北京金融科技产业联盟
BEIJING FINTECH INDUSTRY ALLIANCE

金融业隐私计算互联互通技术研究报告

北京金融科技产业联盟
2023 年 5 月

版权声明

编委会名单

主　　任：潘润红

顾　　问：柴洪峰

编委会成员：涂晓军　何　军　聂丽琴　杨燕明　傅　杰　周　骏
冯云青　丁文定　范力欣　田　江　张　野　彭　晋
王　磊　王　超　姚　明　何　浩　卞　阳　周吉文

编写组成员：高鹏飞　周雍恺　李定洲　张远健　王思婷　葛明嵩
焦惠芸　王梦鸽　龚乐诚　刘　瑞　魏博言　张锦元
许　冠　范　涛　万志辉　邓　凯　王　鹏　樊昕晔
李松涛　尤　萌　黄雅琼　张晓蒙　肖俊贤　余超凡
昌文婷　袁鹏程　胡晓龙　王　玥　曾　成　靳　新
赵永坤　杨天雅　卫　骞　陈治宇　于　欢　何　朔
王　琪　丁亚丹　陈钟正　袁　航　邹　奋　张高磊
张　乐　高　扬　陶建萍　郑培钿　毛　娟　鲁　欣
张大健　王华忠　虞　洋　赵　可　任江哲　王湾湾
张严明　李京贤　孙　赫　国　钰　孙　舒　徐安滢
司忠平　张少敏　黄　璜　张翼飞　刘玉良　钱　菲
谢　谨　张育涵　刘　威　薛祥杰　黄　飞　臧　铖
陈嘉俊　张敬之　曹旭涛　董效稳　陈志豪　王心玥
胡师阳　邱晓慧　杨　波　王　雪　李武璐　白玉真
袁　博　闫　树　宋雨筱　陈浩栋　刘　尧　金银玉
裴　超　韦晓亚　陈　鑫　蒋嘉琦　李亚鹏　王云河

靳　晨　陈　璐　陈　琨　庞　婷　魏　刚　韩　鹏
李秉帅　李克鹏　陈　明　王礼斌　王健宗　黄章成
吴天博　黄翠婷　陈　涛　卞　杰　崔　琢　章　庆
朱　瑶　张鸣皓　肖　坤　杨朋霖　吴　杰　叶家炜
应志伟　冯　浩　王帅强　鲍思佳　李如先　陈　曦
张　剑　张佳辰　金良水　于　博　顾逸晖　蒋美献
赵　荣　苗天麒　王　寰　吴叶国　马　利　盛钟滨
刘　伟　宁立君　唐志徽　阮安邦　魏　明　王铀之
刘立强　赖建章　邓志强　杨劲雄　周　璇　李　帅
徐　静　李延凯　吴博峰　杨　博

编　　审： 黄本涛　张远健　刘宝龙

主编单位：

北京金融科技产业联盟秘书处

中国银联股份有限公司

成方金融信息技术服务有限公司

招商银行股份有限公司

上海浦东发展银行股份有限公司

中国工商银行股份有限公司

深圳前海微众银行股份有限公司

光大科技有限公司

中金金融认证中心有限公司

蚂蚁科技集团股份有限公司

蓝象智联（杭州）科技有限公司

深圳市洞见智慧科技有限公司

上海富数科技有限公司

北京百度网讯科技有限公司

参编单位：

中国农业银行股份有限公司

中国银行股份有限公司

交通银行股份有限公司

中国民生银行股份有限公司

兴业银行股份有限公司

浙商银行股份有限公司

华夏银行股份有限公司

中国邮政储蓄银行股份有限公司

北京银联金卡科技有限公司（银行卡检测中心）

建信金融科技有限责任公司

中国信息通信研究院

北京冲量在线科技有限公司

北京数牍科技有限公司

度小满科技（北京）有限公司

华控清交信息科技（北京）有限公司

华为技术有限公司

腾讯云计算（北京）有限责任公司

深圳壹账通智能科技有限公司

同盾科技有限公司

中国电信股份有限公司

中国移动股份有限公司

复旦大学

海光信息技术股份有限公司

金融信息化研究所

联易融数字科技集团有限公司

上海光之树科技有限公司

神谱科技（上海）有限公司

深圳致星科技有限公司

深圳微言科技有限责任公司

神州融安数字科技（北京）有限公司

北京八分量信息科技有限公司

上海荣数信息技术有限公司

银联商务股份有限公司

杭州趣链科技有限公司

北京原语科技有限公司

北京火山引擎科技有限公司

京东科技信息技术有限公司

目 录

前 言

2022 年 12 月 19 日，中共中央国务院正式发布《关于构建数据基础制度更好发挥数据要素作用的意见》[1]，提出要促进数据合规高效流通使用，充分实现数据要素价值，并支持有条件的部门、行业加快突破数据可信流通、安全治理等关键技术。数据要素的发展催生了隐私计算的繁荣，随着近几年隐私计算行业的快速兴起，联邦学习、多方安全计算、可信执行环境等隐私计算技术逐步从试点走向了商用。然而，各家厂商的隐私计算平台往往根据各自需要采用了不同的技术架构和算法协议，使得各平台实现具有极大的差异性，导致了各平台间难以直接互联互通。这种情况不仅造成各机构厂商系统的重复建设和运营成本的浪费，还间接形成了数据要素流通的技术壁垒，使隐私计算连接的“数据孤岛”开始转变成“技术孤岛”。

隐私计算跨平台互联互通能够有效解决“技术孤岛”问题，有助于促进数据要素跨平台高效流通。为进一步加快隐私计算互联互通技术发展，推动行业级隐私计算互联互通方案及实施路线制定，在北京金融科技产业联盟（简称“金科联盟”）指导下，中国银联联合商业银行在内的主要金融机构、科技公司、互联网公司、电信运营商、隐私计算开源社区和检测认证机构共同启动了隐私计算互联互通（以下简称“互联互通”）的课题工作。此次课题的组织，旨在完成行业级互联互通框架研制、关键技术研究及规范编制工作，本报告内容属于课题阶段性研究成果的汇总体现。

本报告立足于行业视角，针对隐私计算技术发展的趋势和需求，围绕“互联互通”开展一系列的探索和实践，并深入浅出地进行阐释。报告从当前行业发展现状和互联互通的价值意义着手，重点提出行业级隐私计算互联互通探索实践的思路，致力推动可行的跨平台互联互通框架组成研究，以及深入挖掘互联互通关键技术点及配套落地实施方案，并在此基础上进一步展望互联互通产业发展新生态和商业模式，以期更好地激发数据要素流通和算法市场新活力，塑造我国数字经济时代新的竞争优势。

总体而言，报告关于“隐私计算互联互通课题”的探讨具有重要的现实意义。面对金融行业严监管要求和实际应用中复杂多样的平台技术实现方案，形成覆盖管理面、数据面的行业级互联互通统一框架，以更好实现跨数据、跨平台、跨行业互联互通；此外，报告还针对互联互通生态发展需要，分别从技术、标准、业务等方面提出推进思路，以推动金融行业互联互通生态构建和落地，助力形成隐私计算互联互通生态流通网络。

一、隐私计算互联互通当前发展情况

（一）发展背景及意义

1. 背景及现状

数据作为数字经济的核心生产要素之一，已在国家、行业等多层面获得了广泛重视。中共第十九届中央委员会第四次全体会议于 2019 年 11 月 5 日发布的《关于坚持和完善中国特色社会主义制度推进国家治理体系和治理能力现代化若干重大问题的决定》[2]中首次提出将数据列为一种新型生产要素。2022 年 12 月 19 日，中共中央国务院新发布的《关于构建数据基础制度更好发挥数据要素作用的意见》，进一步提出促进数据合规高效流通使用，充分实现数据要素价值，并支持有条件的部门、行业加快突破数据可信流通、安全治理等关键技术。金融行业是典型的数据密集型行业，数据要素正成为金融行业高质量发展的重要驱动力。中国人民银行于 2021 年 12 月 29 日发布的《金融科技发展规划（2022-2025 年）》[3]中提出了推进数据有序共享，要求探索建立跨主体数据安全共享隐私计算平台，提升数据要素资源配置效率。

当前金融行业关于数据安全及隐私保护的需要加快推动了隐私计算技术在金融行业的创新应用。随着近几年隐私计算行业的快速发展，联邦学习、多方安全计算、可信执行环境等隐私计算技术逐步从试点走向了商用。然而，各家厂商的隐私计算平台在早期实现的过程中往往采用了不同的技术架构和算法协议，这使得各平台实现具有极大的差异性，导致了各平台间难以直接互联互通。由于各机构异构平台间无法直接进行交互协同，

且各机构已接入的数据源基本仅能在特定项目、特定平台下使用，导致金融机构现阶段需要重复部署多套异构隐私计算平台来实现跨机构、跨平台的数据合作。这不仅造成了系统重复建设和运营成本的浪费，还间接形成了数据要素流通的技术壁垒，使隐私计算连接的“数据孤岛”开始转变成为“技术孤岛”。

隐私计算互联互通技术在一定程度上能够促进“技术孤岛”联通，进而促进数据要素跨平台高效流通。当前跨平台互联互通已逐步成为行业新共识，业界各方正在从技术和应用层面推动隐私计算互联互通。

2. 互联互通的意义和价值

互联互通的发展能够更好推动隐私计算产业发展，充分激发数据要素价值。可以说，互联互通是隐私计算行业发展的必然趋势。只有真正实现不同行业机构、厂商平台间的互认互用，才能破除平台壁垒，加快数据要素生态市场建设。从现实意义上看，互联互通的意义和价值主要有：

（1）加快形成统一标准体系的金融数据要素流通网络。互联互通能够促进形成统一的隐私计算技术标准体系，有助于建立跨主体数据安全共享模式，充分挖掘数据价值，为实现跨机构、跨地域、跨行业数据资源有序共享和综合应用提供有效支撑。

（2）促进互联互通产业新生态和新商业模式构建。互联互通有助于实现产业各方各领域数据资源最大化利用，促进互联互通良性生态的形成和新商业模式构建，不断拓展金融行业数据要素广度和深度，并在

服务实体经济、助力金融科技发展的同时逐步提升数据综合应用与多向赋能实效。

（3）助力端到端数据流通体系监管落地。隐私计算互联互通可为监管机制的透明化、公开化落地提供可行的实现方式，助力收拢数据应用安全合规漏斗，将金融行业的数据流通实质性地纳入监管，进而推动数据要素市场的规范化发展。

（二）行业驱动力

随着隐私计算开始步入生态发展快车道，互联互通有助于进一步加快百亿级隐私计算市场建设。相比短期内较难实现、目标略显远大的意义和价值而言，产业各方更关注于互联互通能够快速带来的产业价值，这也是行业各方热衷于推进隐私计算互联互通的真正驱动力。

（1）降低成本、减少风险。隐私计算是一门新兴的技术，相关产品的成熟度大多未经过市场长期检验。因此，通过重复部署各类隐私计算平台来实现跨机构多场景平台支持的方式，不仅会带来运营管理成本的问题，其潜在的安全风险相比多年沉淀的成熟信息科技产品也要更高。各机构迫切希望加快异构平台互联互通落地，降低外部引入多个隐私计算平台带来的成本开销和安全风险，提升不同隐私计算平台间的协作效率。

（2）简化行业数据流通要求，加速跨行业数据合作。隐私计算是能够打破行业间数据壁垒的利器，互联互通则是充分发挥这把利器作用的磨刀石。互联互通的发展不仅有助于收敛行业数据要素流通要求、实

现行业数据价值释放的规范化与统一化，还有助于加快跨行业数据合作的落地与实现，使各隐私计算技术厂商可以更专注于互联互通接口细化和开放推广，促进隐私计算技术大规模应用，并进一步推动整个行业发展及产业繁荣，加快数据跨机构、跨行业流通利用。

（3）加快完善数据要素加速流通的政策环境。互联互通的落地，能够帮助各方进一步加快完善数据流通政策环境，建立统一的法律体系、制度体系、组织体系、标准体系和监管体系，促进数据要素的自由流动和合理配置，最终推动全国统一大数据市场的顺利构建。

（三）难点与挑战

互联互通的发展并不是一蹴而就，而是机会与挑战并存。要想真正实现各家机构、厂商隐私计算平台互通及数据要素顺畅流通流转，需要正视和解决当前互联互通存在的难点与挑战：

（1）技术理解和实现差异带来的技术改造困难。由于隐私计算行业尚处于初期，产业各方在认知理解上存在较大差异，在通信网络、密码算法、应用协议、上层接口、数据格式等各个层面的技术实现也差异较大。因此，要想实现互联互通，不仅需要有一套通用化框架以兼容适配各方的差异化平台和接口，还需要考虑未来互联互通后异构平台协同更新时迭代困难的问题。

（2）标准制度体系的缺位。当前业内标准规范体系尚未完备，不同隐私计算技术、不同业务场景、不同标准组织的标准间仍然相互割裂。在此情况下，如何设计更加灵活的互联互通标准和制度，使其既能兼容多个

异构平台，又能使得各异构平台互联互通具备可行性和可落地性，还能保持隐私计算在业务应用中的领先性和创造性，是当下落实互联互通工作的重点和难点。

（3）隐私计算的安全属性需要在互联互通设计中着重考虑。由于互联互通各参与方在安全设计理念、抗风险攻击能力方面存在较大差异，不同隐私计算技术路线之间、相同技术路线不同实现方案之间也存在一定的安全水平差异，因此，安全合规风险是目前隐私计算所面临的又一重大挑战。在尚未定义统一的隐私计算安全评定方式和分类分级尺度的当下，如何能够有效鉴别不安全、不适用的互联互通产品、算法和代码，如何保证隐私计算技术能够被安全、合规、高效地应用，以及如何有效平衡互联互通后隐私计算技术的安全性、效率和精度，是构建互联互通安全评价体系亟待考虑的问题。

（四） 金融行业探索实践

为赋能金融行业数据要素高效流通，提升隐私计算互联互通标准化程度，本小节从隐私计算技术发展需要及互联互通难点挑战出发，尝试探索定义行业级互联互通内涵和研究实现思路，以便更好推动隐私计算互联互通应用发展和标准制定工作。

1. 行业级互联互通的内涵

行业级隐私计算互联互通（见图1）是一整套实现异构隐私计算平台间数据安全流通的通用技术规范及配套生态支撑体系。在不暴露平台内部设计细节且不受平台自身更新、升级、扩容影响的前提下，行业级隐

私计算互联互通应立足金融行业发展需要，遵循统一的行业数据交互标准和接口规范，支持异构平台数据、算法、算力安全的交互与协同，支撑使用不同技术平台产品的不同金融机构共同协作完成同一隐私计算任务，并在平台互通、便于推广的基础上助力推动良性产业生态建设。

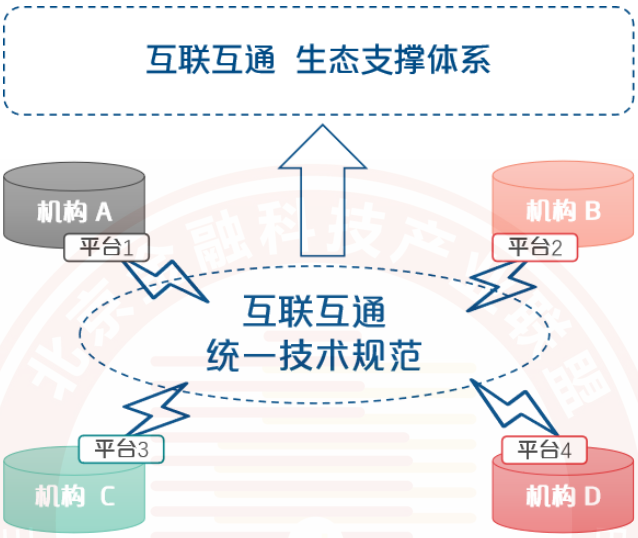

图1 金融行业隐私计算互联互通内涵示意图

行业级互联互通不仅要从技术层面进行顶层设计，更要从生态推广角度进行全盘考虑。具体可以从五个维度进行内涵解读：

（1）全局视野，中立客观。推进隐私计算互联互通，参与方不仅需要拥有全球化、全局性的视野定位，还应该秉持中立客观、兼容开放的立场，不计较眼前得失，积极参与贡献和集思广益。

（2）关照诉求，解决痛点。所谓行业级互联互通，不仅需要在技术上充分考虑产业痛点及各方诉求，各方共建一个既满足安全要求又能适应各平台产品多样性的互联互通统一框架，还要能够同时兼顾各方商业利益和知识产权保护诉求，做到互利共赢。

（3）架构合理，凸显安全。作为行业级互联互通框架，其架构应该是兼容、合理和安全的。兼容为先、合理为本、安全为基，既要能够尽可能兼容各类代表性技术路线和主流开源技术平台，相应东西向和南北向协议接口和流程设计更要高效而合理，同时还要以安全为最核心诉求和考量。

（4）成果切实，便于推广。互联互通统一框架应具备算法易开发、产品易集成、生产易运维、框架可测试、数据交互可审计等主要功能特点，可支持产业各方更好地开展集成对接和改造落地工作，使互联互通成果得以快速推广和实施。

（5）生态助力，相互促进。以统一框架生态为基础，各方将得以从现有技术路线竞争态势中脱离出来，集中自身优势资源，全力促进先进技术的迭代更新和商业价值的快速增长，并为上层算法市场提供更多样化的产品和增值服务。

2. 行业级互联互通实现思路

根据当前业界隐私计算技术发展及互联互通工作开展情况，行业级互联互通工作可结合“自下而上”和“自上而下”两条线进行推进实施。总体而言，宜按照“自下而上”思路以多方协作联合实践的方式推动互联互通框架和规范制定，以及按照“自上而下”思路从中立性组织和开源社区层面助推互联互通规范实际落地和产业生态建设。具体推进方式上，可通过以下三条路线的形式分头推进：

● **依托中立组织开展多方合作：**互联互通之所以难，不仅难在技术层面的问题，更是难在合作开放过程中各平台之间的规则兼容、责任匹配和

利益协同。基于此，需要有一个能够秉持中立立场、具有一定行业影响力的中立组织机构来牵头组织互联互通工作推进。该机构既要能够客观看待各类隐私计算技术，还要能够及时把握产业发展趋势，跟进监管方要求，并在最小代价前提下实现技术发展、规则制定和各方利益的有效权衡。

