YJBC2.0 存证系统说明书

南京元几科技有限公司
www.yuanji.tech

联系人: 刘先生
电话: 18551717618 (同微信)
版本: v1.0.0

2024 年 01 月 18 日


元几科技
www.yuanji.tech

目录

第1章　系统概述

区块链存证系统是一种基于区块链技术的电子数据存储和验证系统。它的目标是实现去中心化的社会结构，保障数据的不可篡改性和可靠性。以下是关于区块链存证系统的目标、功能、特点、原理和操作环境的详细介绍：

1.1　目标

区块链存证系统的目标是实现去中心化的社会结构，通过区块链技术打破传统的中心化存储模式，将数据的所属权归还给用户，确保数据的真实性和不可篡改性。

1.2　功能

区块链存证系统具备多种功能，包括数据存储、数据验证、数据追溯等。它能够将用户上传的数据进行哈希运算和加密处理，并将数据存储在区块链上，保证数据的真实性和不可篡改性。同时，它还能够通过追溯服务，帮助用户追溯数据的源头和流转过程，验证数据的真实性和完整性。此外，区块链存证系统还支持数据的共享和授权功能，方便用户对数据进行分发和使用。

1.3　特点

区块链存证系统具有去中心化、不可篡改、全程留痕、可以追溯、集体维护、公开透明等特点。它通过多台计算机节点共同验证并记录交易数据，确保数据的真实性和完整性。存储在区块链上的数据不仅可以实现永久保存，并且不受任何个人或单个组织的控制，保护了数据的隐私、安全去中心化等特性。

1.4 原理

区块链存证系统的原理是利用区块链技术实现数据的不可篡改性和可靠性。区块链是一种去中心化的分布式账本技术，通过多台计算机节点共同验证并记录交易数据，确保数据的真实性和完整性。存储在区块链上的数据可以被打包成一个区块，每个区块都包含了前一个区块的哈希值，使得每个区块都与前面的区块紧密相连，成为一个不可修改的链条。当数据被存储在区块链上时，就可以通过区块链的共识机制实现数据的验证和认证，确保数据的真实性和可信度。

1.5 操作环境

YJBC2.0 区块链存证系统的操作环境主要包括硬件设备和软件环境两部分。硬件设备包括服务器、网络设备、存储设备等，需要具备高性能、稳定可靠和安全防护等特点。软件环境包括操作系统、数据库、开发工具等，需要支持区块链技术的运行和开发。同时，还需要建立完善的安全管理体系和技术防范措施，确保系统的安全和稳定。总之，区块链存证系统是一种基于区块链技术的电子数据存储和验证系统，具有去中心化、不可篡改、全程留痕等特点。它可以应用于各个领域，如证书认证、版权保护、司法诉讼等，为数据的保护和安全提供了一种新的解决方案。

