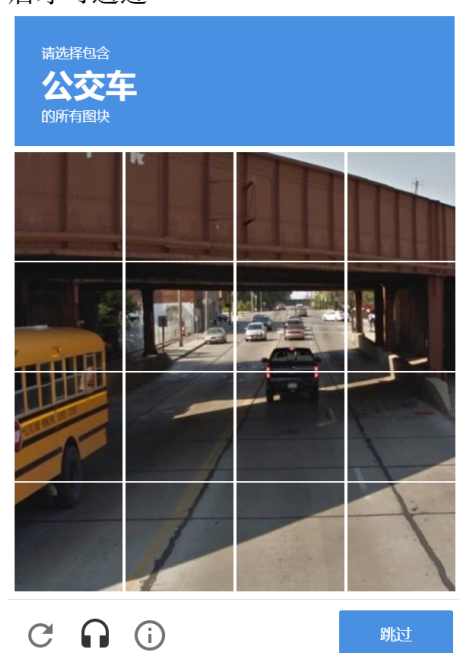
## 5.8 验证码

验证码的作用是为了给恶意破解密码、反复自动提交表单攻击造成麻烦。本 CA 系统中要求用户登录/注册时进行人机身份验证。

本实验采用的验证码是 Google 的 reCAPTCHA，系统会对用户在网站上的动作进行评分，如果正常点击进行人机身份验证即可通过



否则系统判别异常，就会要求用户选择包括某一物体的所有图块，正确选择后才可通过



## 5.9 口令

区别于密码，口令应该容易被人记忆，但难以被敌手破解。本实验要求用户在申请证书时输入口令用来加密私钥。之后用户可以通过输入口令得到私钥，CA 只保存加密的私钥而不保存口令，其他任何人在不知道口令的情况下即使破解数据库也没用（数据库中的私钥用口令 AES 加密）。

本实验采用的是由英语单词构成的口令，英语单词在 65536 个单词的列表中随机选取。这样用户只需要记忆几个英语单词即可，而靠蛮力破解口令非常困难。

### 5.10 密码隐藏显示



本实验登录、注册页面密码默认隐藏，可以避免偷窥攻击。

## 6 结束语

经过了八周的密码学实验实践，从最初基本的构思到最终各种功能的实现，实现了一个具有申请、撤销、下载、查询等基本功能的 CA 认证系统

通过这次实验，我学习了前端框架 react，掌握了 mysql 数据库的各种基本操作，了解到了传输安全、存储安全等概念，通过自行设计数字证书对数字证书有了更深的理解

最后，我和我的队友们三个人合作为了将各自的程序交互起来，在开发过程中团队积极相互沟通，解决了很多困难，我也理解到了团队合作的重要性

## 7 参考文献

- X.509-维基百科 [<https://en.wikipedia.org/wiki/X.509>]
- Optimizing Performance - React [<https://reactjs.org/docs/optimizing-performance.html>]
- reCAPTCHA-维基百科 [<https://en.wikipedia.org/wiki/ReCAPTCHA>]
- bcrypt-维基百科 [<https://zh.wikipedia.org/zh-cn/bcrypt>]
- How To Safely Store A Password [<https://codahale.com/how-to-safely-store-a-password>]