● **“实践+标准”双轮驱动：**互联互通最后一步需要产业各方精诚协作、合力共进。其中，各方宜采用“实践+标准”双轮驱动模式开展互联互通框架建设，不仅要从产业发展的高度开展行业统一标准制定，更要从泛行业支撑的角度开展跨平台多方实践。标准和实践两者齐头并进，相辅相成，缺一不可。

● **开放透明，合作共赢：**互联互通离不开规范的开放透明，只有将使用隐私计算技术的门槛降低，才能真正实现数据要素的流通。隐私计算的开放协同能够有效提升各大主流异构平台之间的兼容性和协同性，一方面有利于整合各方力量，节省重复性技术资源投入，聚合各自生态资源和扩大各自项目影响；另一方面也有助于加快基础设施建设，打通隐私计算产业生态，促进数据流通场景商业闭环构建，并最终助力实现跨行业数据协作共赢。

3. 金科联盟互联互通课题组织

北京金融科技产业联盟数据专委会于 2022 年初开始布局《金融行业异构隐私计算平台互联互通技术规范》《金融业隐私计算互联互通技术研究报告》两项重点课题（统称为“课题”），提出加快隐私计算互联互通统一框架建设，制定可落地互联互通技术规范和实现方案，以及推动产业各方开展基于主流隐私计算产品的互联互通可行性验证（见图 2）。课题

以团体标准和研究报告作为主要产出。

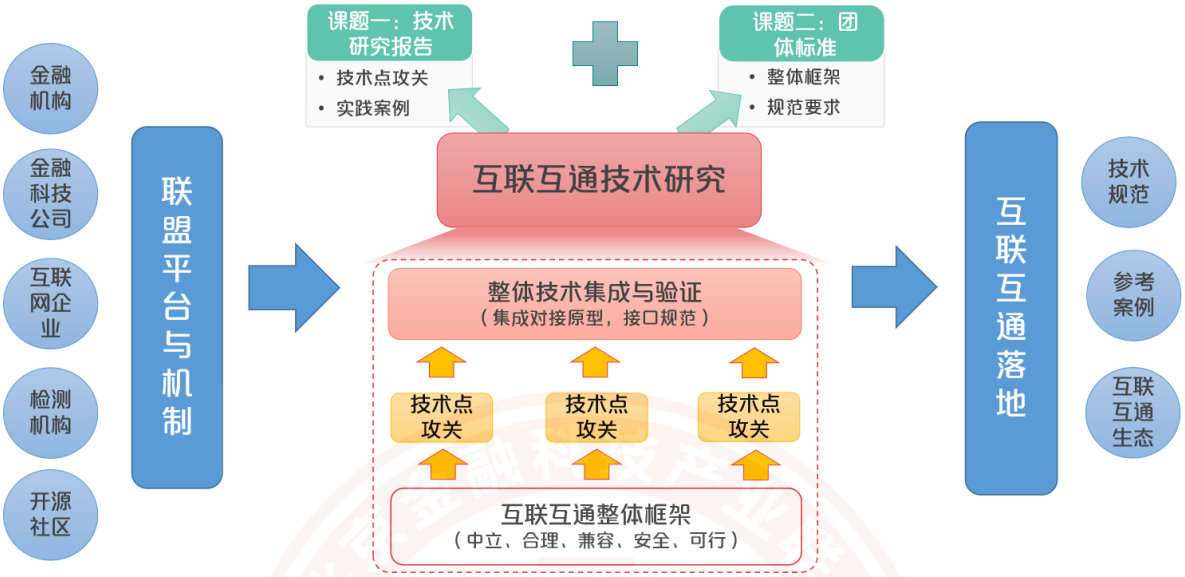

图2 金科联盟互联互通课题组织架构图

课题自提出以来受到产业各方的普遍关注和广泛认可，并于 2022 年 4 月正式获批立项，由中国银联牵头，主要商业银行在内的金融机构、科技公司、互联网机构、电信运营商、检测机构、开源社区等 50 余家单位共同参与。主要参与单位名单如表 1 所示：

表1 金科联盟互联互通课题参与单位名单（按拼音首字母排序）

北京金融科技产业联盟秘书处	北京百度网讯科技有限公司	北京八分量信息科技有限公司
北京冲量在线科技有限公司	北京火山引擎科技有限公司	北京数牍科技有限公司
北京银联金卡科技有限公司	北京原语科技有限公司	成方金融信息技术服务有限公司
度小满科技（北京）有限公司	复旦大学	光大科技有限公司
海光信息技术股份有限公司	杭州趣链科技有限公司	华控清交信息科技（北京）有限公司
华为技术有限公司	华夏银行股份有限公司	建信金融科技有限责任公司
交通银行股份有限公司	金融信息化研究所	京东科技信息技术有限公司
蓝象智联（杭州）科技有限公司	联易融数字科技集团有限公司	蚂蚁科技集团股份有限公司

上海富数科技有限公司	上海光之树科技有限公司	上海浦东发展银行股份有限公司
上海荣数信息技术有限公司	神谱科技（上海）有限公司	深圳前海微众银行股份有限公司
深圳市洞见智慧科技有限公司	深圳壹账通智能科技有限公司	深圳致星科技有限公司
深圳微言科技有限责任公司	神州融安数字科技（北京）有限公司	腾讯云计算（北京）有限责任公司
同盾科技有限公司	兴业银行股份有限公司	银联商务股份有限公司
招商银行股份有限公司	浙商银行股份有限公司	中国电信股份有限公司
中国工商银行股份有限公司	中国民生银行股份有限公司	中国农业银行股份有限公司
中国银行股份有限公司	中国银联股份有限公司	中国邮政储蓄银行股份有限公司
中金金融认证中心有限公司	中国信息通信研究院	中国移动股份有限公司

课题以金科联盟平台为依托，以**“安全互通、兼容开放、生态共荣”基本原则**为宗旨，积极拉通产业各方共识，在充分考量行业各方诉求的基础上，共同制定金融行业互联互通统一框架及实现方案；同时还将进一步借助和发挥各家单位技术特长，以“机构+厂商”牵头合作模式开展互联互通关键技术点研究攻关，群策群力，并在更大范围内推动跨平台互通场景探索和可行性验证，确保互联互通方案和规范的可落地、可评估和可检测。在金科联盟及数据专委会、50余家参与机构的共同努力下，当前课题已取得阶段性进展，本报告即是当前课题的重要成果之一。

二、金融业隐私计算互联互通框架设计

遵循第一章所阐述行业级互联互通内涵及实现思路，课题组启动了互联互通框架设计工作。在针对业界各方需求及技术影响因素开展大量调研工作的基础上，课题组梳理了互联互通框架设计的考量因素，并研制形成一套中立、兼容、安全、可行的行业级互联互通框架，为行业级互联互通的实际落地奠定了坚实基础。

（一）框架设计考量因素

1. 行业需求因素

行业级互联互通需兼顾产业参与各方诉求，包括商业化多样性、应用可落地性以及数据安全可监管性。为保证各方诉求内容的完整性，课题组前期联合金融机构、科技公司、检测机构、监管机构等类别的参与单位进行了深入的沟通和调研，进一步梳理明确了各方对于互联互通的主要诉求。大致如下：

（1）金融机构主要诉求：

- 避免烟囱化部署：尽可能仅部署一套系统底座，减少管理成本，降低系统安全风险；
- 数据流动安全可控：互通后隐私计算系统的执行过程以及数据流动需要具备一定的透明性和组合可替换性。

（2）科技公司主要诉求：

- 减少准入开销：能够减少针对准入产品的重复安全测试开销；

- 给予多样性空间：互联互通规范不要过多限制产品自由度，为特色功能与特性优化留有空间；
- 保护商业利益：增强互联互通商业利益保障，尤其是商业算法与开源算法的知识产权保护，同时希望在互联互通发展呈现规模效应后能够尽快形成良性的市场拓展态势。

（3）检测机构主要诉求：

- 需要有成型的互联互通技术检测框架体系；
- 隐私计算的检测应更有针对性与专业性；
- 在互联互通框架的基础之上能够形成良性的互联互通生态。

（4）监管机构主要诉求：

- 加快重点领域金融数字化转型标准制定，助力金融基础设施互联互通；
- 跨机构、跨地域、跨行业数据资源有序共享，提升数据要素资源配置效率；
- 充分释放数据要素潜能。

2. 技术实现因素

技术实现因素考量方面，考虑到隐私计算技术正在快速演变过程中，很难一步到位地实现全技术互联互通，因此，拟采取从最小必要设计到深化实现的分阶段方式来逐步推动隐私计算互联互通工作。第一阶段为本课题主要落实工作，主要思路是实现**系统架构级的最小必要互联互通**，即对隐私计算系统各层级进行合理解析，再对各层的关键应用程序编程接口（API，如图3中高亮色所展示部分）进行规范化与标准化（在尽可能不干

涉隐私计算系统层级的情况下为接口外各层级模块的灵活实现预留空间），做到规范化标准化与多样性兼容性之间的有效平衡；下一阶段将在本阶段实现框架级互通的基础上继续推动包括算法、算力和应用在内更深层次的互联互通工作。本课题也将同步围绕异构算法互通开展一些初步探索，为后续互联互通技术研究提供更好支撑。

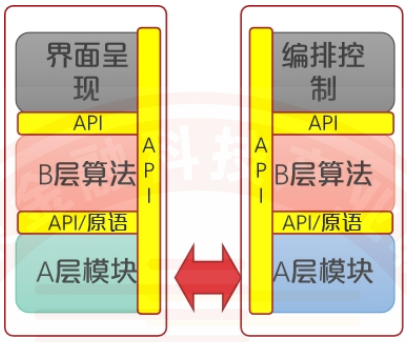

图3 框架级互通接口标准化设计示意图

总体而言，主要涉及如下几项关键设计考量：

● **最小必要原则。**从互通的角度，互通规范仅需定义节点、资源、任务、算法等最小必要的元素，其他涉及产品化的元素（例如租户、用户体系等）不在互联互通的探讨范围之内。

● **互通的防御性假设。**由于互联互通对端节点内部实现逻辑并非完全可控，故除了遵循所定义的接口外，还需要在流程处理上预留一定的安全冗余，以安全地适配各种可能的接口调用；在互通授权策略设计上，需要充分考虑策略定义的合理性，做到尽可能的通用和灵活，并通过机制保障资源能够得到受控访问。

● **架构解耦。**隐私计算的架构解耦有两个层次，一是框架底座与隐私计算算法的解耦，二是在隐私计算算法中应用算法和安全算子解耦。

● **安全可控。**安全算子作为隐私计算最核心的一部分，其安全性至关重要，可通过与应用算法解耦的方式来实现安全可控：一是通过解耦使安全算子可替换，使得合作方在不信任某个安全算子的实现时，仅需替换为另一个符合安全算子 API 定义且己方信任的安全算子实现，而不对上层应用算法的正常运行造成影响；二是解耦后的安全算子 API 是标准化的，因此数据的流动可以更有效地被追踪分析（例如可采用服务网格追踪等方法进行追踪）。在这两点特性共同作用下，互联互通机制的安全性将得以大幅提升。

（二）总体框架设计

基于上述考量点分析，本课题创新提出了基于“管理面与数据面切分，管理面分模块定义，数据面逐步解耦”核心理念的隐私计算互联互通总体框架方案（如图 4 所示）。

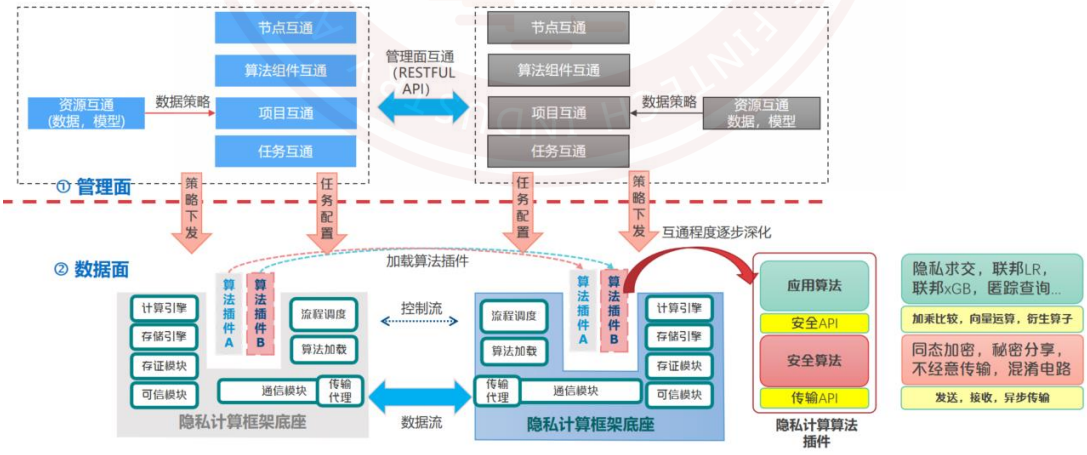

图4 隐私计算互联互通框架图

其中，最重要的一个设计思路就是**管理面和数据面的切分：**

1）管理面主要负责各隐私计算元素的互通（包括节点、算法组件、项目、任务等元素信息的互通同步），以及其中关于资源审批授权流程的交互；

2）数据面是真正执行隐私计算算法的部分。根据前期需求调研情况，各机构普遍希望仅部署一个隐私计算框架底座，对于各类功能算法则可以通过加载各异的算法组件来实现。

初期算法组件可以以一个整体打包的算法容器的形式存在，随着互通程度逐渐地深化，将逐渐解耦成本章第（一）节中所提到的安全算子与应用算法相分离的形态。其中，安全算子包括同态加密、秘密分享、不经意传输、混淆电路等，应用算法包括隐私求交（PSI）、逻辑回归（LR）、极端梯度提升树（XGB）、匿踪查询等；安全算子和应用算法间的安全API主要涉及对安全算子的功能性操作，如加乘比较、向量运算以及衍生运算等；传输层API则包括发送、接收以及异步的传输等。上述所提及两层API的规范化，是本课题所涉互联互通框架着力希望突破的点，这对于提升隐私计算互通算法运行过程的可控性具有重要意义。

3）在管理面与数据面之间，仅有作业任务和策略权限的配置定义会涉及两个面的耦合，其他管理面元素及流程调度相关的规范化定义原则上都可以分开进行，如此将得以实现最大程度的解耦化。

（三）框架组成解析

互联互通框架定义仅是基础，要想切实推动框架落地，还要从更细粒度去分析各框架模块的组成细节，以确认各模块所涉相关技术实现

点。基于此，可对上述总体框架作进一步的解析，并拆分成如图5所示的层次。

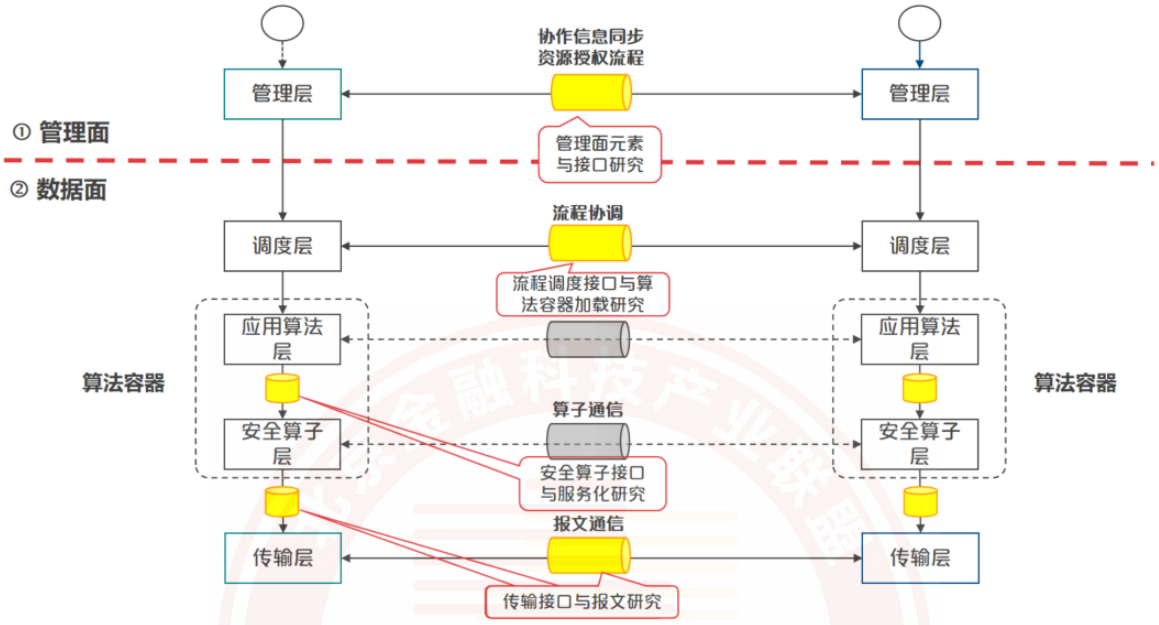

图5 互联互通总体框架解析图

1）管理面相对较为简单，由于管理面仅涉及管理层间交互，因此管理面的关注点在于管理层间协作信息同步及相关资源的授权。

2）数据面相对较为复杂，包括流程调度、算法容器加载及对外传输信息的传输层三大部分。其中，算法容器还可再细分为应用算法层和安全算子层两层。

对于跨平台互联互通而言，最理想的方式是能够统一定义其间所有异构隐私计算平台间的API，即将图5中水平圆筒状的API都进行标准化定义。就当前业界的整体发展水平而言，管理层、调度层和传输层这三层异构隐私计算平台间（东西向）的API标准化是相对可行的，至于隐私计算算法层面在东西向的API则难以通过一致性抽象的方式来形成标准化定义。为

此，本课题将转换一种思路，通过采用隐私计算平台层次间通信调用（南北向)接口定义替代东西向的接口定义的方式来解决算法层面各模块对齐的问题；如此，仅需要在应用算法层和安全算子层间以及在算法层和传输层间各定义一层接口（图4中高亮色圆柱形接口），即可实现一个初步版本的互联互通（与东西向接口相比，这两层南北向接口所需实现的功能是相对清晰也更易于定义）。

在图 5 基础上，还需要进一步结合互联互通框架底座（如图 6 数据面蓝绿颜色模块标注）才能形成完整的互联互通框架。该底座宜包括流程调度模块、计算引擎、存储引擎、日志存储模块、机密计算模块、算法容器加载模块等偏系统与功能级的模块。一般而言，隐私计算互联互通的实现会更多地关注到算法容器和部分偏系统级模块的接口定义，如计算、存储、日志、度量等一些接口。此外，可信执行环境（TEE）作为当前隐私计算三大路线之一，也将会涉及异构 TEE 系统之间的互联互通问题，故底座系统模块还可能配置有基于 TEE 技术的机密计算模块。