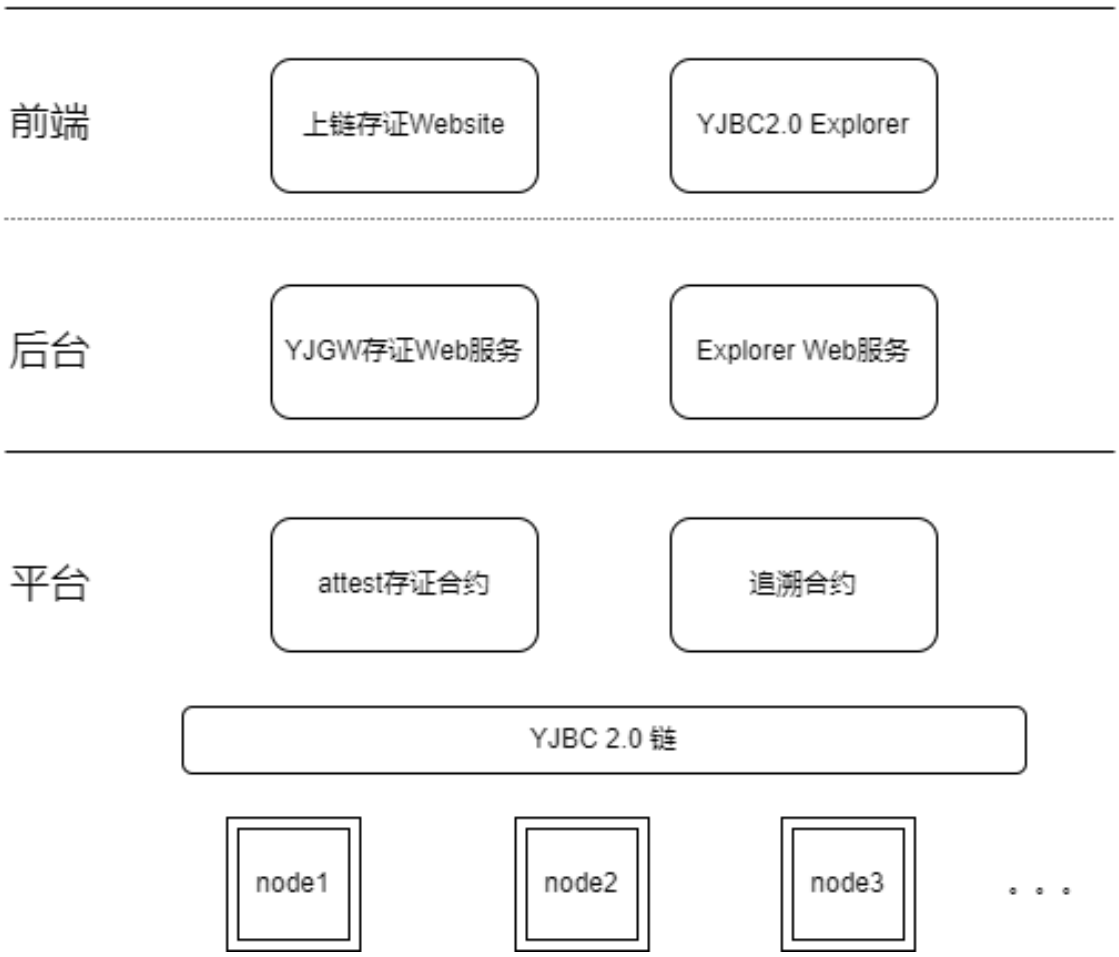

图 **1.1:** 架构图

前端是用户与系统交互的界面，用户通过前端可以上传数据、查询数据和下载数据等操作。前端通常包括网页、APP 等形式，提供友好的用户界面和交互体验。

后台是系统的核心部分，包括应用交互层、数据层、区块链层等。后台主要负责处理用户请求、存储和管理数据、维护区块链网络等功能。

平台通常指整个区块链存证系统，包括前端和后台的所有组成部分。平台提供完整的区块链存证服务，包括数据上传、存储、查询、验证等功能。

前端、后台和平台之间的关系是相互依存的。前端和后台都需要平台提供技术支持和基础设施，平台也需要前端和后台的配合才能实现完整的功能。同时，前端和后台也需要相互协作，共同完成用户请求和处理业务流程。

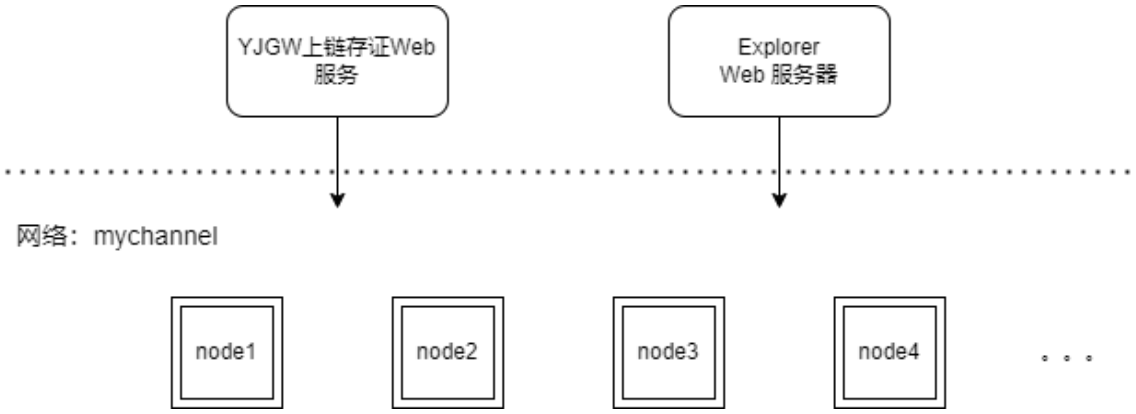

图 **1.2:** 消息结构图

1.6 用户登录

介绍如何通过用户界面进行登录和身份验证的步骤，包括用户名和密码的输入、身份验证等方面的信息。

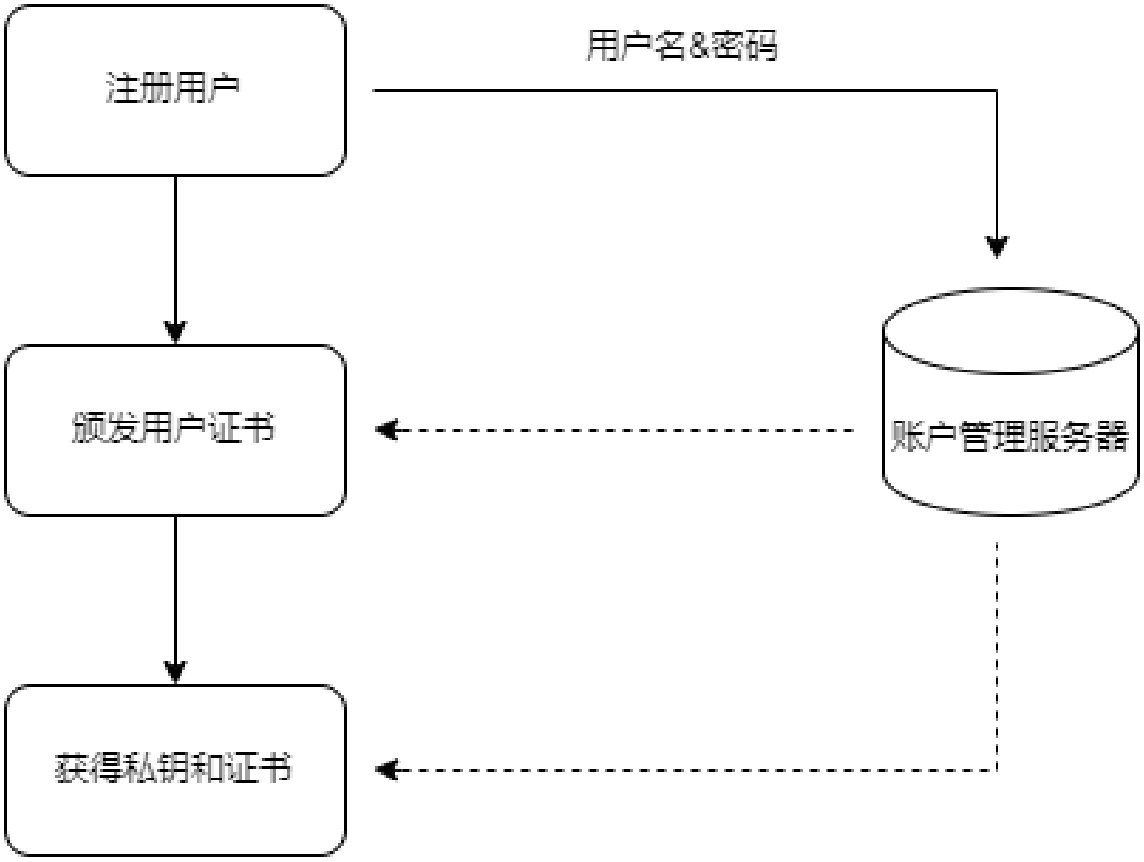

图 **1.3:** 开户服务图

在区块链存证系统中，用户需要先进行注册，然后才能获得用户证书。通常，用户证书由链管理机构颁发，用户需要提交必要的信息和资料，并经过审核后才能获得用户证

书。用户证书是用户在区块链存证系统中的身份凭证，用于证明用户的身份和权益。

除了用户证书外，用户还需要生成私钥。私钥是用户的唯一标识符，用于加密和解密数据，保证用户数据的安全性和完整性。用户需要妥善保管自己的私钥，不要轻易泄露给他人，以免造成不必要的损失。

在获得用户证书和私钥后，用户就可以开始使用区块链存证系统的各项功能了。需要注意的是，用户证书和私钥是用户的唯一身份标识，用户需要保护好自己的私钥和证书，不要轻易泄露给他人。同时，区块链存证系统也需要保证用户信息和数据的隐私和安全，避免用户的个人信息和数据被泄露或滥用。

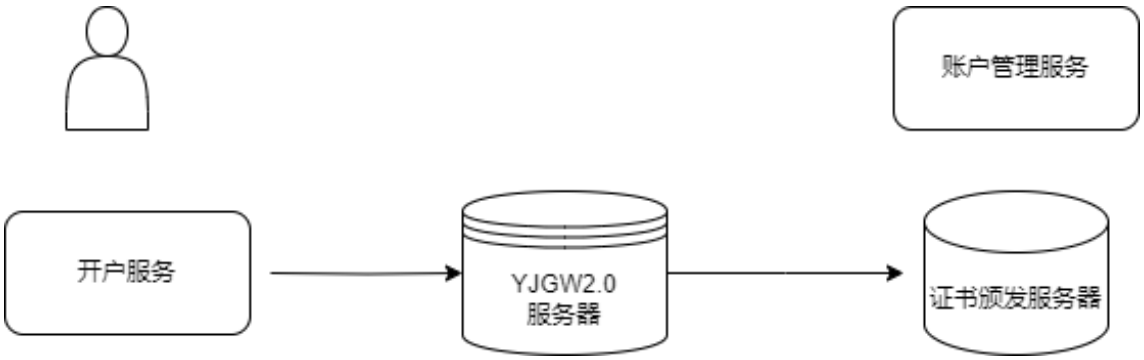

图 **1.4:** 前端调用-开户服务图

链服务器：链服务器是区块链存证系统的重要组成部分，负责维护和管理区块链网络。链服务器会同步区块链数据，验证交易和区块的合法性，确保区块链的安全性和可靠性。同时，链服务器还提供 API 接口，供前端和后台进行数据交互和查询。

证书颁发服务器：证书颁发服务器用于生成和颁发数字证书，用于验证用户的身份和授权。当用户注册或激活账户时，证书颁发服务器会为该用户生成数字证书，并存储在区块链上。用户在使用区块链存证系统时，需要提供数字证书进行身份验证，确保只有合法的用户才能访问和使用系统。

账户管理服务：账户管理服务负责对用户账户进行管理和维护，包括用户登录、密码修改、个人信息更新等功能。账户管理服务会对用户提交的请求进行验证和处理，确保账户的安全性和可靠性。同时，账户管理服务还会根据用户的需求和权限，提供相应的账户设置和个性化服务。