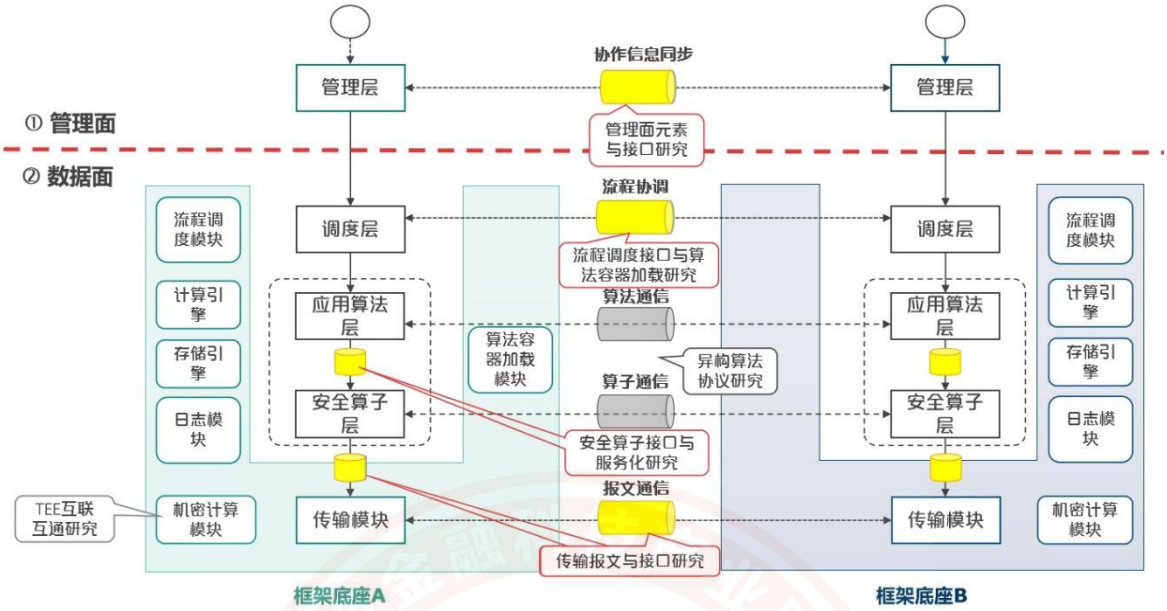

图6 互联互通研究点分解

综上，上述互联互通框架可以分解成“管理面元素与接口研究”“流程调度接口与算法容器加载研究”“安全算子接口与服务化研究”“传输接口与报文研究”“异构算法协议研究”“TEE 互联互通技术研究”等六个研究点。为切实推动互联互通框架落地，本课题已组织各参与单位按照子课题的形式有序开展上述研究板块的研究验证工作，并在第三章中予以详细阐述。

（四）框架功能特色

由上述的分析，本课题所提出的互联互通框架具备如下的特点：

● 全面覆盖隐私计算的主流技术路线，兼容多方安全计算（MPC）、联邦学习以及可信执行环境（TEE）各主要技术路线。

● 管理面与数据面低耦合切分，整体互通架构更为清晰。

● 算法与框架底座解耦，借助灵活的算法可插拔式设计，实现可行性高。

● 应用算法与安全算子解耦，着重体现隐私计算的安全属性，有助于从机制层面增强互通隐私算法的安全可控。

● 各层的解耦以及标准化，有利于形成隐私计算的良性生态。

三、互联互通技术分解研究

依照第二章设计，互联互通将分解成“管理面元素与接口研究”“流程调度接口与算法容器加载研究”“安全算子接口与服务化研究”“传输接口与报文研究”“异构算法协议研究”“TEE 互联互通技术研究”等六个部分。在课题组织工作中，这六部分以子课题的形式进行推进，各子课题牵头单位及参与单位如表 2 所示：

表2 互联互通牵头单位及参与单位简称名单（按子课题贡献度排序）

子课题名称	牵头单位	参与单位
管理面元素与接口研究	招商银行、浦发银行 富数科技、蓝象智联	工商银行、中国银行、光大科技、兴业银行、浙商银行、交通银行、度小满、蚂蚁集团、洞见智慧、融安数科、光之树、神谱科技、中国移动、同盾科技、数牍科技、中国银联
流程调度接口与算法容器加载研究	浦发银行、富数科技 洞见智慧、百度网讯 微众银行	中国银行、华控清交、蓝象智联、度小满、蚂蚁集团、融安数科、八分量、同盾科技、中国银联
安全算子接口与服务化研究	蓝象智联、洞见智慧	工商银行、中国银行、浦发银行、招商银行、BCTC、华控清交、神谱科技、联易融、度小满、蚂蚁集团、富数科技、融安数科、华为、同盾科技、中国银联
传输接口与报文研究	光大科技、蓝象智联	成方金信、中国银行、浦发银行、微众银行、华控清交、度小满、蚂蚁集团、洞见智慧、富数科技、同盾科技、数牍科技、壹账通、百度网讯、中国银联
异构算法协议研究	蚂蚁集团	工商银行、农业银行、浦发银行、招商银行、浙商银行、华控清交、神谱科技、洞见智慧、壹账通、联易融、富数科技、同盾科技、华为、中国电信、中国银联

TEE 互联互通技术研究	工商银行、蚂蚁集团	成方金信、农业银行、浦发银行、招商银行、冲量在线、百度网讯、富数科技、壹账通、光之树、致星科技、同盾科技、中国电信、中国移动、海光、中国银联

本章将介绍各子课题的主要研究内容与思路，第四章将着重阐述其中所涉及关键技术点的研究成果。

（一）管理面元素与接口研究

1. 概述

在互联互通当前面临的难点中，如何在不同平台的应用层达成管理功能互通，安全可控地实现资源同步，是后续异构平台协同隐私计算任务的前提和基础。一个完整的隐私计算平台，除了核心算法外，还包括节点发现、资源授权、项目管理、流程编排等控制管理功能，不同平台的管理层架构也因各自的设计思路和业务侧重不同而存在差异，因此，跨异构平台执行隐私计算任务，就必须先打通基础环节的管理功能。

管理面元素与接口研究，是在充分考虑各技术平台差异的前提下，对隐私计算平台的基本元素、各级资源的授权流程及支撑授权流程的 API 接口进行标准化定义，从而对互联互通的隐私计算平台内外部各元素进行协调管控，实现管理层级的互联互通。

2. 研究目标

管理面元素与接口研究旨在形成一套行业认可的、规范统一的管理面元素与接口定义规范，为隐私计算产品的管理功能设计及管理层级互通提供参考模式，实现节点管理、数据管理、项目管理、流程管理、作业管理、

组件管理等操作，继而实现安全可控的跨平台互通协作，使不同异构平台能够协作完成相同的隐私计算任务。

3. 研究内容

管理面元素与接口包括以下几个方面的研究内容：

(1) **规范管理面基本元素。**从异构平台互联互通出发，对最小必要的实体元素进行抽象和定义。

(2) **设计管理面互通流程。**分层级依次对节点互通、数据互通、项目互通、流程互通、组件互通等节点间资源授权步骤进行定义，确保相应权限得到有效管控，安全性得到有效保证。

(3) **制定流程有向无环图（DAG）及作业配置（CONF）通用结构。**考虑不同平台间协同计算的兼容问题、标准化流程DAG及作业CONF结构，支撑数据平面对计算资源进行统一调度。

(4) **制定管理面互联互通服务接口标准。**基于管理面互通流程，分模块定义互通服务接口，充分考虑安全性、必要性、普适性、开放性、易用性，完成管理面互联互通服务接口设计和定稿。

4. 设计原则

考虑到隐私计算技术产品之间存在较大的差异的现状，互联互通不能要求所有平台同质化、统一化，需要在保证各平台的独立性、完整性和安全性的基础上对实现互联互通的最基础环节求同存异。因而，管理面互联互通的架构设计应满足以下特性：

● 最小必要：在满足功能要求的前提下，抽象出各隐私计算平台通用的管理面元素最小必要单元，简化跨平台交换信息，平台内部的扩展元素

独立管理，在保证平台独立、功能完整、策略安全的前提下，实现最基础环节的求同存异。

● 一致性：管理面互联互通基于通用、规范的接口和互联协议，进行跨平台的互联和同步操作,应能保障各层级资源信息在建立互联的平台间的最终一致性。

● 安全可控：不同平台间的协同、交互应通过统一的安全策略、认证与授权机制等保障各层级资源安全互联。由于不同技术平台对资源的授权管控粒度存在差异,建立多方之间的资源授权时,应提供灵活的安全策略,避免多方安全策略上的耦合。

● 灵活兼容:管理面互联互通应充分考虑当前隐私计算平台的技术架构与业务架构差异,兼容各类隐私计算平台,支持各平台灵活加入或退出,并可随技术发展适配更多新的隐私计算平台架构，具备可扩展性。

5. 研究思路

金融行业隐私计算互联互通总体框架中,将管理面与数据面拆分解耦,让管理面聚焦于节点、数据、项目、流程、作业配置、组件等各层级资源的静态协同，而数据面关注作业运行时的动态协同。

管理面元素与接口研究的实现路径从“提炼最小必要元素-分层次定义互联协议”的思路出发进行研究讨论：统一的概念模型是实现管理面互联互通的基础，本子课题围绕“自底向上、分层解耦、最小必要”抽象出一套隐私计算对象基本模型规范，包括节点、数据集、项目、流程、作业、任务、组件、模型八大基本元素实体；将元素实体看成不同层次的资源，以统一的平台间通信要求为基础，进一步约定各层次资源在发现、申请、

授权、同步等环节的规范流程和要求；适配调度层通用设计，对流程中的组件编排和作业的参数配置这两个具体计算执行时依赖的静态描述信息进行统一规范（见图 7）。另外，管理面互联互通必须建立在安全底座上，互联协议中各开放接口的调用需要满足接口鉴权的安全要求。

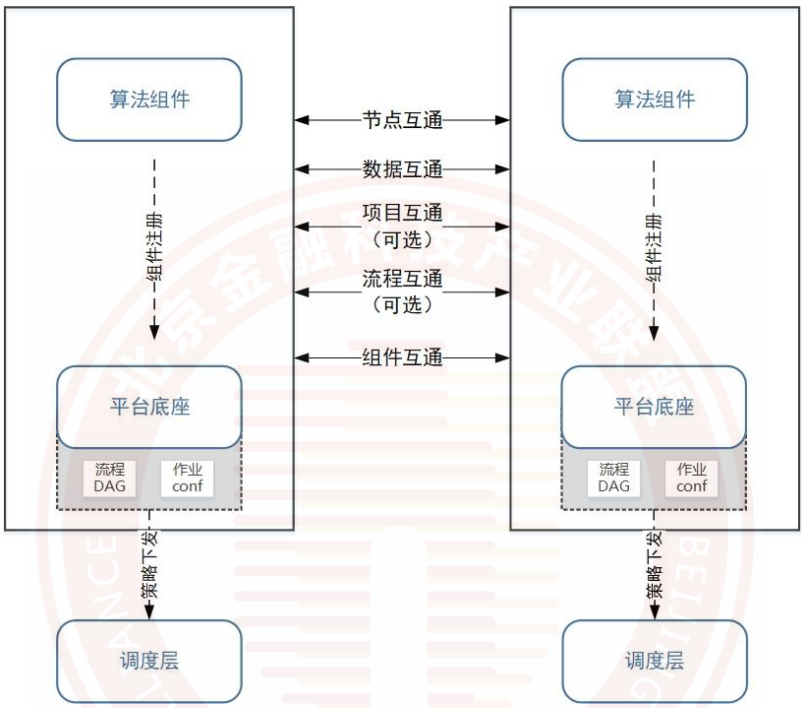

图7 管理面元素与接口研究架构图

（二）流程调度接口与算法容器加载研究

1. 概述

本子课题主要包含两大内容，分别是隐私计算过程中流程调度接口的设计以及隐私计算算法容器加载的研究，是整个隐私计算互联互通技术研究的核心部分。

在流程调度方面，由于隐私计算是一个涉及双方数据协同的运算过程，

在互联互通的过程中会涉及双方网络传输以及异步运算等关键步骤，网络延迟或者计算资源调度异常都会导致隐私计算任务等待或者失败，因此在异构情况下，需要标准化的调度方案，使用标准化的作业与任务协同接口，来统一协调异构双方隐私计算任务的执行。

在算法容器加载方面，算法是隐私计算的核心，而算法的加载就是使用标准化的接口或方式来管理异构算法，其实现方式决定了算法的可扩展性和易扩展性，并且其设计方式也会影响到算法的执行效率。因此，我们需要制定统一的算法镜像构建标准与接口并定义一套规范的镜像加载机制与流程，从而保证整个算法容器加载过程的安全、高效及高可用。

2. 研究目标

本子课题研究旨在形成一套行业认可、规范统一的作业调度与算法容器加载流程以及对应的镜像构建标准与接口定义规范，为隐私计算产品作业运行过程中调度方对作业与任务的控制与任务组件的加载过程提供参考模式，实现作业调度控制、任务调度控制、组件注册发现、容器管理与加载等操作，从而实现安全可控的跨平台互通作业运行，使不同异构平台能够协作完成相同的隐私计算任务。

3. 研究内容

本子课题包括下面几个方面的研究内容：

(1) **研究标准化作业及任务的核心调度流程，确定调度面的基本要素及相互关系，定义节点内及节点间调度面的核心流程。**基本元素参考为：数据集、模型、作业、任务等；核心流程参考为：数据集加载、模型加载、作业运行、作业重跑、任务运行、任务重跑等。

(2) **标准化设计作业及任务的调度和管理接口。**描述作业及任务调度加载通信资源、计算资源和存储资源的标准形式。

(3) **研究算法容器镜像加载机制与安全性验证机制。**对标准算法镜像进行约定（其中包括：镜像名称、版本、必要目录结构、必选的参数定义、标签信息等）。

(4) **规范算法容器加载流程。**根据该流程设计标准化的算法容器接口。

(5) **构建算法容器加载机制与流程调度的实验案例。**以此验证课题可行性。

(6) **系统接口研究与定义（可选）。**定义算法容器与隐私计算底座之间所涉及的系统层面接口，包括分布式计算接口、硬件加速接口、存储接口、日志与度量接口等。

4. 设计原则

现如今隐私计算行业内由于隐私计算技术的百花齐放，各家产品的作业调度系统及算法组件的加载方式都存在着较大的差异。若要求各家产品使用统一的调度系统以实现互联互通，那么对各家而言改造技术难度大、工作量重，并且这样的方式失去了技术多样性，也不利于行业发展。因此，流程调度的互联互通需要在保证各平台的独立性、完整性和安全性的前提下，对实现互联互通的最基础部分进行标准化定义与设计。综上，流程调度与容器加载互联互通的架构设计应满足以下特性。

● 最小必要：在满足基本作业调度功能的前提下，抽象出各隐私计算平台作业调度流程中通用的最小必要实体，简化整个流程调度与容器加载

过程，仅对平台间的接口进行定义与设计，而不深究各平台内作业调度细节，在保证平台独立、功能完整的前提下，实现最基础环节的求同存异。

● 标准化：作业调度层面的互联互通应基于标准化的作业流程调度过程、算法镜像构建规范、组件注册与发现机制，保证作业、任务、容器等运行信息在平台间的一致性。

● 安全可验证：不同平台构建的算法镜像应满足标准化的算法镜像构建规范，满足镜像防篡改、漏洞扫描及镜像签名等安全验证机制，保证算法镜像的安全可验证性。

● 灵活兼容：作业调度层面的互联互通应充分考虑当前隐私计算平台的技术架构差异性，兼容各类隐私计算平台，并可随技术发展适配更多新的隐私计算平台架构，具备可扩展性。

5. 研究思路

流程调度与算法容器加载可拆分为调度核心流程设计与算法容器加载机制研究两个部分，前者聚焦于隐私计算作业调度中的实体、过程及状态变化的设计，后者则聚焦于作业中各组件的注册发现与加载机制的研究（见图 8）。

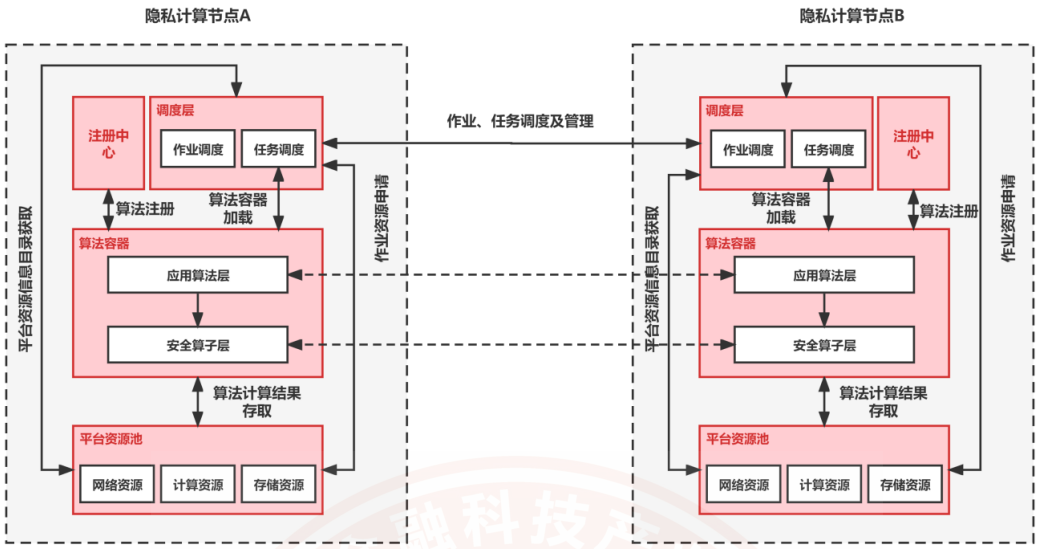

图8 流程调度与算法容器加载架构图

流程调度与算法容器加载的实现路径从隐私计算过程中作业的调度到作业中组件的加载这一思路出发进行研究讨论：统一的概念模型是实现调度面互联互通的基础，本子课题在“管理面元素与接口研究”子课题的基础上进一步定义了流程、作业、任务、组件、资源 5 大基本调度面实体及实体属性；根据统一的作业与任务状态的定义与状态间的转换关系，本子课题设计了作业与任务的核心调度流程，并明确了作业与任务的生命周期；为了标准化算法容器组件的镜像，课题制定了组件镜像构建规范并设计了标准的组件加载管理机制与容器加载流程；课题还对组件的实例状态、注册发现流程、实现逻辑流程及管理服务流程进行了规范定义，以实现更完整、更统一、更兼容的算法容器加载过程。