1.7 数据上传

介绍如何将需要存证的数据上传至区块链存证系统的步骤，包括数据选择、文件上传、哈希值计算等方面的信息。

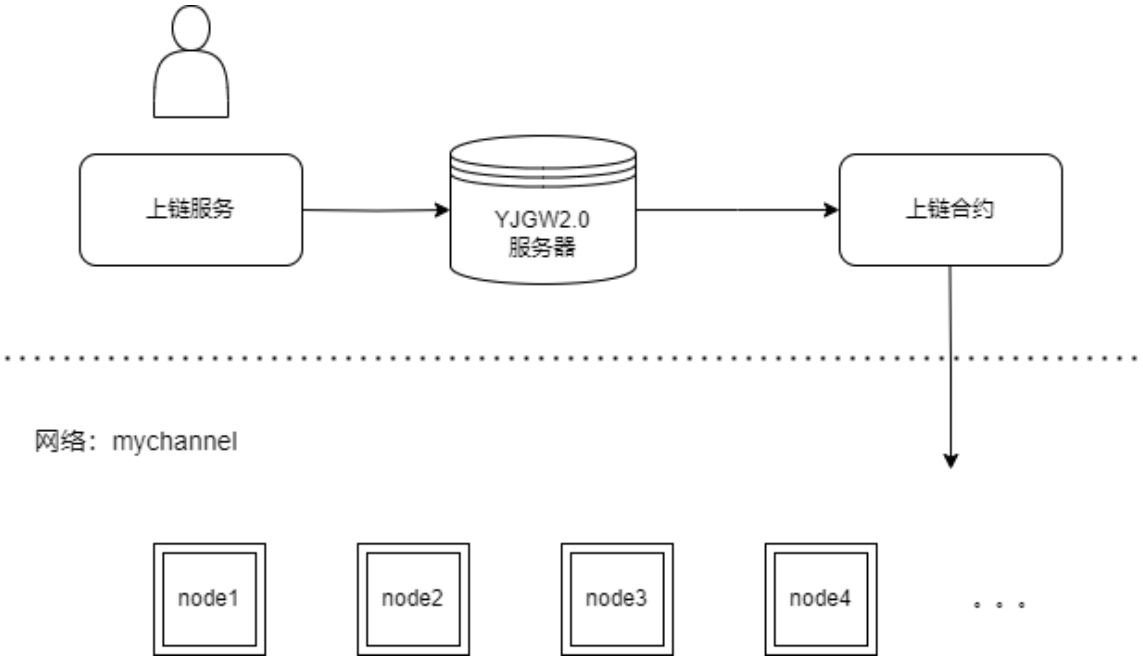

图 **1.5:** 前端调用-上链服务图

区块链存证系统上链服务主要包括以下步骤：

链下数据采集：首先需要采集需要上链的数据，并进行哈希运算，生成数据的数字指纹。

节点验证：将生成的数据指纹广播到区块链网络中的各个节点，节点会对数据进行验证，确保数据的真实性和完整性。

区块打包：经过节点验证的数据会被打包成区块，并添加到区块链中。

链上存储：完成区块打包后，数据将被存储在区块链上，形成一个不可篡改的数据记录。在区块链存证系统中，链服务器是用于支持区块链网络的分布式计算服务器。它将数据存储在多台服务器上，实现数据的分布式存储和处理，从而提高数据处理效率，降低单点故障的风险。

上链合约是将合约在区块链上完成存证过程的一种技术。它通过将合约内容进行哈希运算，并将哈希值存储在区块链上，实现合约的不可篡改性和时间戳记录。上链合约具有数据可追溯、不可篡改、多方参与的特点，为解决传统合同存证不安全、不真实、被破坏丢失等业务痛点提供了技术手段。

1.8 数据存储

介绍如何将上传的数据存储至区块链网络的步骤，包括数据打包、交易发送等方面的信息。

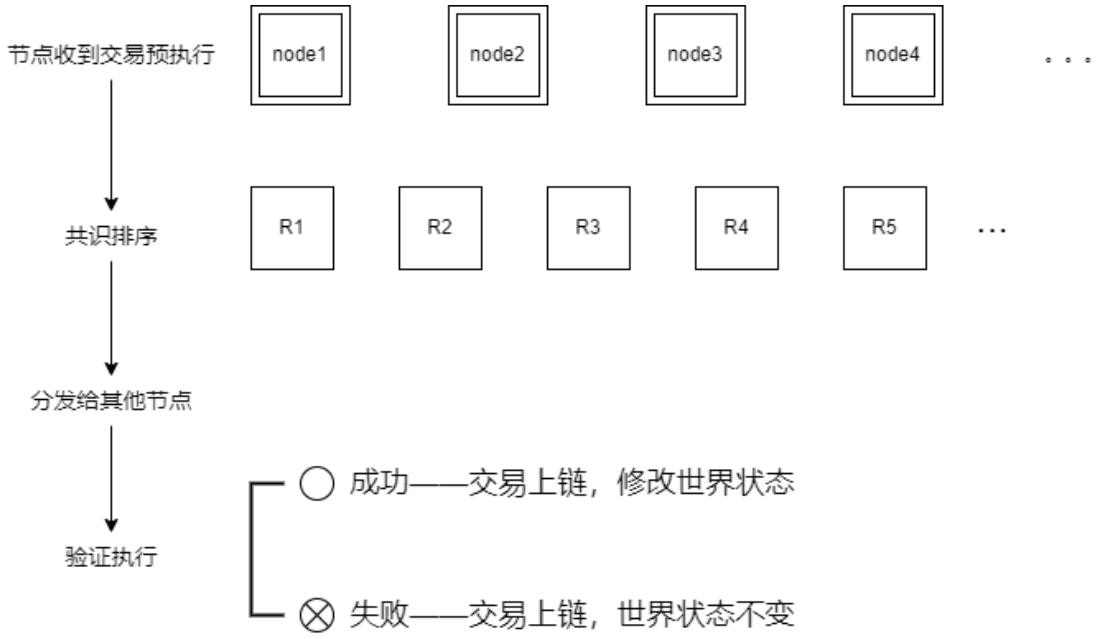

图 **1.6:** 新建执行流程图

区块链存证系统的节点收到交易预执行后，会进行共识排序，将交易按照一定的顺序进行排列。然后，节点会将排序后的交易分发给其他节点，进行验证和执行。

具体来说，节点在收到交易预执行请求后，会先对交易进行验证，确保交易的有效性和合法性。如果交易验证通过，节点会将交易加入到本地的交易池中。

接着，节点会根据一定的共识算法对交易进行排序。常见的共识算法包括工作量证明（Proof of Work）、权益证明（Proof of Stake）等。节点会选择一定数量的交易作为区块的候选交易，并根据共识算法确定这些交易的顺序。

一旦共识达成，节点会将排好序的交易分发给其他节点进行验证。其他节点会对收到的交易进行再次验证，并用自己的私钥签名确认。如果大多数节点对同一交易签名确认，该交易就会被添加到区块链上。

最后，节点会执行交易，更新本地账本的状态。如果交易涉及到智能合约的执行，节点还会调用智能合约的代码，完成相应的业务逻辑处理。

通过以上步骤，区块链存证系统实现了去中心化的交易处理和数据存储，保证了数据的安全性和可信度。同时，通过共识算法和节点间的相互验证，保证了数据的真实性和不可篡改性。

1.9 数据查询与检索

介绍如何通过用户界面进行数据查询和检索的步骤，包括数据筛选、结果展示等方面的信息。

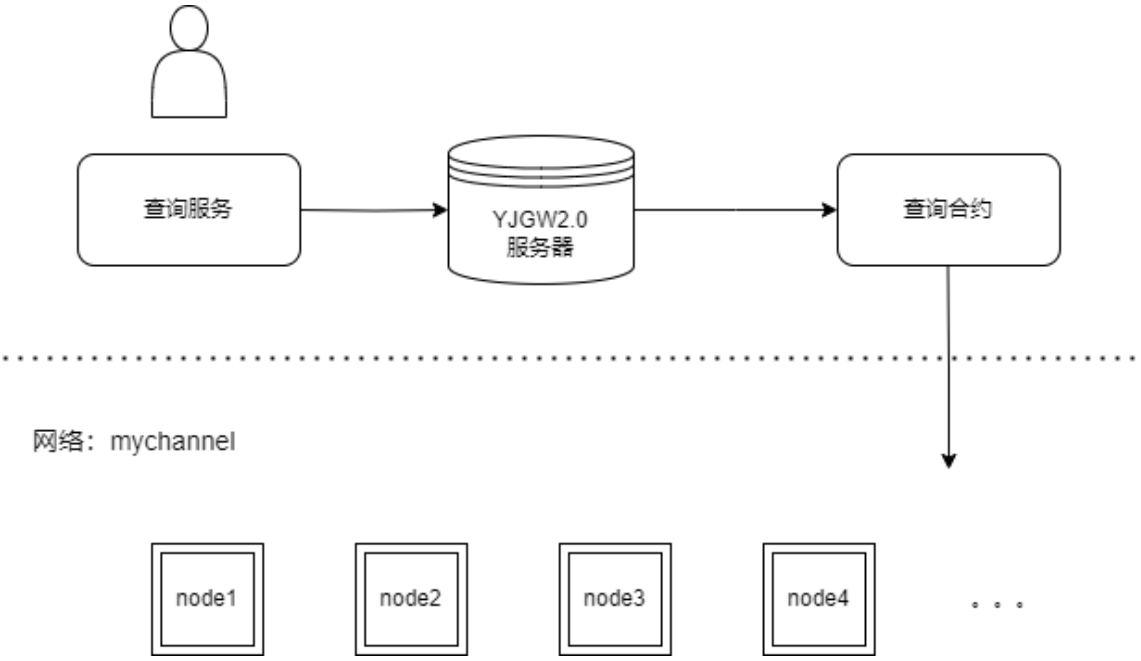

图 **1.7:** 前端调用-查询服务图

区块链存证系统的查询服务主要包括以下步骤：

用户发出查询请求：用户通过系统提供的界面或 API 接口发出查询请求，需要查询的数据包括上链的数据和链下的数据。

链服务器处理查询请求：链服务器接收到查询请求后，会根据请求的类型和内容进行相应的处理。如果查询的是链上的数据，链服务器会通过查询合约来获取数据；如果查询的是链下的数据，链服务器则需要与数据提供方进行交互，获取数据。

查询合约返回数据：查询合约是部署在区块链上的智能合约，用于存储和管理上链的数据。当链服务器向查询合约发出查询请求时，查询合约会根据请求的内容返回相应的数据。

链服务器返回数据给用户：链服务器将获取到的数据返回给用户，用户可以查看、下载或使用这些数据。

数据验证与确权：在查询过程中，为了确保数据的真实性和完整性，需要对数据进行验证和确权。验证是指对数据进行哈希运算，并与区块链上存储的哈希值进行比对，以确保数据没有被篡改；确权是指根据用户的身份信息，验证用户是否有权访问和查询这些数据。

在整个查询过程中，区块链网络的作用是确保数据的不可篡改性和去中心化。区块链网络中的每个节点都会参与数据的验证和打包过程，确保上链的数据是真实可信的。同时，由于区块链的去中心化特性，即使部分节点发生故障或被攻击，也不会影响整个系统的稳定性和可用性。

综上所述，区块链存证系统的查询服务通过链服务器、查询合约和区块链网络等技术手段，实现了数据的快速、安全、可信的查询。同时，通过数据验证与确权等手段，确保了数据的真实性和完整性，为用户提供了可靠的存证服务。