（三）安全算子接口与服务化研究

1. 概述

在互联互通的框架中，安全算子接口与服务化研究主要负责安全算子服务层的框架、功能、接口设计和实际应用场景的一些建设指南。鉴于互联互通需要异构算法组件支持，使得不同架构的隐私计算平台之间需要部署、使用多个异构算法组件才能实现互通。这些异构算法组件虽然实现原理不同，但是对于最小必要计算算子（例如：加解密运算、多方安全算子计算等）存在重复实现和不利于统筹管理等问题。本互联互通设计将安全算子抽离出来形成安全算子服务层，由隐私计算平台统一管理安全算子任务。该方案解耦了隐私计算算法与安全算子，并使用容器化的方式封装，有利于安全算子在异构平台间进行共享、管理、部署和使用。

2. 研究目标

(1) 设计通用的安全算子层框架与功能接口。初期主要是对子课题的安全算子层进行框架设计、功能定义以及相关接口设计。设计阶段中，应对实际应用场景服务的稳定性、安全性和计算的性能进行充分考虑，使得安全算子层能够满足实际场景下各种部署需求。

(2) 提出可落地方案的建设指南。在子课题的方案设计中针对不同类型的应用场景，总结出一些常用的部署方案，为后续的互联互通实际落地提供建设性思路。

(3) 通过大量的测试案例验证子课题的可行性。在初期设计阶段完成后，后期针对子课题中的功能设计出对应的测试案例，测试目标主要包

含功能、性能、安全三方面。功能方面主要是验证定义的功能和接口能否满足互联互通的基本要求;性能方面主要是验证框架设计的方式对隐私计算任务计算效率的影响;安全方面主要是验证安全算子容器化部署的安全性。

3. 研究内容

本子课题研究包括下面三个方面的内容:

(1) **研究安全算子服务化的架构。**结合整体互联互通框架的需要,对算子服务化进行定义,并明确算子与算法的边界;

(2) **制定安全算子服务接口标准。**充分考虑在安全性、普适性、开放性、易用性等方面的诉求,确立接口设计原则;

(3) **制定安全算子服务化能力建设指南。**对技术难点深入分析,力求务实、有效地输出有价值的建设建议和观点。

4. 设计原则

● 职责明确:明确算子服务的功能范围,适应总体互联互通架构,吸收行业经验,谨慎论证和定义算子服务化的边界。

● 标准一致:定义统一的算子服务化接口,在保障各平台功能层面的独立性、先进性的基础上,统一算子与其他模块的交互操作,作为业内统一算子服务标准的基础。

● 安全可控:凸显出算子的安全性,根据当下业内实际使用需要,控制算子的内容和范围,满足算子的安全原子性,为安全算子提供可控、可评估的安全保障。

● 务实有效:提供切实有效的建设指南,在确保算子安全性前提下兼

顾实际场景建设在算力资源、大数据处理、系统稳定性、高可用等多方面的差异化要求，提供具备伸缩性的算子服务系统技术建设指南。

5. 研究思路

本子课题是从互联互通中算法算子难以协商的问题以及 AI 技术与 MPC 技术融合成本高的问题入手，提出了安全算子服务化的解决思路，并以金融行业异构隐私计算平台项目为抓手，以“实践+标准”的双轮驱动方式推进，深入持续地开展研究工作（见图 9），整体研究思路如下：

(1) 分析互联互通和行业技术痛点，明确算子服务化的意义和价值；

(2) 提出安全算子服务化架构设想和安全算子服务接口，多方论证其兼顾性和合理性；

(3) 结合实践验证并提出安全算子服务化能力建设指南，以最佳实践论证其可行性；

(4) 加速算子在业内的标准化进程，推动互联互通的业态落地；

(5) 以标准化为基础建立安全验证方法，为安全评估降本增效。

图9 安全算子接口与服务化架构图

（四）传输接口与报文研究

1. 概述

传输接口与报文研究，是在充分考虑各技术平台差异的前提下，对平台传输层的传输接口、传输协议、报文格式等进行标准化定义，从而为隐私计算平台间的互联互通提供底层的通信基础。

2. 研究目标

本子课题拟提出能够在异构的隐私计算平台之间，建立一套能够指导各隐私计算平台算法和算子相互协作的传输规范，为隐私计算平台间的传输层互通提供参考模式，并进一步规范传输接口、传输协议、报文结构等内容，继而实现安全可控的跨平台互通协作。

3. 研究内容

传输接口与报文子课题的研究内容包括：

(1) **设计组网结构。**研究传输层的组网方式，包括传输主体的身份、组网的认证方式、网络的拓扑结构；

(2) **分析安全问题。**研究包括传输安全与业务安全在内的相关问题，对传输数据提出安全需求，并规范跨平台合作方识别、权限控制等业务内容；

(3) **规范标准传输 API。**研究算法与算子调用传输的标准 API，以此规范算法和算子在通信编程上的方式，确保算法和算子在传输层面满足可移植性；

(4) **统一标准传输协议。**研究当前隐私计算平台主流的网络传输模

块，以此建立节点间标准的网络传输协议、通用报文编码格式；

(5) **规范协议间转换。**研究不同传输协议间的转换和实现；

(6) **夯实端到端传输。**研究端到端的通信链路网络，确定全链路传输所需要的传输基础设施形态和部署模式。

4. 设计原则

跨平台互联互通场景中的传输框架应满足最小必要性、通用性和安全性要求：

● 最小必要性：在满足功能要求的前提下，抽象出各隐私计算平台通用的传输层元素最小必要单元，在保证传输安全的前提下，实现底层通信环节的求同存异。

● 通用性：通用性原则要求，在互联互通场景下，传输框架应兼容市场中主流的传输协议与报文格式，并对算法与算子调用的传输 API 的最小化接口做出标准化规定。

● 安全性：安全性原则又具体包括传输安全、业务安全。其中，传输安全是指通过采取必要措施，确保数据在传输阶段，处于有效保护和合法利用的状态，满足数据机密性、完整性、不可抵赖性等安全需求。具体针对网络传输场景，则需要通过相关管理与技术手段，保障数据通过网络传输过程中，可以有效地抵抗第三方通过网络嗅探、数据包拦截、数据包篡改、数据包重放、身份伪造等手段破坏传输数据的安全需求；业务安全是指保护业务系统免受安全威胁的措施或手段，确保业务所提供的服务的安全，防范业务流程中出现风险，避免业务遭受割裂威胁或遭受经济损失，保障整体业务逻辑的顺畅。具体针对隐私计算互联互通场景，业务安全主

要包括跨平台合作方识别、权限控制、安全漏洞防范等方面。

5. 研究思路

基于本子课题的框架图，如图 10，并结合推进方式，我们将该子课题拆分为六个部分，分别为“标准传输 API”“标准传输协议”“协议间转换”“组网结构”“安全问题分析”和“端到端传输”。其中“标准传输 API”对应图中算法算子模块调用的传输层 API 的接口设计；“标准传输协议”和“协议间转换”对应图中隐私计算框架底座间的传输协议与报文格式的互通规范；图 10 中以直连的组网结构为例，整个链路的传输是基于传输安全和业务安全得到保障的前提下进行。

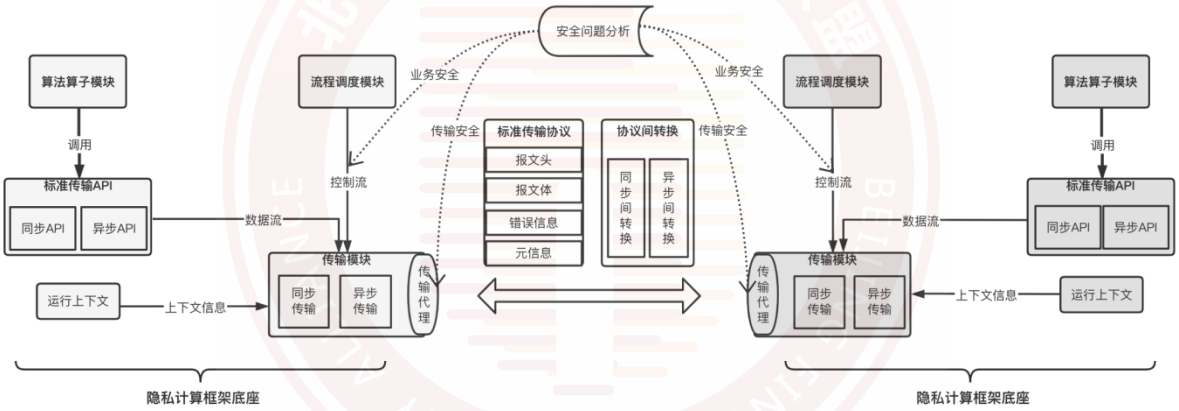

图10 传输接口与报文研究的研究路线图

为了规范算法算子调用传输层的接口标准，本子课题提供两种接口调用模式，即同步调用和异步调用；在研究过程中，也启发式地将缓冲机制纳入标准化范围。为规范隐私计算平台间的通信标准，本子课题提出了同/异步兼容的设计模式，并基于规范化的报文头结构定义了标准报文传输格式。另一方面，本子课题还提供了一套协议间转换的标准和要求，包括同步间协议转换和异步间协议转换。

考虑到不同隐私计算平台之间的连接方式，本子课题给出直连模式和路由转接模式两种基础拓扑结构。对于传输层涉及的安全问题，本子课题将其拆分为传输安全和业务安全以分别讨论研究。最后，本子课题也会提供端到端传输的常见模式，以供相关人员参考。

（五）异构算法协议研究

1. 概述

异构平台开放算法协议的互联互通是指通过定义算法执行流程中交互接口协议，再由平台独立开发算法来实现异构平台间算法互通的模式。

在一个多方安全计算或者联邦学习的训练任务中，由于同一种算法存在完全不同的算法流程思路，或者算法思路相同却由于不同厂商通信框架及具体实现流程不同使得各方算法实现上存在较大差异，从而导致无法实现基于平台自有算法的互联互通。以基于同一算法不同算法流程的LR为例，虽然基于同态及秘密分享技术的LR（HESS-LR）与基于联邦学习的LR（Hetero-LR）都是逻辑回归算法，但在算法流程和细粒度的交互细节上是完全不同的。因此，即使是针对同一个算法开展互联互通也是存在很大的挑战。因此，基于开放算法的互联互通研究是有价值的，它将能够帮助我们更快、更好地达到以下目标：

(1) 增加算法安全透明度，提升安全可信度。开放算法协议互通，本质的作用就是不再把算法流程、底层密态计算过程当作黑盒，而是把交互流程打开公开出来，以便不同厂商能够根据这个要求进行实现和交互。

对于隐私计算算法来说，核心的安全风险是在中间数据交互中的信息泄露，若这个交互的信息是被明确定义的，则算法的安全性将得到较好的保证。

(2) 降低算法容器的安全审核成本，提升了隐私计算业务合作效率。基于算法容器实现的互通模式，通常要求互通双方底层是同构技术平台，但在通过这种模式链接新数据源时，如果新数据源的底层技术引擎是来自不同的技术厂商，那么其中一方需要引入另一方的技术引擎底座才能完成互通。考虑到新技术引擎底座需要接触数据源做开放算法通信交互，合作方往往出于数据保护的需要重新执行安全审核评估等流程，进而导致安全审核成本的增加及双方合作效率的降低。

(3) 基于开放算法协议的互通协议允许异构平台有自有的技术实现；若新数据源的技术平台已经遵循开放算法协议，则无需再额外引入新的外部算法容器就可以完成互通，进而避免集成来自不同厂商的算法容器带来的风险，缩短安全审核周期。例如，机构 Alice 部署了厂商 P 的隐私计算产品，机构 Bob 部署了厂商 Q 的隐私计算产品，如果机构 P 和 Q 都遵循了开放的算法互通协议，那么对 Alice 来说，不需要重新审核厂商 Q 出品的技术产品，不需要再次进行复杂的安全审核等过程就可以直接链接上机构 Bob 的数据源，最终理想状况有机会达到数十上百家不同技术产品互联的效果，潜在可以链接的数据源数量和效率也将大幅提高，也有助于减少机构内部众多不同厂商的算法包管理。

(4) 促进行业算法迭代与进步。基于异构平台开放算法协议的互通，通过透明的算法交互设计，可以在明确出域信息安全性透明的前提下，对不同算法参与方的效果/性能进行直接验证和评价。若合作过程中出现某

一参与方性能成为瓶颈的情况，则业务方有更多理由、算法提供方也有更强动力去做算法优化，进而提升整体算法效果性能。

2. 研究目标

本项研究旨在通过对异构平台开放算法协议研究，尝试探索一些可实践可推行的基于开放算法协议的互通方案，以为后续更深层次的跨平台互联互通实践落地带来更多可能性。

鉴于行业现状和隐私计算算法的复杂性，本子课题并不试图完成全部算法能力的通用互联互通的定义，而更多通过一些具体案例给出异构互联互通的参考路径和思路。对于课题的研究目标计划为：

(1) 所制定的标准协议是针对业界广泛隐私计算公司认可的安全算法流程。例如 PSI 算法变种特别多，ECDH-PSI、KKRT16、SPOT19、Circuit-PSI、Labeled-PSI，算法分支种类特别多，而且每个算法的细节流程都不太一样， 但大多数厂商有能力提供 ECDH-PSI 的实现。那么 PSI 将会选择以 ECDH-PSI 作为标准来兼顾更多的厂商，而不是比较新的，或者刚发布出来的算法。再例如秘密分享较为广泛使用的协议有 spdz-semi2k、ABY3 等，那也会考虑选择一个更为常用的协议来拟订。

(2) 算法范围初定为常见的算法。由于开放算法协议的互通制定工作量会比较大，技术挑战也比较大，一期目前暂定会聚焦在 PSI，特征工程算法 WOE，以及通用秘密分享协议可以支持的金融风控算法等。

大多数情况下，不同计算方使用不同 MPC 算法协议是无法互通的。譬如做一个矩阵乘法，P 平台用 SPDZ 协议做，Q 平台用混淆电路协议做，此时两者是无法互通的，因为理论上不直接满足互通可行性。虽然目前有

混合协议（例如 ABY、ABY3 等），但是能互通的前提也是双方同步在同一形态（例如双方同时转到 Arithmetic / Boolean / Yao 等）。异构算法的互通在理论层面还待更多的研究,本子课题也会尝试做异构算法协议的数据转换研究。

3. 研究内容

异构平台的开放算法协议研究课题的研究内容包括以下几个方面：

(1) 确定头部要支持的算法以及不同流派算法的开发协议互通架构思路。异构平台开放算法协议的关键技术难点之一在于隐私计算的算法种类繁多，工作量大，技术路线也繁多。例如联邦学习和纯多方安全计算的异构平台开放协议实现思路差别明显。本子课题的关键之一在于要对隐私计算算法的众多技术路线进行归类并且提供设计原则。

(2) 确定头部算法的握手流程，确定握手过程中开放互通必要的算法参数和安全超参标准。以 PSI 为例，譬如确定为 ECDH-PSI 流式算法，超参情况确认是否双方都能得到交集结果，还是某一方得到结果。PSI 由于是专用 MPC 协议，适用性主要是 PSI 本身，而为了提升课题的通用性，目前计划是通用协议和安全算子接口子课题一起。若 MPC 协议原语能通，并且上层安全算子接口也标准化，则未来所有的通用 MPC 下的应用算法（例如 SS-LR、SS-Pearson’s R）能互通。而 ECDH-PSI，或者 FedAVG 等已经广泛使用的专有算法流程则单独制定算法接口协议,具体开展上可以先以最小化支持为标准。协议内容将具备可扩展性，并且和语言弱耦合（例如不会将诸如 pickle 这类和 python 强相关的技术栈作为协议标准）。具体内容会比较细致，例如 PSI handshake 需要指明 PSI 算法，

曲线类型（如约定各方都是 curve25519 等）需要明确批的大小，然后各自根据对方的信息进行响应。

(3) 原则上协议不对编程模式进行限定，但是本子课题会给出推荐的参考架构实现。在编程模式上，原则推荐支持异步化的 Actor 模式（大多数远程过程调用编程框架支持类 Actor 模式），计算和通信解耦，适合异构互通。本子课题不做强制限定，仅做协议内容的标准限定。

(4) 本子课题是研究性质课题，最后计划产出选定算法的详细开放算法关键步骤交互内容协议，比如握手流程，以及关键算法交互流程。这个协议会落在开源仓库（repo）上，并且会在开源上给出异构语言的参考实现，在一到两个算法上达到开放算法协议的互通。

4. 设计原则

考虑到隐私计算算法及实现的多样性，需要保障各厂商的实现成本尽可能不要过高，在算法协议一致的基础上还能留有一定的自主实现空间。因此，课题的设计的原则有：

● 算法本地实现开放： 在满足开放算法协议一致的前提下，对于本地计算部分的实现原则上不约束，对于不必要的流程不过度设计接口协议，从而希望给各参与平台更大的自由度。

● 支持算法和安全机制扩展：不同隐私计算路线下的算法都会支持一些公共的算法类参数，这类参数应当予以区分以便未来有新的隐私计算算法出现时，新技术能够遵循和复用已有的算法类参数，方便上游的接入。同时，安全类参数比如椭圆曲线类型，同态类型等都需要可扩展，以便未来安全领域有新发展时，本子课题能够快速扩展和适应。

● 交互出域信息透明：开放算法协议原则上能更好地实现算法的安全性保障，是因为出域的信息是透明的，这也要求算法协议的设计与实现的安全性须得到保障。

5. 研究思路

异构平台开放算法协议是在其他研究课题的基础上，进一步将算法的跨域传输内容进行公开透明化。面对异构平台开放算法协议隐私计算的技术路线和算法种类繁多、工作量大的现状，本子课题的关键之一在于要对隐私计算算法的众多技术路线进行归类并且提供设计原则。如图 11 所示。