1.10 数据验证

介绍如何验证存储的数据的真实性和完整性的步骤，包括哈希值计算、数据比对等方面的信息。

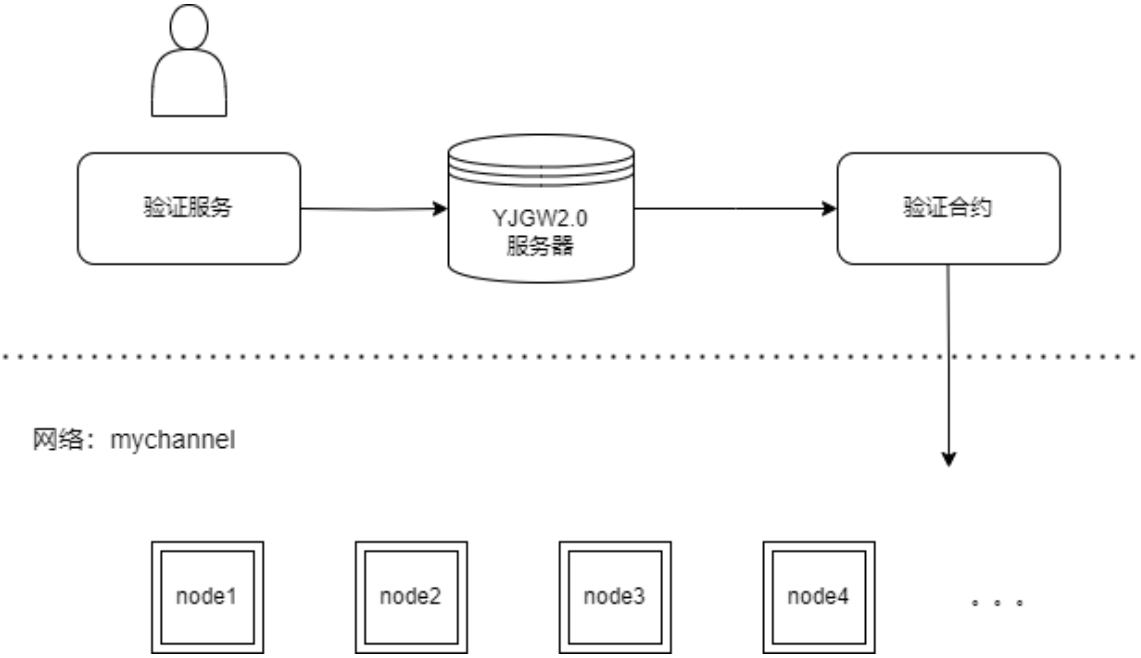

图 **1.8:** 前端调用-验证服务图

区块链存证系统的验证服务主要包括以下步骤：

数据验证：用户上传的数据需要进行哈希运算，并将生成的哈希值与区块链上存储的哈希值进行比对，以验证数据的真实性和完整性。

节点验证：区块链网络中的节点会对数据进行验证，确保数据的有效性和合法性。节点会对数据进行再次哈希运算，并与区块链上存储的哈希值进行比对，以验证数据的真实性和不可篡改性。

合约验证：如果数据涉及到智能合约的执行，节点会调用智能合约的代码，完成相应的业务逻辑处理。合约的代码也需要进行验证，确保合约的正确性和安全性。

链服务器验证：链服务器会对整个区块链网络进行监控和管理，确保数据的存储和处理符合安全和可靠的要求。链服务器还会对节点的行为进行监测，确保节点行为的合规性和可信度。

区块链网络验证：区块链网络本身也需要进行验证，确保网络的去中心化、安全性和可靠性。区块链网络的共识算法和加密算法也需要进行验证，确保算法的安全性和可信度。

总的来说，区块链存证系统的验证服务通过数据验证、节点验证、合约验证、链服务器验证和区块链网络验证等步骤，确保数据的真实性和不可篡改性，以及智能合约的正确性和安全性。这些验证服务为解决业务痛点提供了技术保障。

1.11 追溯服务

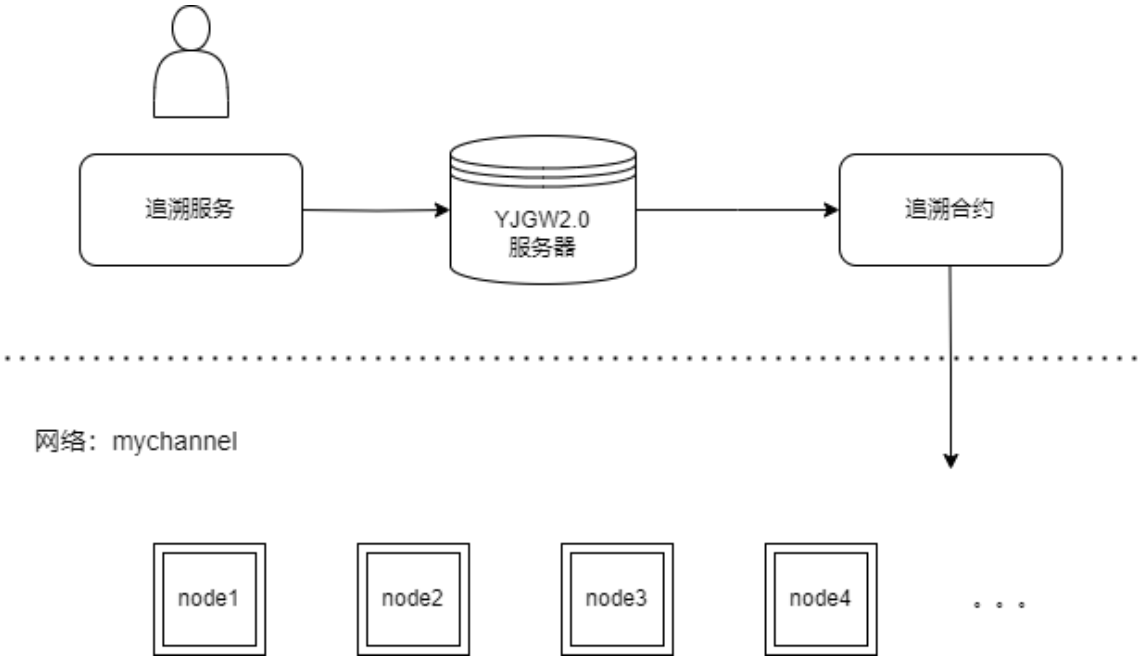

图 **1.9:** 前端调用-追溯服务图

区块链存证系统的追溯服务主要利用区块链技术的不可篡改性和时间戳记录特点，实现数据和交易的追溯和验证。以下是相关步骤：

数据和交易的写入：用户在区块链存证系统中进行数据和交易操作时，这些数据和交易会被写入区块链网络中。

数据和交易的存储：区块链网络中的节点会对这些数据和交易进行存储，并保证数据和交易的真实性和完整性。

数据和交易的追溯：利用区块链网络中的时间戳记录和哈希值验证机制，可以对数据和交易进行追溯。通过比对区块链上存储的数据和交易的哈希值，可以追溯到数据和交易的源头，验证其真实性和完整性。

链服务器支持：链服务器会对整个区块链网络进行监控和管理，提供数据和交易的存

储和处理服务。同时，链服务器也会对追溯服务提供支持，帮助用户快速定位到需要追溯的数据和交易。

追溯合约执行：如果数据和交易涉及到智能合约的执行，追溯合约也会被调用。追溯合约会对智能合约的代码进行验证，确保合约的正确性和安全性。同时，追溯合约还会记录智能合约的执行过程，以便后续的追溯和验证。

区块链网络验证：区块链网络本身也需要进行验证，确保网络的去中心化、安全性和可靠性。区块链网络的共识算法和加密算法也需要进行验证，确保算法的安全性和可信度。

通过以上步骤，区块链存证系统的追溯服务实现了数据和交易的可追溯性和验证性。这有助于解决传统追溯方式中存在的数据易篡改、不易追溯等问题，提高了追溯服务的可信度和可靠性。

第 2 章　YJBC2.0 系统参数说明

2.1　系统环境配置部分参数

1. cpu 体系架构: x86_64
2. 内存: 32GB(及以上，推荐 64GB)
3. 硬盘: 1TB(及以上, 视具体实际业务而定)
4. 操作系统:ubuntu 22.04 LTS (操作系统安装时，把默认用户设为 vi)
5. docker:Docker version 24.0.5, build 24.0.5-0ubuntu~22.04.1
6. docker-compose: version 1.29.2
7. gcc:12
8. node:14; 16; 18
9. mongodb:v7
10. mongodb-c-driver:1.25
11. mongodb-cxx-driver:3.9

2.2　系统文件说明

相关文件到下面的链接上下载。

http://www.igcc.cc:9900/download/yjbc/

相关文件说明:

1. env_config.tar.gz 包含项目依赖的第三方软件及开发库工具等
2. images.tar 链节点的 docker 镜像文件

3. yjweb.tar.gz 链网关 ts 部分

4. ais.tar.gz yjbc2.0 存证服务器后台 + 开发调试界面

5. cc.tar.gz 存证相关的智能合约

6. src_explorer.tar.gz yjbc2.0 区块链浏览器

7. opt.tar.gz 链依赖的第三方软件

8. shell.tar.gz 系统开发过程中产生的脚本

第 3 章　接口 API 参数说明

YJBC2.0 存证服务所有 API 接口都以 http 协议对应用业务系统提供。主要包含了: 存证上链服务，存证查询服务，数据核验，存证历史追溯服务，用户管理服务。

为方便描述，本文档以开发环境 http://igcc.cc:8081 为示例。实际应用中，用户需要根据实际地址替换。

3.1　上链服务

接口路径:/attest/write

http 方法:POST

http 头:'Authorization:0bf886a5-7c7b-4595-9c07-f5f39dd9b595' Authorization 值为用户登录后获得的 token, 见用户登录接口。

参数说明: 参数格式为 json, 其中需要包含 data 字段 (必选), 为需要上链的存证的数据; message 字段 (可选) 表示数据说明。

返回数据说明:

attest.id: 为用户数据存证后返回的唯一 id，用户的业务系统需要将此 id 保存并与上链数据建立映射关系，后续查询，数据核验接口都需要提交此 id 字段。

attest.fingerprint: 为用户数据指纹，些指纹存于区块链上，不可篡改。

attest.ctime: 存证发生的时间。

attest.memo: 描述备用字段, 一般为空。

transaction_id: 此存证链上交易号, 可以在 YJBC2.0 区块链浏览器按此交易号查找查看此交易详情。

使用命令行 curl 调用示例:

```
curl http://igcc.cc:8081/attest/write -H 'Authorization:0bf886a5-7
   c7b-4595-9c07-f5f39dd9b595' -X POST -d '{"data":"hello","message
   ":"attest write request from curl"}'
```

```
调用成功后返回:
{
 "attest": {
  "id": "590a62ed-fda5-4f94-b421-785dc28aeff1",
  "fingerprint": "
     e84de5fa08ec31982b429ce8e53f820c2cd1f53f1ddc6ea51711128ad0ef3a81635e62d7
     ",
  "ctime": "2024-01-16 20:26:35.678",
  "memo": ""
 },
 "transaction_id": "
    eed693351b543fe6748bb5445f2f6ad677798d3f6b3f992e5cd9cdeca1d38369
    "
}
```

3.2 存证查询服务

接口路径:/attest/read/attest_id

http 方法:GET

参数说明: 其中 url 路径最后的 attest_id 字段 (必选), 为需要查询的存证的 id, 业务调用需要用实际的 id 替换上述接口路径中的 attest_id

返回数据说明:

attest.id: 为用户数据存证后返回的唯一 id，用户的业务系统需要将此 id 保存并与上链数据建立映射关系，后续查询，数据核验接口都需要提交此 id 字段。

attest.fingerprint: 为用户数据指纹，些指纹存于区块链上，不可篡改。

attest.ctime: 存证发生的时间。

attest.memo: 描述备用字段, 一般为空。

transaction_id: 此存证链上交易号, 可以在 YJBC2.0 区块链浏览器按此交易号查找查看此交易详情。

使用命令行 curl 调用示例:

```
curl http://igcc.cc:8081/attest/read/590a62ed-fda5-4f94-b421-785
   dc28aeff1 -H 'Authorization:0bf886a5-7c7b-4595-9c07-f5f39dd9b595
   '
```