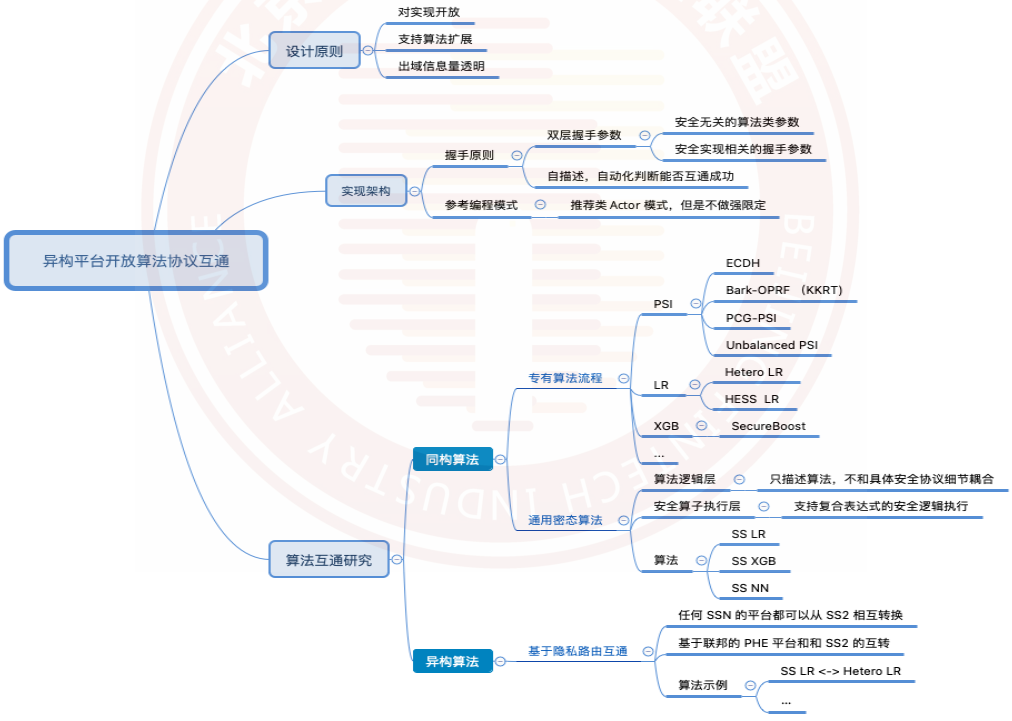

图11 异构平台开放算法协议研究路线图[1]

1名词引用：

Hetero LR[4]：https://arxiv.org/abs/1711.10677

其中同构算法的实现需要双方遵循一致性的隐私计算算法流程，核心在于对出域信息的协议指定。异构算法则没有此限定要求。

由于隐私计算流派众多，即使是同构算法也会存在不同的技术路线。以联邦学习为例，关键子流程可采用半同态或者多方安全计算的方式来保护一些专有算法流程，也可以采用通用密态算法技术路线实现。对于专有算法流程技术路线，专有流程不一定总是联邦学习的技术路线，也可能是PSI、PIR 等专有 MPC 协议。对于通用密态协议技术路线，鉴于该协议由很明显的密态协议层及密态协议之上的逻辑层两部分组成，故需要对其进行分层处理，以满足本子课题对未来新协议的扩展性。例如将输入步骤通过分片全部代理到计算集群再由底层的通用密态协议具体负责算法流程的执行。为了较好分治不同技术路线的异构平台互通，暂分为上述两种方式进行处理，但目前来看这个分类不是绝对严格的，待未来有更复杂的处理时，可以进一步迭代更新该分类领域。

与此同时，由于隐私计算算法的发展非常迅速，不仅有新的算法协议在陆续发布，还有一些软硬件结合的算法开始出现。因此，从长期技术发展与科研角度来看，开放算法协议的设计还有不少难点：

● 算法定制工作量大。开放算法协议需要对算法内核进行打开设计，目前行业中对于隐私计算算法的实现有很多变种，若要满足行业全部需

HESS LR[5]: https://arxiv.org/abs/2008.08753

PCG-PSI[6]: https://eprint.iacr.org/2022/334

Bark-OPRF PSI[7]: https://eprint.iacr.org/2016/799.pdf

SecureBoost[8]: https://arxiv.org/abs/1901.08755

要，则可能要全部重新定义算法流程。这对于算法的研究开发而言难度很大，对于把创新算法作为重要知识产权的公司来说也是很大的挑战。

● 需定制完备的传输接口。为保障算法的正常运行，首先需要基于开放算法协议的算法，明确定义全流程通信传输接口与通信要求，防止出现接口缺失时无法实现算法正常运行使用的问题。

● 算法设计需要具备良好扩展性。为了让定义好的算法具备长期应用价值和可扩展性，算法设计时往往需要做到足够兼容和拓展，以ECDH-PSI算法为例，不仅椭圆曲线有多种选择，还要区分是国密或者非国密的，未来甚至还可能出现新的更好性能的曲线。因此，一个好的算法设计应当具备较好的兼容性，从而实现更好的可插拔。

因此，为更好适应隐私计算行业发展，本子课题将专注于定义算法协议通用流程与接口，对于异构算法协议将仅约束到网络通信接口层面，而不去约束平台内部实现协议。

本子课题将在PSI、联邦为主的专有算法流程，以及通用密态算法流程中分别选取典型代表在课题内进行细致的协议定制研究和讨论。同时，课题也将尝试对异构算法的互通思路进行探索和讨论，以及尝试选取实际的算法样例对异构互通结构进行考察。后续，待研究足够成熟时，再考虑进一步研究和细化具体协议定制流程。

（六）TEE 互联互通技术研究

1. 概述

目前，业界基于多方安全计算、联邦学习的互联互通技术研究工作正

在逐步开展，但基于可信执行环境的互联互通技术还处于起步探索阶段。TEE 技术以基于硬件生成的可信任根为基础，实现了对隐私数据完整性和安全性的全面保护，因其优异的性能和强大的可拓展性在政务、金融、运营商等众多场景中得到了广泛应用。随着国内外 TEE 方案越来越多，众多机构和企业都已经开始面临不同架构 TEE 方案混合使用的情况，互联互通的需求也随着产业和应用的成熟逐渐增加。

TEE 互联互通技术研究致力于不同 TEE 方案之间互联互通的实现方式，降低 TEE 技术在跨机构协作中的落地、可信应用开发和移植的成本，提升 TEE 易用性，以方便更多用户使用 TEE 技术。同时，TEE 互联互通也是隐私计算互联互通技术中不可或缺一部分，TEE 互联互通能够促进隐私计算互联互通的发展，帮助 TEE 技术路线和其他技术路线融合。

2. 研究目标

TEE 互联互通技术研究主要致力于探索如何屏蔽不同 TEE 技术架构和具体实现方案的底层细节差异，以实现基于异构 TEE 平台的可信应用和系统之间的互联互通。同时，TEE 互联互通课题还希望通过研究不同 TEE 实现方案中共存的基本安全特性来总结提炼面向互联互通 TEE 系统的基本安全要求和建议。

3. 研究内容

TEE 互联互通课题以实现异构 TEE 应用系统互联互通为目标，围绕如何抽象统一的远程证明流程，基于远程证明流程建立安全信道，规范安全信道上数据流通格式，以及规范 TEE 系统互联互通基本安全要求四个方面展开。TEE 互联互通课题主要包括以下四个方面的研究内容：

（1）统一远程认证流程规范。这部分主要研究如何对不同TEE方案的远程证明相关接口、报告数据结构、校验规则做规范化抽象。

（2）统一的安全信道建立流程。这部分主要研究如何把远程证明流程和密钥协商流程、TLS流程等相结合，建立安全信道，为实现不同TEE架构之间数据安全流通提供基础。

（3）统一的TEE应用系统数据流通格式。这部分主要研究如何规范统一的TEE应用数据格式，实现数据层面不依赖TEE实现的、安全的、高效的互联互通。

（4）TEE系统互联互通基本要求。这部分研究可信应用乃至计算系统在互联互通前需满足的基本安全要求。

4. 设计原则

考虑到 TEE 系统既具有 TEE 的独特性，又具有通用系统的特点，建议 TEE 互联互通系统设计遵守以下基本安全原则：

●安全算法要求：TEE 应用中应该使用业界共识的安全的密码学算法，并给出相关参考。

●远程证明增强安全：互联互通的双方 TEE 平台必须通过远程证明校验之后，才能信任运行在 TEE 中的可信应用程序计算逻辑；必须结合远程证明机制鉴别可信应用程序身份之后，才能授权操作或者安全通信。

●系统完整性安全：互联互通的 TEE 平台需利用可信启动或安全启动流程保证系统的完整性，利用已证明的可信内核和可信软件组合需要的业务能力。

●安全运维辅助：互联互通的 TEE 平台需利用安全配置，安全审计、

安全运维保证系统全流程和生命周期安全。

5. 研究思路

基于上述研究内容，提出如图 12 所示异构 TEE 互联互通架构模型：

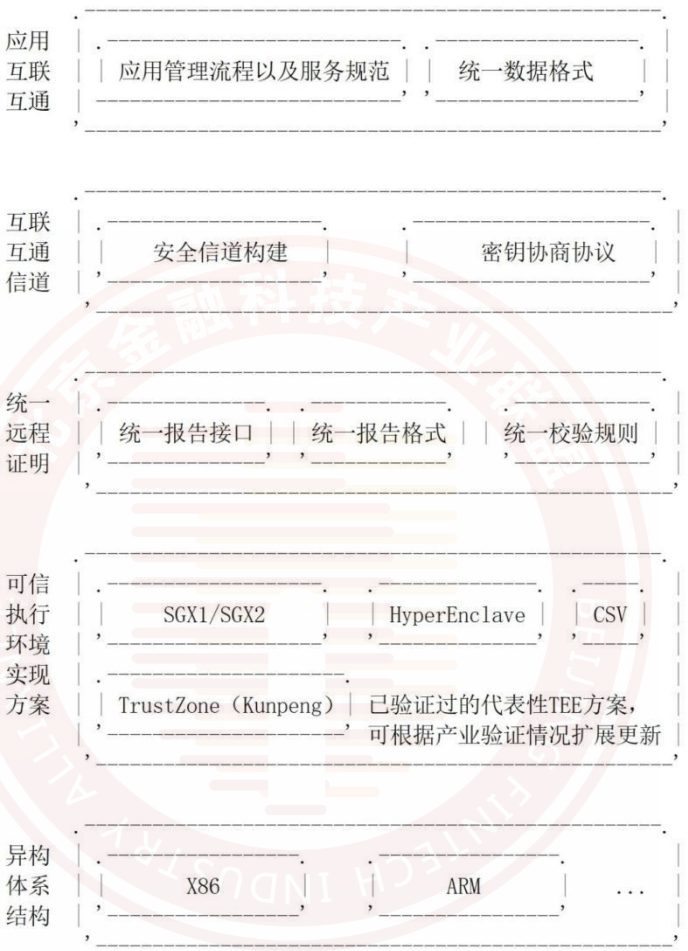

图12 异构TEE互联互通架构模型图

框架层说明：

● 异构体系结构：表明不同 TEE 实现方案的硬件体系结构基础，从 CPU 维度进行划分。

● 可信执行环境：主要参考业内主流 TEE 方案体系，并基于不同路线 TEE 技术方案选取具有代表性的 TEE 实现方案。目前在课题内验证过的方

案包括 Intel SGX、Huawei Kunpeng TrustZone、Ant Group HyperEnclave 和 Hygon CSV 等。

● 统一远程证明：用于屏蔽不同 TEE 方案的远程证明细节，为互联互通提供基础。

● 互联互通信道：基于统一远程证明，用于 TEE 系统之间互联互通安全信道的建立。

● 应用互联互通：基于安全信道的应用层面互联互通相关细节要求。

其中应用管理流程以及服务规范参考第三章（一）管理面元素与接口研究、第三章（四）传输接口与报文研究等章节内容。

四、互联互通关键技术点攻关

第三章提及的6项技术子课题在研究推进的过程中，对其中所涉及的关键问题进行了深入的探讨与攻关，提炼形成了“管理面最小必要元素设计”“多方资源访问控制协同策略”“DAG & CONF 的通用化设计”“流程调度互通设计”“组件容器化加载方案”“基于最小化约束的算法容器设计”“隐私计算安全算子服务设计”“传输层同步异步兼容设计”“异构平台开放算法协议设计”“TEE 统一远程证明流程设计”等 10 大关键技术点。表3列举了互联互通各项技术与关键技术点的对应关系。本章将对这些技术点的研究成果进行汇总呈现。

表 3 互联互通各项技术及其对应关键技术点

互联互通技术	互联互通关键技术点
管理面元素与接口研究	管理面最小必要元素设计
	多方资源访问控制协同策略
	DAG&CONF 的通用化设计
流程调度接口与算法容器加载研究	流程调度互通设计
	组件容器化加载方案
	基于最小约束的算法容器设计
安全算子接口与服务化研究	隐私计算安全算子服务设计
传输接口与报文研究	传输层同步异步兼容设计
异构算法协议研究	异构平台开放算法协议设计
TEE 互联互通技术研究	TEE 统一远程证明流程设计

（一）管理面最小必要元素设计

隐私计算互联互通首先需要提炼出最小必要的实体元素，参与互联互通的各参与方需要对齐这些实体元素的定义和数据结构。“管理面与接口研究”子课题对各实体进行了定义，并对实体应具备的通用属性形成共识，对与节点互通无关的个性化属性不作限制。具体包括以下实体：

● 节点（Node）：隐私计算生态中的抽象功能单元，用来指代由机构或组织部署的隐私计算平台。

● 数据集（Dataset）：可供参与隐私计算的数据资源。

● 项目（Project）：面向特定目标的，提供一项独特产品、服务或成果的隐私计算方案，该实体为可选项。

● 组件（Component）：独立执行隐私计算任务的模块单元，其经过封装、符合开放接口规范、可以完成某个特定计算或算法，可独立部署。

● 流程（Flow）：采用 DAG 结构定义的、可编排的隐私计算作业运行模板。

● 作业（Job）：一个隐私计算流程通过配置运行参数后的运行实例。

● 任务（Task）：组件运行实例的载体，每个运行实例通过任务来管理。

● 模型（Model）：指通过隐私计算技术训练完成，可用于进一步推理的模型，或是已有的现成模型。

结合上述实体定义，实体间的关系图如图 13 所示。首先，参与互联互通合作的异构平台作为节点加入互联互通网络，一个节点可以创建多个

项目，每个项目中可能包含多个参与该互联互通项目的多个节点，通过项目实体来统一管理项目中相关的节点、数据集和模型等资源，项目中多个合作的节点可以创建多个流程，流程运行多次生成多个作业，不同作业可以按照“CONF 通用化设计”配置不同的运行时参数，一个作业可能由多个任务组成，任务承接作业中对应的运行时参数配置，每个任务通过调度底层组件对应的算法资源进而执行任务，生成模型、报告等隐私计算结果。其中，每个组件都是独立的算法模块，例如 PSI、LR 等，组件可以按照“DAG 通用化设计”提前配置调度顺序、输入输出关系等，从而编排形成一个具有完整计算逻辑的流程，作业和任务按照 DAG 配置和 CONF 配置调度和运行相应资源，生成对应结果。

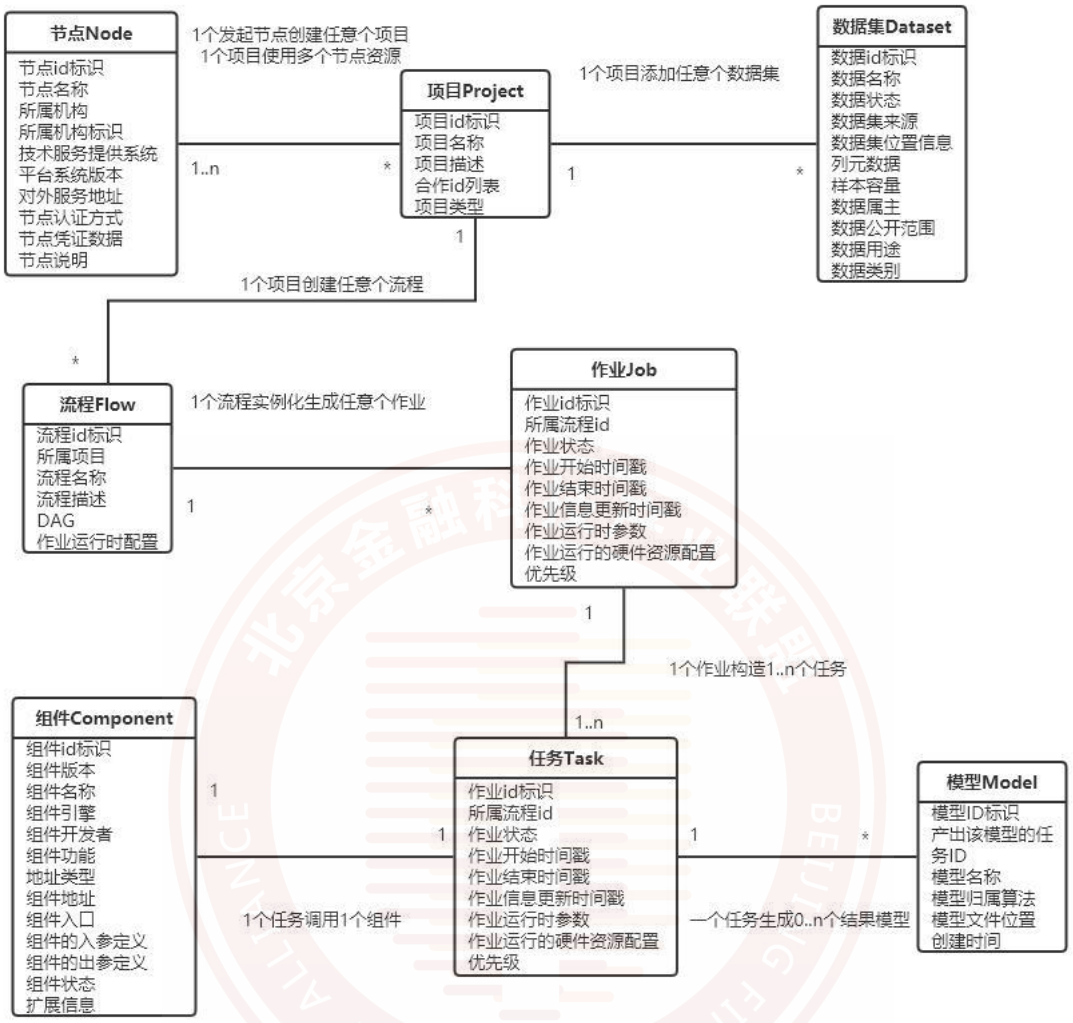

图 13 管理面最小必要元素关系图

围绕上述最小必要元素，设计隐私计算互联互通必需的管理操作包括：节点互通管理、数据集互通管理，项目互通管理、组件互通管理、流程互通管理、模型互通管理。

(1) 节点互通管理

节点互通管理流程约定不同隐私计算平台间如何互相发现节点、建立合作、同步节点变更。节点互通管理相关操作可包括：节点签约、节点

解约、合作意向确认、合作意向查询、节点信息查询等。关键操作时序图如图 14 所示。

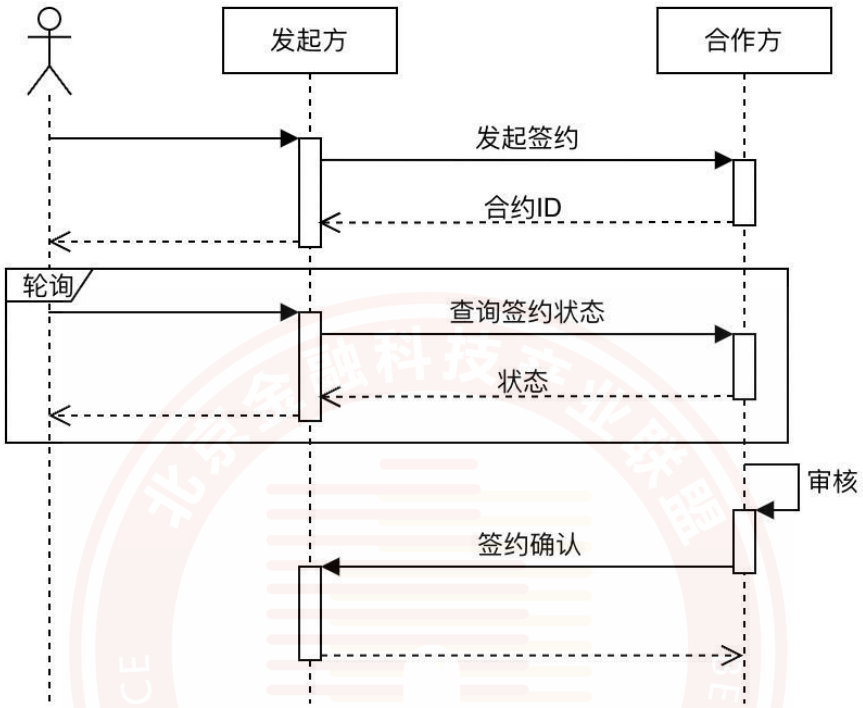

图14 节点互通关键操作时序图

(2) 数据集互通管理

数据集互通管理流程约定不同隐私计算平台间的如何互相发现数据资源、互相授权数据资源、解除授权。数据集互通管理相关操作包括：公开数据集列表查询、公开数据集明细查询、数据集授权申请、数据集授权确认、数据集授权状态查询等。关键操作时序图如图 15 所示。

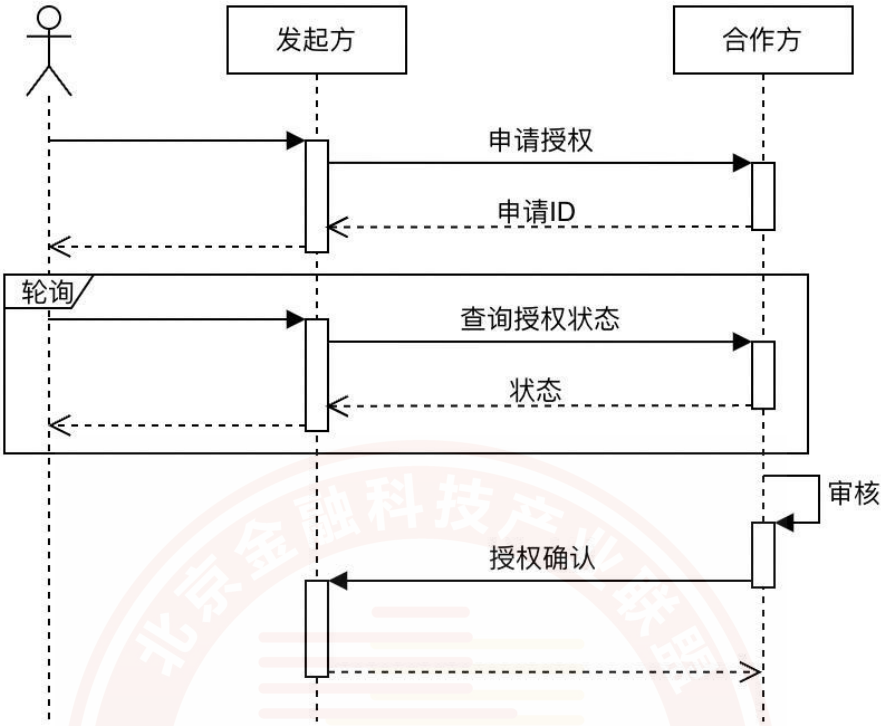

图15 数据集互通关键操作时序图

(3) 项目互通管理

项目互通约定不同隐私计算平台间的如何创建合作项目、同步项目信息。项目互通管理相关操作可包括：合作项目列表查询、项目创建、项目信息查询、项目状态更新等。

(4) 组件互通管理

组件互通管理相关操作可包括：公开组件列表查询、公开组件信息查询、组件注册、组件列表查询、组件信息查询、组件心跳保活、组件健康检查、组件参数验证。关键操作时序图如图 16 所示。

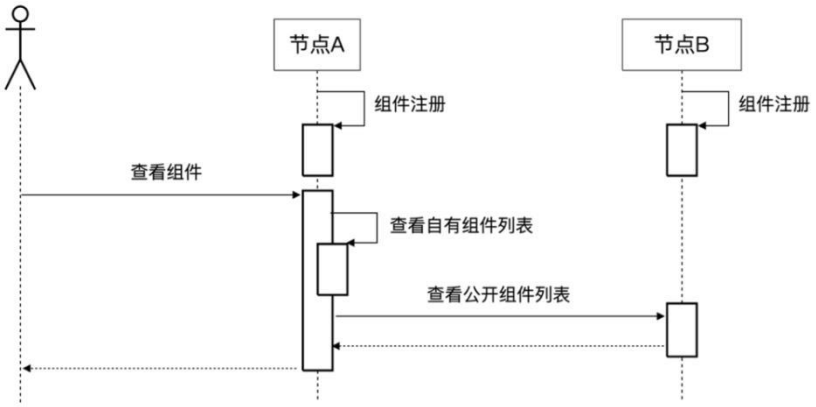

图16 组件互通关键操作时序图

(5) 流程互通管理

流程互通管理相关操作可包括:流程审批、审批确认、审批状态查询、流程创建、流程更新、流程列表查询、流程信息查询、流程删除。关键操作时序图如图 17 所示。

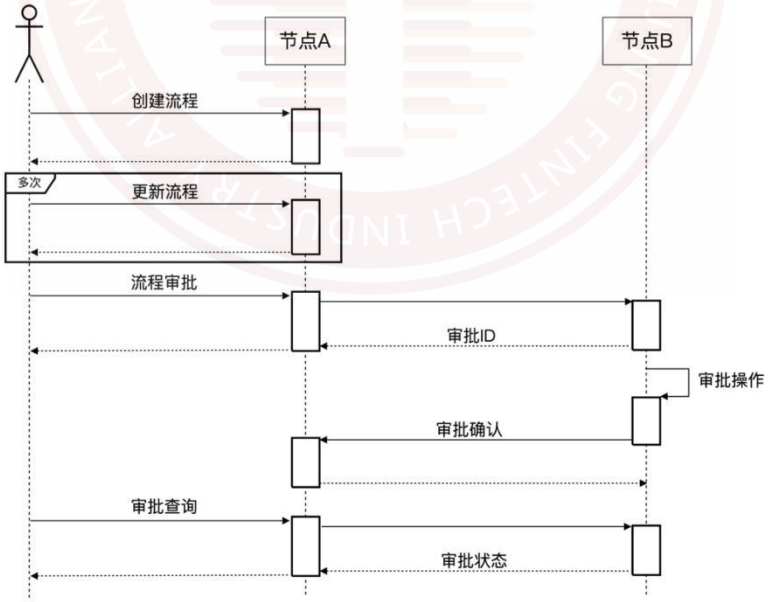

图17 流程互通关键操作时序图

(6) 模型互通管理

模型互通管理相关操作可包括：模型列表查询、模型信息查询等。

（二）多方资源访问控制协同策略

在互联互通的情况下，各家厂商之间的平台是异构化的：一方面，需要保障各厂商、各参与节点的安全策略彼此独立；另一方面，又能建立一套有效的机制，各方之间能相互协同。此外，还要充分考虑各节点使用不同的安全策略、不同资源访问控制粒度上的灵活性，以及实施落地上的可操作性。

在业内现有技术实现上，还未有能解决异构隐私计算平台在资源访问控制上的通用方法。对于各厂商现有实现方法中采用的资源访问控制机制，主要依托特定厂商的隐私计算系统产品，通过在各节点间建立对等资源访问控制的机制来实现。

1. 技术特点

● 创新性：形成异构隐私计算平台之间多方资源访问控制的有效方法；

● 灵活性：面向隐私计算的各种资源访问控制的策略可以灵活订制；

● 通用性：适用于异构隐私计算平台互联互通，适用于点对点、中心化的多种网络结构，适用于双方或多方间的安全控制策略；

● 可维护性：可满足安全需求的变化，提供安全策略的维护、升级、兼容能力；

● 可实施性：对现有隐私计算平台具有较低的功能侵入性，对已部署节点在规范化资源访问控制机制过程中，可通过灵活的安全策略，确保分

阶段实施过程中的兼容性问题。

2. 创新点

在隐私计算互联互通的场景下,构建一个平台节点间的资源访问控制方法，主要技术创新点包括：

● 在系统认证和资源分类的基础上，通过对资源类型分级的方法，按资源等级分级授权；

● 在鉴权中，各方都采用最高资源等级的凭证，实现安全策略的自动对齐，实现各节点安全策略的内聚，以此解耦多方节点间安全策略强关联的问题；

● 确保了计算执行阶段能，能够对资源访问控制进行检查；

● 其松耦合的安全策略,实现异构隐私计算互联互通中安全策略的多样性，具备通用性、可维护性、可扩展性。

3. 处理步骤

(1) 构建前提条件

1、本方案建立的前提是在隐私计算节点之间，已经完成节点间的网络联通和身份认证。

2、基于隐私计算行业内的共识，对隐私计算系统所涉及的资源进行统一分类，包括节点、项目、数据集、流程、作业、任务、模型、组件八类资源。

(2) 构建授权关系

1、各节点内对资源进行描述，并对各类资源进行分级，各节点可独立分级，无需多方统一。资源由三个字段进行描述，如下：

- ResourceType：资源类型，业内统一资源类型；
- ResourceID：资源 ID，各节点可独立编码；
- ResourceLevel：资源级别，各节点可独立分级。

各节点可独立定义资源分级，示例如表 4 所示：

表 4 资源分级示例

级别	资源类型
1	节点
2	数据集、模型
3	项目
4	流程
5	作业
6	任务

2、在各资源持有方，建立独立的安全策略，其资源访问控制的授权、鉴权策略及处理，集中在资源所属节点上，资源使用方不需要关注具体安全策略，根据安全访问控制所需要的信息要素，形成标准化的凭证，在节点之间，传递标准的凭证即可。

图 18 描述了构建安全策略与通过安全策略授权的设计方法。安全策略是在资源与授权实体（资源使用节点）之间，建立实体对资源的访问控制规则，具体的安全策略，资源持有节点可以自行定制，比如可以制定全局安全策略，分组安全策略，以及单个实体单独制定的安全策略，在建立资源与授权实体的关系后，在关系上加入限制条件，即可构建出具备资源访问控制的安全策略。

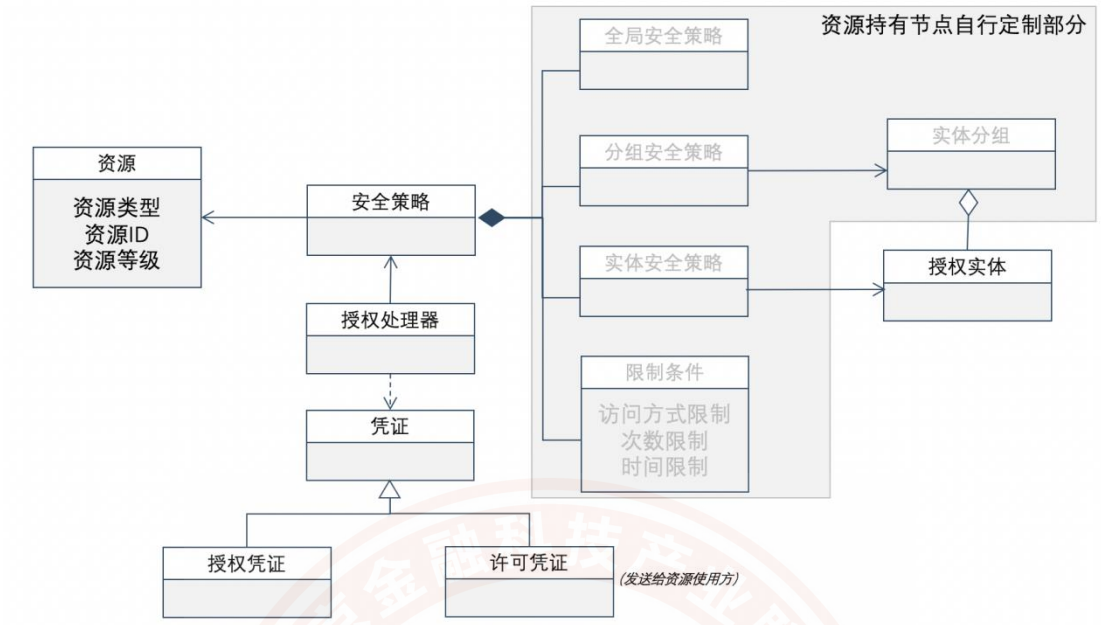

图 18 安全策略设计示例图

授权操作，由授权处理器完成，授权处理器是根据安全策略，产生一组凭证，分为授权凭证以及许可凭证，授权凭证保留在资源持有方，在鉴权时使用，许可凭证在授权过程中，由资源持有方发送给资源使用方，资源使用方在鉴权时，根据许可凭证，获取资源访问权限。

对于各方之间传递的凭证，建立凭证的定义规范，以确保能够在多方之间进行协同，凭证内容应包括以下信息：

- Token：用于核验的令牌，颁发给资源使用方的唯一授权标识，与授权主体、资源绑定；
- ResourceLevel：资源等级；
- ResourceType：资源类型；
- ResourceId：资源 ID；
- ResourseNodeId：资源提供方的节点 ID，用于标识物理节点；

- ResourseInstId：资源提供方的机构 ID，用于标识实体身份；
- RequestNodeId：资源使用方的节点 ID，用于标识物理节点；
- RequestInstId：资源使用方的机构 ID，用于标识实体身份；
- TimeLimit：限制条件项，访问时间限制，可选；
- TimesLimit：限制条件项，访问次数限制，可选。

3、在隐私计算平台之间，由合作流程、授权、审批产生的授权关系，来建立多方资源访问控制凭证，资源所属节点记录并持有资源的授权凭证，资源使用节点记录并持有授权许可凭证（见图 19）。

图 19 多方之间建立资源授权关系流程图

(3) 访问鉴权

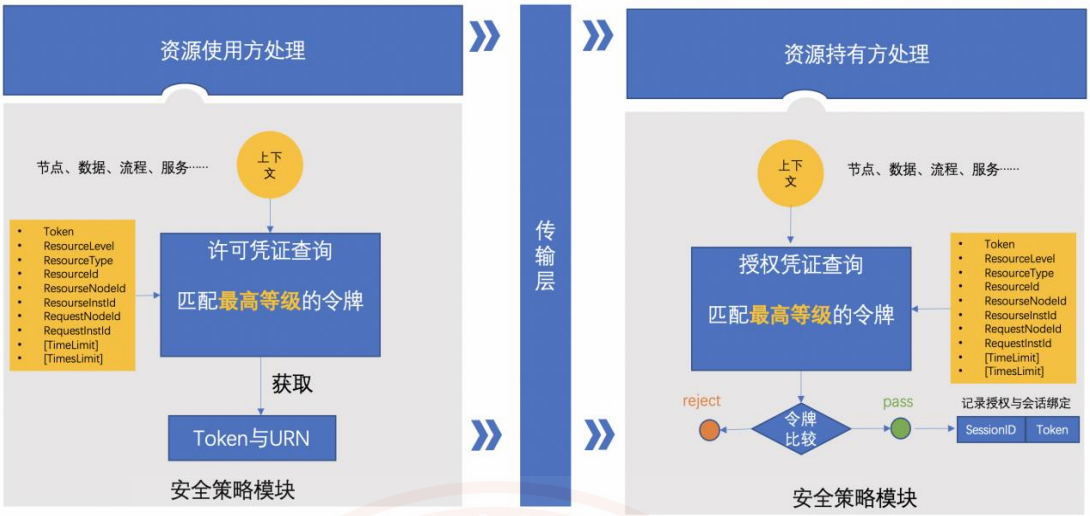

图 20 多方之间资源访问鉴权流程图

1、多方资源在使用时，资源使用方根据上下文信息中所需要使用的外部资源信息，检索出本地持有的**最高资源等级的许可凭证**，与资源使用请求一起发送给资源持有方（见图 20）；

2、资源持有方获取资源使用请求后，根据请求中包含的资源信息，从本地检索出**最高资源等级的授权凭证**，并与请求中的许可凭证进行核验，通过资源级别的比对，确保不存在纵向越权，通过资源检索，确保不水平越权。

3、资源持有方完成凭证的核验，最后根据核验结果进行访问请求的许可判断，如果核验通过，则记录会话与令牌的绑定关系。

4、计算执行阶段令牌有效性校验，资源持有方在对任务的资源访问完成授权后，在本地记录任务所对应的会话 ID 和授权令牌。

(4) 与互联互通架构结合

在异构隐私计算互联互通技术研究中，业内对互联互通架构提出了一个满足行业互联互通标准的分层架构方案，从上往下，分别是管理

面，控制面和数据面。

1、管理面对各实体的定义、实体应具备的通用属性形成共识，并定义了各类资源实体的信息的交互和授权接口，以标准化的方式，来满足当前隐私计算资源信息层面互联互通的标注流程。