调用成功后返回:


```
{
 "message": "read attest succeed",
 "attest": {
  "ctime": "2024-01-16 20:26:35.678",
  "docType": "attest",
  "fingerprint": "
     e84de5fa08ec31982b429ce8e53f820c2cd1f53f1ddc6ea51711128ad0ef3a81635e62d7
     ",
  "id": "590a62ed-fda5-4f94-b421-785dc28aeff1",
  "memo": ""
 },
 "verison": "tsweb v1.0.0"
}
```


3.3 数据核验服务

接口路径:/attest/verify

http 方法:POST

参数说明:

参数格式为 json, 其中 attest_id 字段 (必选): 表示与业务数据映射的存证 id

data 字段 (必选), 为需要核验的已上链存证的数据

返回数据说明:

message: 表示如果核验过程执行了，即为”verify ok”。如果因为系统故障未能执行核验结果，那么返回”verify failed”

origin: true, 表示所需核验的数据与上链存证时一致，未被修改过。false, 表示数据已被修改过。

status: 核验过程被执行了则为 200，否则为其它错误号：500 或 400 等。

使用命令行 curl 调用示例 (核验通过):

```
curl http://igcc.cc:8081/attest/verify -H 'Authorization:0bf886a5-7
   c7b-4595-9c07-f5f39dd9b595' -s -X POST -d '{"attest_id":"590
   a62ed-fda5-4f94-b421-785dc28aeff1","data":"hello"}'
```

调用成功,且数据经核验与原始上链时一致，未被修改过，则后返回(origin为true):

```
{"message":"verify ok","origin":"true","status":"200"}
```

使用命令行 curl 调用示例 (核验未通过, 数据已与上链时不一致):

```
curl http://igcc.cc:8081/attest/verify -H 'Authorization:0bf886a5-7
   c7b-4595-9c07-f5f39dd9b595' -s -X POST -d '{"attest_id":"590
   a62ed-fda5-4f94-b421-785dc28aeff1","data":"hell1"}'
```

调用成功,且数据经核验与原始上链时一致，未被修改过，则后返回(origin为false):

```
{"message":"verify ok","origin":"false","status":"200"}
```

3.4 用户开户注册服务

接口路径:/attest/user_register

http 方法:POST

参数说明:

参数格式为 json, 其中 user_name 字段 (必选): 表示用户需要注册的用户名。

passwd 字段 (必选), 为用户设置的密码。

返回数据说明:

message: 描述注册执行过程与结果情况。

ctime: 注册发生时间.

passwd_sha384: 密码经过哈稀后的字符串, 后台不保存用户原来的密码。

user_name: 注册的用户名。

memo: 描述备用字段, 一般为空。

使用命令行 curl 调用示例:

```
curl -v http://igcc.cc:8081/attest/user_register -X POST -d '{"
   user_name":"tom", "passwd":"123"}'
```

```
ok respond:

{"ctime":"2024-01-21 23:27:15.615","id":"09a3f727-9066-4e7d-9091-
   e4d02e570933","memo":"","message":"user_register succeed from c
   ++ web service","passwd_sha384":"9
   bd942d1678a25d029b114306f5e1dae49fe8abeeacd03cfab0f156aa2e363c988b1c12803d
   ","user_name":"tom"}

failed respond:

{"ctime":"2024-01-21 23:37:12.899","id":"04c9cc50-d8c4-4798-aea5-
   be4c309e9f7b","memo":"","message":"user_register failed, user
   already exist, user:tom1","passwd_sha384":"9
   bd942d1678a25d029b114306f5e1dae49fe8abeeacd03cfab0f156aa2e363c988b1c12803d
   ","status":"400","user_name":"tom1"}
```

3.5 用户登录服务

接口路径:/attest/user_login

http 方法:POST

参数说明:

参数格式为 json, 其中 user_name 字段 (必选): 表示用户需要注册的用户名。

passwd 字段 (必选), 为用户的密码。

返回数据说明:

message: 描述登录执行过程与结果情况。

ctime: 登录发生时间.

status: 登录结果状态码。

token: 与当前用户登录关联的 token，存证服务需要将此值设置为 http 头中 Authorization 的值。以表示当前用户已登录被授权。

使用命令行 curl 调用示例:

```

```

```
curl http://igcc.cc:8081/attest/user_login -XPOST -d '{"user_name
   ":"tom","passwd":"123"}'

{"ctime":"2024-01-23 00:50:16.116","message":"user_login succeed
   from c++ web service","status":"200","token":"b1e5fec8-0621-4e93
   -91ce-a02276b6abcf","user_name":""}

if passwd is not right, respond:

{"ctime":"2024-01-23 00:53:04.909","message":"user_login failed,
   passwd not right, user:","status":"401","user_name":""}
```

3.6 用户退出服务

接口路径:/attest/user_logout

http 方法:POST

参数说明:

参数格式为 json, 其中 user_name 字段 (可选): 表示用户需要注册的用户名。

token 字段 (必选), 为用户登录时获得的 token。

返回数据说明:

message: 描述退出过程与结果情况。

ctime: 退出发生时间.

使用命令行 curl 调用示例:

```
curl http://igcc.cc:8081/attest/user_logout -XPOST -d '{"user_name
   ":"tom","token":"b1e5fec8-0621-4e93-91ce-a02276b6abcf"}'

{"ctime":"2024-01-23 00:51:30.166","message":"user_logout succeed
   from c++ web service"}
```