2、控制面标准化定义多方在作业、任务层面的协调机制和标准化接口，实现作业、任务在不同隐私计算平台之间的协调。

3、数据面是将要执行的算法算子组件以容器化的方式进行执行，以数据存储、通信传输、安全算子方面的标准接口和模块化设计，来实现计算层面的互联互通。

多方资源访问控制机制，需要层管理面、控制面、数据面的三层一起实施，协调实现，图 21 显示了各模块和层面的功能职责，以及整体的协调机制。

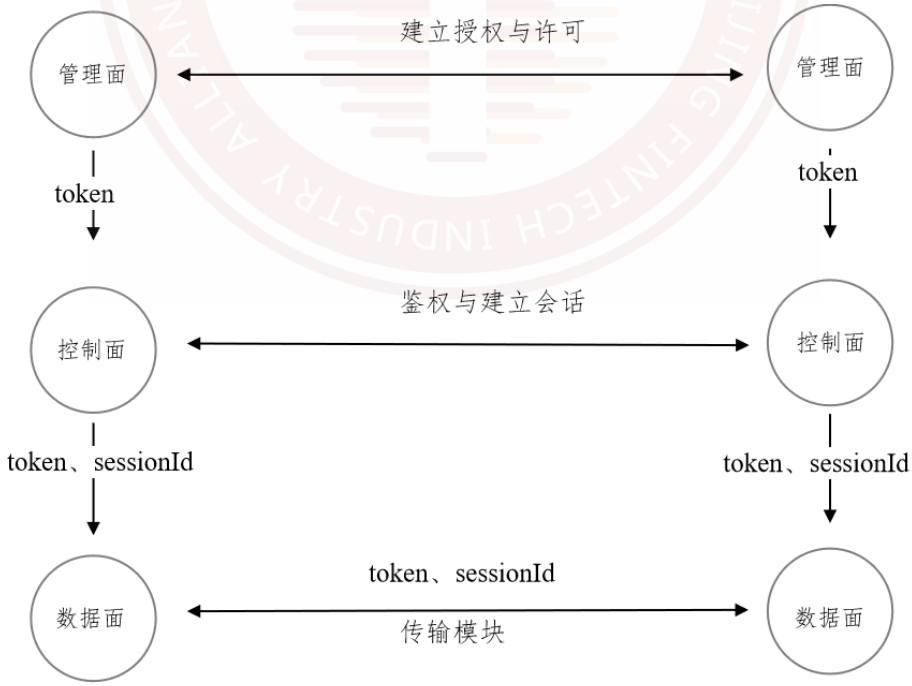

图 21 互联互通分层架构下的资源访问控制图

如图 21 所示，在管理面，需要由管理面建立资源的授权和许可关系，各方按照管理面流程，以 token 的形式，对资源的授权和许可记录进行登记。

控制面控制作业和任务的启动与执行，依托控制面接口的实现，对资源的方案完成鉴权，并根据控制面协调的任务编号，建立关联会话编号（sessionId），在互联互通架构中，各任务的执行以独立的容器化进行，通过 sessionId 来确保多个任务同时执行时，多方之间建立正确的关联。

在数据面，以容器化的方式执行计算任务，使用 sessionId 保持不同参与方之间的建立会话。使用 token，再次进行令牌的有效性检查，确保任务在执行阶段资源访问控制的有效性。

（三）DAG & CONF 的通用化设计

1. DAG & CONF 的通用化设计思路与原则

由于各家隐私计算产品的异构性，隐私作业的任务参数及任务流程的配置方法及形式也大相径庭，这导致各家产品无法正确解析对方发送来的作业配置信息，从而无法保证互联互通过程中隐私计算作业的正确运行。因此有必要形成一套业内统一且标准的作业配置以及作业流程的描述形式，来规避各家因作业配置的差异性而导致的互联互通失败。

同时，由于各家产品中隐私计算算法的异构性，算法所需的输入或输出元素以及算法需要配置的参数具有显著的区别。因此在对作业配置以及作业流程的描述形式进行通用化设计时应当尽量兼容各家产品，并满足之后新算法参数的可拓展性。

根据上述原则，我们对互联互通作业的流程 DAG 与配置 CONF 做出如下的通用化设计。

2. DAG & CONF 的通用化设计参考实现

(1) 流程 DAG 定义

流程 DAG 是用来描述流程中组件依赖关系的语言,一旦流程确定则内容不变,DAG 的配置文件采用 JSON 格式,整个配置文件为一个 JSON 对象。DAG 模板结构如图 22 所示。

```
{
    "components": [{
        "name": "${component_name}",
        "code": "${component_code}",
        "version": "${component_version}",
        "module_name": "${component_module}",
        "input": [{
            "type": "${type}",
            "key": "${component_name}.${key}"
        }
        //示例
        {
            "type": "model",
            "key": "intersection_1.model_0"
        },
        {
            "type": "train_data",
            "key": "intersection_1.data_0"
        },
        {
            "type": "validate_data",
            "key": "intersection_2.data_0"
        }
        //
        ],
        "output": [{
            "type": "${type}",
            "key": "${key}"
        }
        //示例
        {
            "type": "model",
            "key": "model_0"
        },
        {
            "type": "train_data",
            "key": "data_0"
        },
        {
            "type": "report",
            "key": "report_0"
        }
        //
        ]
    }],
    "version": "${version}"
}
```

图22 DAG模板结构图

模板说明：

a. 组件列表（components）

component_name由建模人员自行定义，需在单个DAG中保证唯一，用于DAG中依赖关系的引用。component_code与component_version分别表示

组件ID标识与组件版本，module_name表示组件使用的功能模块，version表示当前DAG配置文件的版本。

b. 输入（input）

输入数据由

[{"type":"${type}","key":"${component_name}.${key}"}]

表示，其中${type}为输入数据类型，例如training_data, model, report等数据类型，${key}为DAG内的唯一数据引用标识，"${component_name}.${key}"表示对上游组件输出数据的依赖。

c. 输出（output）

输出数据由[{"type":"${type}", "key":"${key}"}]表示，其中${type}为输出数据类型，数据类型参考输入配置项，${key}为DAG内的唯一数据引用标识用于下游数据输入的引用。

(2) 作业 CONF 定义

作业CONF用于设置各个参与方的信息，作业系统参数、运行时任务系统参数以及配置信息，CONF的配置文件采用JSON格式，整个配置文件为一个JSON对象。CONF模板结构如图23所示。

图23 CONF模板结构图

模板说明：

a. 作业调度方（initiator）

描述作业调度方的role和node_id。

b. 所有参与方信息（role）

在role字段中，每一个元素代表一种角色以及承担这个角色的node_id。每个角色的 node_id以列表形式存在，因为一个任务可能涉及多个node担任同一种角色。

c. 作业系统参数（job_params）

配置作业运行时的主要系统参数，作业系统参数参考如下表所示，当系统参数应用于所有参与方时，使用common范围标识符；仅应用于某参与方时，使用role范围标识符，(role:)role_index（如role.guest[0]）用于定位被指定的参与方，直接指定的参数优先级高于common参数。作业系统参数建议包括：

- 作业类型，如训练作业train和预测作业predict
- 作业申请的总CPU核数上限
- 作业申请的总内存数上限
- 作业运行时任务状态收集模式，如主动上报模式callback与轮询模式poll
- 模型id，当作业类型为predict时必选
- 模型version，当作业类型为predict时必选

d. 任务系统参数以及配置信息（task_params）

配置任务运行时的系统参数以及配置信息，任务系统参数参考如下表所示，不同组件的配置信息由组件自行定义。范围标识符应用规则与作业相同。任务系统参数建议包括：

- 任务申请的总CPU核数上限
- 任务申请的总内存数上限
- 任务超时时间，单位秒
- 任务失败自动重试最大次数

（四）流程调度互通设计

在一个完整的隐私计算作业执行过程中，流程调度的目的在于将隐私计算作业配置（DAG 格式）下发到各个参与方，在多个参与方之间协调资源、同步状态，按照流程编排顺序协同完成作业的执行。流程调度处在作业编排和作业执行之间，起到一个承上启下的作用，因此在隐私计算行业内，不同平台在流程调度上的设计也会因上下游组件的一些设计差异而各不相同。为了实现互联互通的总体设计目标，本子课题在流程调度互通设计关键技术上，采用“概念抽象、共性打通”的思想，抽象出流程调度在设计模式上的共性，并将共性部分统一规范定义，仅约定作业调度执行最小必需的流程步骤，实现彼此打通。

流程调度互通设计的方案由流程调度概念定义、流程调度时序设计和流程调度系统建设建议三个部分组成。流程调度概念定义，明确了流程调度环节的名词定义，并阐明了概念的作用范围和含义。流程调度时序设计，对调度操作进行规范化定义，设计了简明的作业参与方角色，

约定了作业执行的全生命周期管理规范。流程调度系统扩展，在满足流程调度互通需求之外，提供了隐私计算作业执行的扩展性建议。

1. 流程调度概念定义

隐私计算流程调度过程中的概念（或称为实体）包含静态和动态两类，静态的实体包括：组件、流程和资源，可以类比为面向对象编程中的“类（Class）”，动态的实体包括：作业、任务，可以类比为面向对象编程中的“对象（Object）”，是一个运行时的概念。

2. 流程调度时序设计

流程调度系统接收上层的作业请求，协调各个参与方节点运行作业，通过将一个作业拆分成若干个任务,按照任务的组合顺序来执行每一个任务。在任务被调度开始执行前，由每个任务的配置驱动，启动对应的算法组件容器，实现隐私计算平台对异构算法组件的调用。

在流程调度环节，隐私计算参与方分为两类角色（见表 5）：

表5 流程调度隐私计算参与方角色

角色名	角色定义
调度方	负责协调所有参与方任务的角色，调度方可以为独立的第三方节点，也可以为其中一个参与方节点，在任务创建时选择对应节点来确定。
参与方	通常代表一个隐私计算节点，在某个隐私计算任务中提供计算能力的角色。

隐私计算作业执行必需的流程调度操作包括：创建作业、启动作业、查询作业状态、停止作业、任务回调、任务异常停止。

(1) 创建作业

由发起方创建作业配置，下发到调度方的流程调度层，流程调度层在接收到该作业后，将其分发给各个参与方的流程调度层。然后由调度方协调每个参与方节点，进行作业的下一步启动运行（见图 24）。

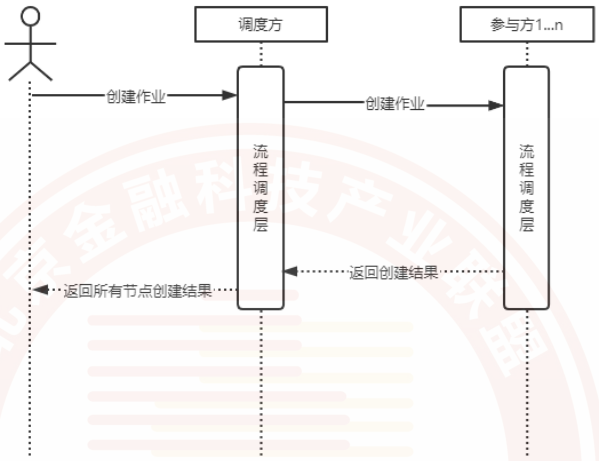

图24 创建作业关键操作时序图

(2) 启动作业

各个参与方接收到创建作业后，由调度方来协调各个参与方并发起启动作业请求，每个参与方的调度层接收到该请求后，将作业拆分成若干个任务，然后将每个任务下发到对应的算法组件服务（见图 25）。

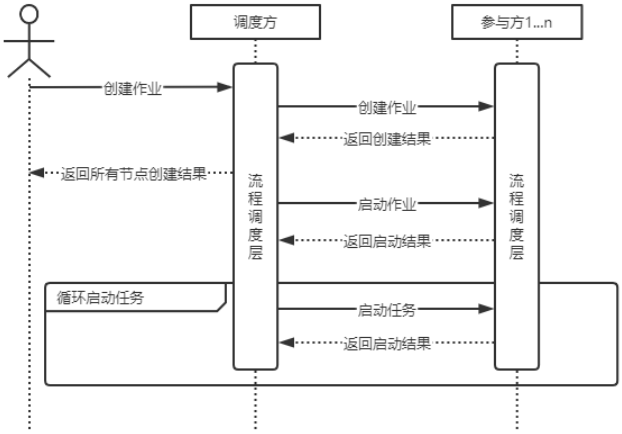

图25 启动作业关键操作时序图

(3) 查询作业状态

由某个参与方发起作业状态查询请求，发送到调度方的流程调度层，调度方在接收到该请求后，向各个参与方发起查询作业状态的请求，然后汇总所有节点的作业状态后，返回给请求的发起方（见图 26）。

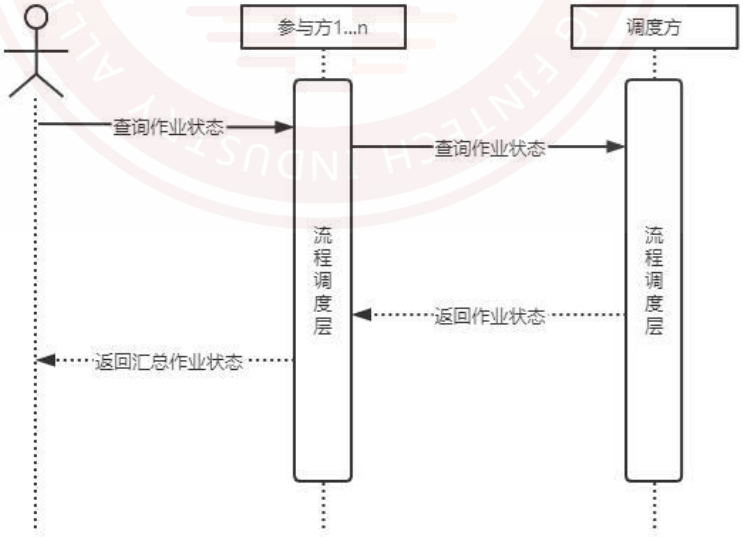

图26 查询作业关键操作时序图

(4) 停止作业

由某个参与方发起作业停止请求，发送到调度方的流程调度层，流程调度层在接收到该请求后，向各个参与方发起停止作业请求，每个参与方接收到停止作业请求后，向正在运行的任务所对应的算法组件发起停止任务请求，然后将作业停止结果返回给请求发起方（见图 27）。

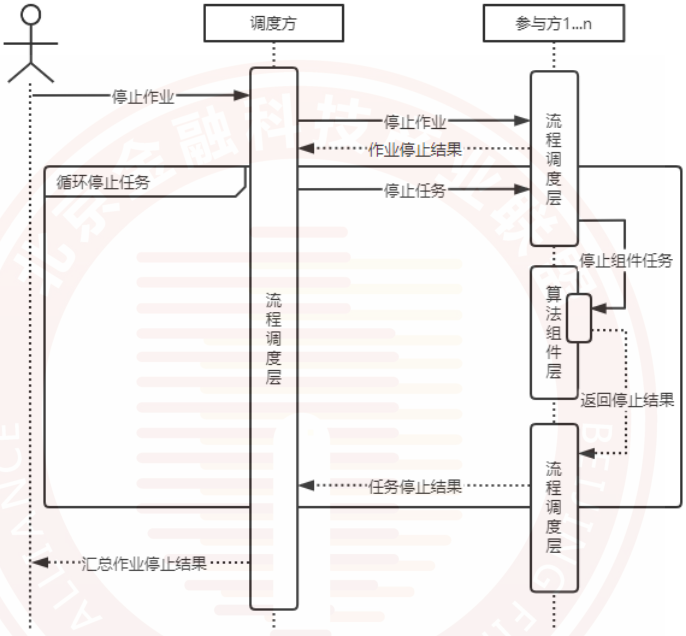

图27 停止作业关键操作时序图

(5) 任务回调（poll）

主动查询（轮询）方式的任务执行结果获取：各个参与方在运行过程中将任务的状态、运行的中间结果等等，通过回调的方式反馈给己方的流程调度层，然后由调度方向各个参与方主动查询任务的回调结果（见图 28）。

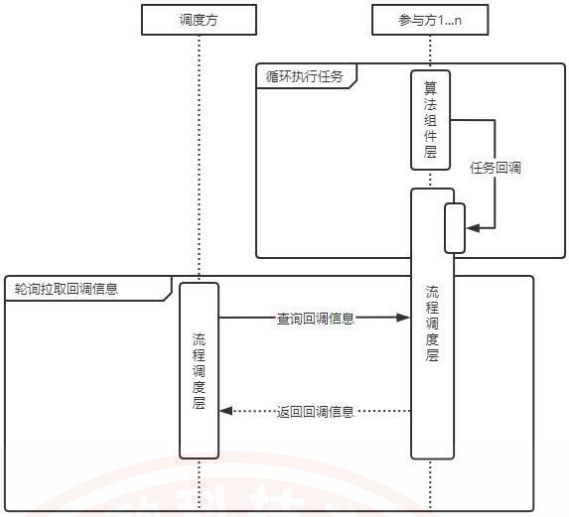

图28 任务回调(poll)关键操作时序图

(6) 任务回调（callback）

被动等待回调方式的任务执行结果获取：各个参与方在运行过程中将任务的状态、运行的中间结果等，通过回调的方式，反馈给己方的流程调度层，然后主动推送任务的回调结果给调度方（见图 29）。

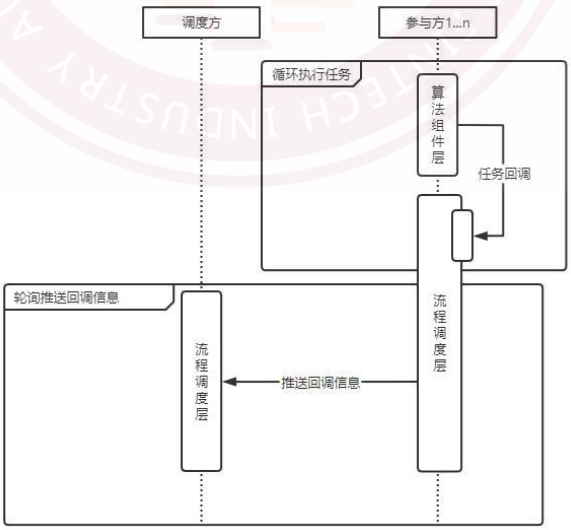

图29 任务回调(callback)关键操作时序图

(7) 任务异常停止

当任务运行过程中，由于各种原因导致当前运行的任务失败时，运行任务的算法组件将失败的任务状态同步给流程调度层，然后流程调度层再将该失败状态同步给调度方，然后由调度方再发起停止作业请求（见图30）。

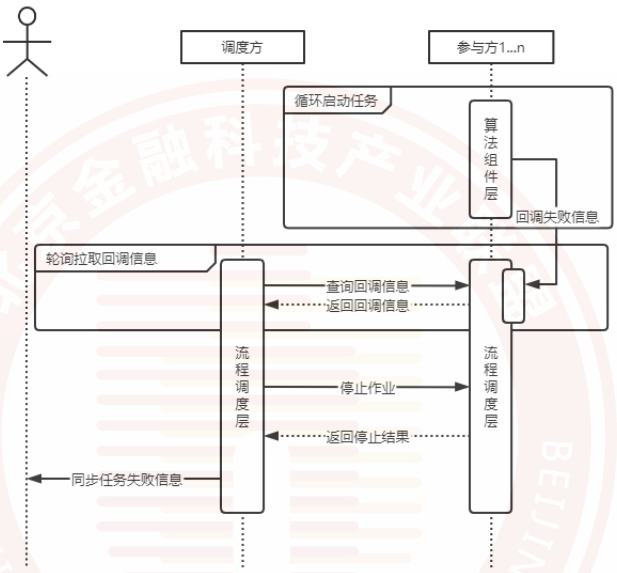

图30 任务异常停止关键操作时序图

3. 流程调度系统扩展

流程调度除了保障隐私计算作业的多参与方协同顺利执行，在很大程度上也决定了作业执行的并行度、系统总体容量。因此平台在满足基本作业调度功能之外，建议可通过作业队列、动态调度等机制实现隐私计算作业的高效执行。

流程调度可以支持多种调度策略，如：FIFO、Capacity、Fair、优先级保障等，满足在生产应用时对调度策略的灵活要求。同时，流程调度作为作业执行的指挥官，需要保证系统的高可用性，通过分布式的子系统负

载共同维护统一的调度服务能力，保证整体系统的稳定运行。

（五）组件容器化加载方案

本子课题研究了组件容器化加载的详细方案，提供标准化的组件容器加载流程和关键技术，提供统一的镜像定义规范和组件名称定义规范，最后给出组件实现和组件管理服务的详细容器加载工作流程。

1. 组件容器加载整体设计

首先组件容器化加载的整体关系结构示意图如图 31 所示。

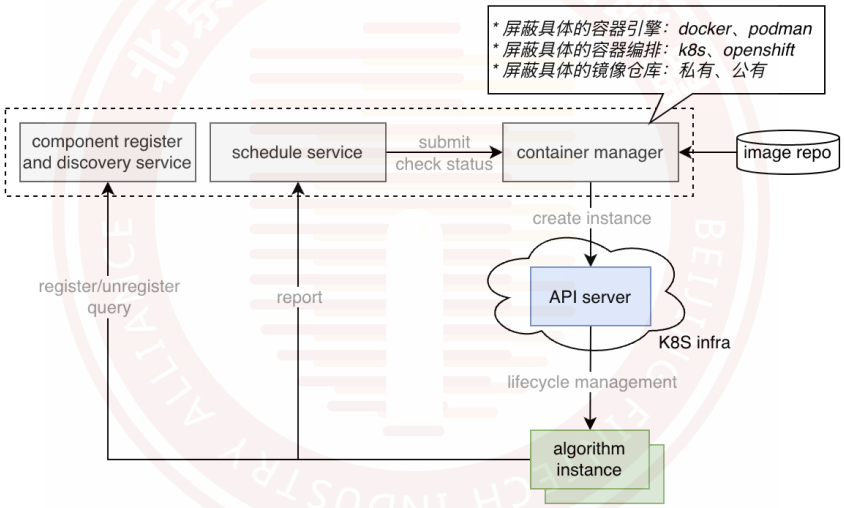

图31 组件容器化加载关系结构图

与隐私计算平台相关的模块包括：

(1) component register and discovery service：组件注册与发现。用于管理组实例的服务，提供组件的注册和查找服务。组件实例启动后进行注册，实现算法服务发现和路由

(2) schedule service：调度服务。对算法作业、任务进行调度

(3) container manager：容器管理器。对底层容器编排能力的封装，接收调度服务的请求

注：上述各模块是逻辑上的抽象，不对应具体的服务，物理形态不作限制。

组件通过加载机制，将组件的镜像加载成为组件实例，并将响应的接口注册到注册和发现服务上，供任务进行调用。任务和组件实例之间，可以是多个任务对应一个组件实例，也可以是一对一形式。组件和任务的关系示意图如图 32 所示。

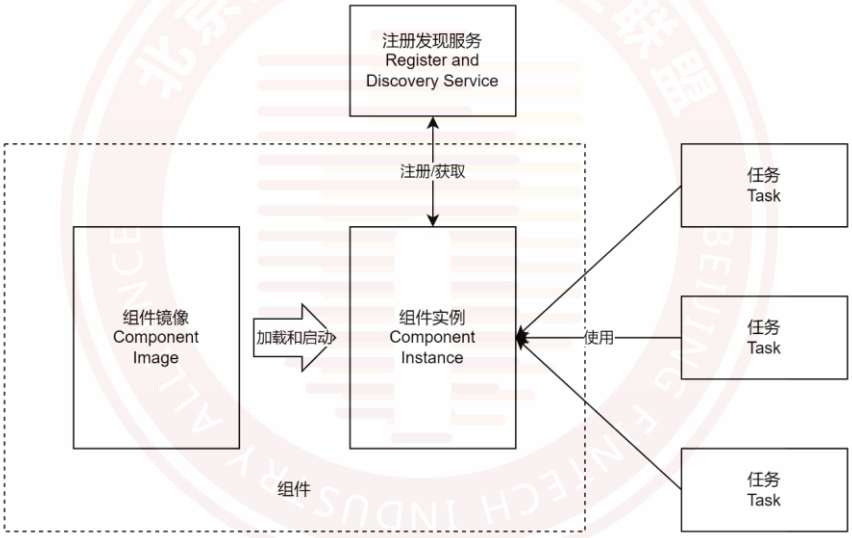

图32 组件和任务关系示意图

其中组件镜像（Component Image）指用于表示组件的打包后的容器镜像文件。组件实例（Component Instance）指在加载和启动后用于提供组件功能的进程或者服务。组件实例可以是常驻或者单次形态进行运行，但组件的工作流程和步骤都保持一致。

● 常驻形态：组件经过加载后，形成组件实例将持续存在系统中，供一个或者多个任务来进行使用。组件启动，可以在组件第一次使用时进行加载，或者在系统启动时完成拉起。组件停止，可以在系统停止时，或者根据空闲情况进行停止。

● 单次形态：组件随着任务的执行开始而启动，随着任务的执行完成而进行停止。

下面进一步描述标准组件加载管理机制，由组件管理服务和组件支撑服务两部分组成。标准组件全生命周期通过组件容器管理器进行管理。容器加载结构如图 33 所示。

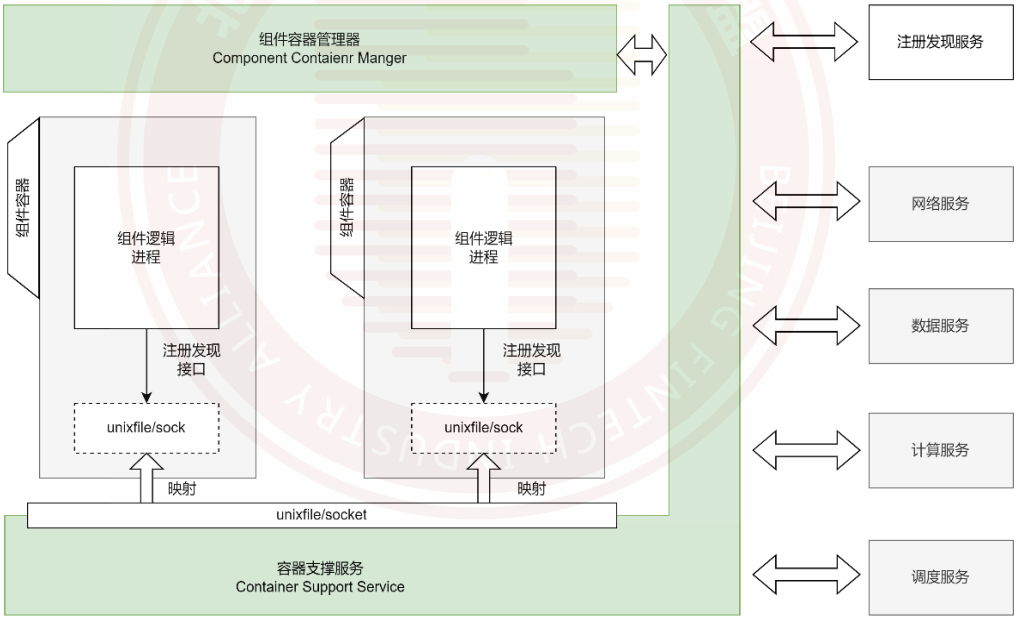

图 33 容器加载结构图

其中：

● 组件管理服务 Component Management Service：用于管理组件及其实例的服务。负责管理组件所运行的容器的管理，与容器平台（例如，Docker，Swarm 和 Kubernetes）进行对接，完成容器的编排管理。

● 组件支撑服务 Component Support Service：用于向组件提供基础操作能力的服务。用于对组件容器内所使用的功能，提供标准的服务接口。其会将基础服务通过 unixfile 文件形式将基础接口功能映射到容器内部并接收并响应组件所提供的请求。

2. 镜像定义规范和组件名称定义规范

(1) 镜像定义规范如下：

a. 镜像命名：基本格式为 /<developer>/<image-name>:<tag>

● developer：组件的开发者，公司/组织名，如 fudata、baidu，建议由联盟进行各家名称的标准化

● image-name：需体现组件的功能、分类等重点信息，如 hetero-lr（纵向逻辑回归）、dh-psi（基于 DH 密钥协商的 PSI）、fate-collection（FATE 算法包）

● tag：版本号，遵循 major.minor.patch[-state] 规范（状态是可选的），如 1.0.0、1.0.1-beta。镜像一旦发布则不可更改

b. 镜像标签：通过 LABEL k1=v1 k2=v2 的方式添加键值对，为镜像提供更多的信息，具体含义需在对外文档上说明

c. 目录结构：通过 Dockerfile WORKDIR 指定工作目录，在镜像标签上指定日志目录（见表 6-1）

表 6-1 镜像标签信息

LABEL key	描述
log_dir	日志文件目录，如 /home/admin/app/logs

d. 网络通信：

● TCP/IP socket：最常见方式，通过监听端口来暴露内部服务。需指定端口使容器对外可访问。在镜像标签里提供如表 6-2 所示信息：

表6-2 镜像标签信息

LABEL key	描述
port	端口号，如 80

可能出现端口冲突问题，需要利用基础设施（如 K8S）的能力来解决。

● Unix file（可选）：利用 Unix Domain Socket 技术， 通过监听本地 socket 文件来暴露内部服务;访问外部服务也是通过 socket 文件。该方式可屏蔽网络架构的细节，但需要组件支撑服务实现流量转发。在镜像标签里提供如表 6-3 所示信息：

表6-3 镜像标签信息

LABEL key	描述
uds_dir	socket 文件所在目录，如 /var/uds。 文件命名可参考“组件名称定义规范”中的子接口类型，例如 register.sock

e. 镜像安全性保护手段可参考如下：

● 防篡改：比对镜像的 sha256 摘要值与开发者提供的是否一致

● 漏洞扫描：各家可根据自身基础设施的情况，利用开源或自研工具进行 CVE 漏洞扫描

● 镜像签名：可利用 Notary 服务实现发布方加签与使用方验签

(2) 组件名称定义规范（Component Name）如下：

a. 组件名称 Component Name 用于唯一表示组件的名称字符串，按照<前缀>.<版本>.<类型>.<名称>方式进行命名。

例如：intercom.v1.algorithm.hetero_lr

- 前缀：互通标准前缀
- 版本：接口版本
- 类型：子接口类型
- 名称：接口的名称

b. 组件实例状态

组件实例状态描述如表 7 所示：

表 7 组件实例状态描述

状态标识	状态名称	状态描述
UNLOADED	未加载	当组件镜像未加载到系统时，所处在的状态
LOADED	已加载	当组件镜像完成下载和验证，加载到系统的状态
STOPPED	已停止	当组件镜像停止运行，可以准备进行重新启动或者卸载时所处在的状态
STARTED	已运行	当组件完成了初始化工作，可以对外提供服务时。
FAILED	已故障	当组件镜像运行之后，出现故障后时所处在的状态

组件实例状态转换关系如图 34 所示。

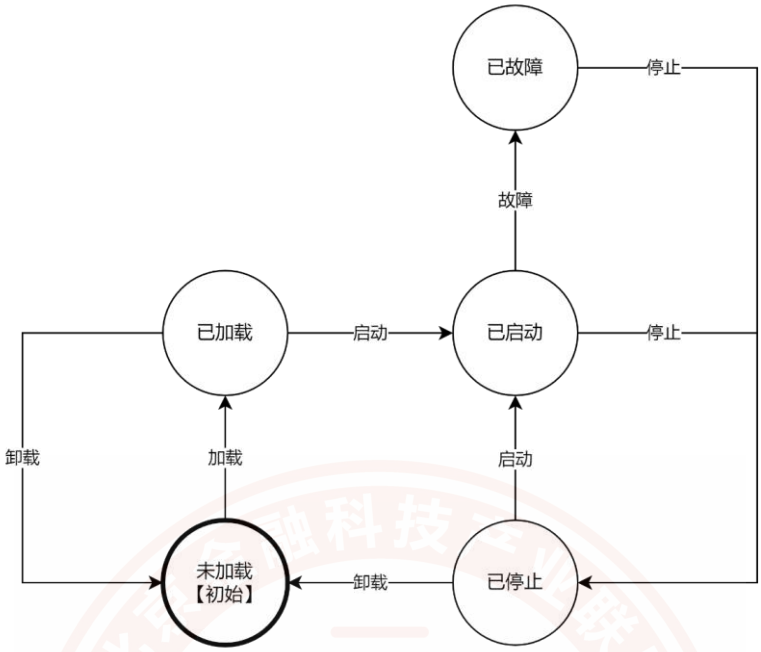

图 34 组件实例状态转换关系图

3. 容器加载流程

各家基于自身的基础设施，通过 container manager 实现容器加载能力。以 K8S 为例，container manager 构建好 Deployment 配置后调用 API server 的 REST 接口(或 kubectl 命令)创建 pod(即 component instance）。

整体交互时序图如图 35 所示。

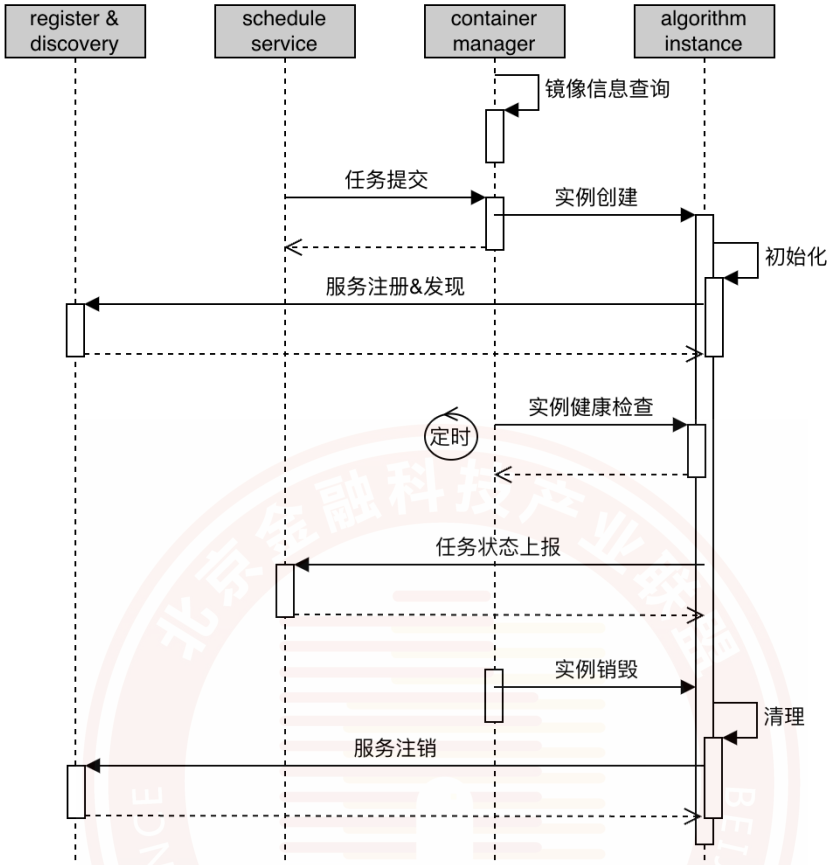

图 35 容器加载流程图

（六）基于最小化约束的算法容器设计

通常，在算法容器中运行的算法或者算子非常依赖系统所提供基础功能支撑接口来实现其内部的运作逻辑。因此，是否存在一套灵活、能满足容器内部逻辑、又能满足兼容性要求的容器支撑接口设计，决定了算法容器设计成败。

事实上，容器支撑接口不是唯一确定或者一成不变的。首先，从接口需求层面来讲，联邦学习等上层应用算法和底层安全算子的实现本身就不

是一成不变并且会随着时间发展而变化，因此，容器需要的接口类型和功能是可能会发生变化；其次，从系统支持层面来讲，各厂商所提供底层支撑系统大多都是异构的，其所能提供的系统功能、接口也各不相同；最后，从互通协议层面来讲，随着算法应用、底层支撑系统及隐私计算技术的不断迭代更新，系统间互通协议也需要不断迭代更新。

综上，算法容器及其配套容器支撑接口的设计需要满足以上技术更新和发展的需求，且应满足以下几点能力要求：

(1) 算法容器需要满足对历史算法容器镜像向下兼容的能力。由于算法镜像的开发可以独立于隐私计算系统的开发，因此，随着隐私计算技术及应用的发展，同一个算法可能会因为不同版本的实现而不能很好兼容，所依赖的底层接口也可能由于新增方法改造实现而不兼容。要想较好地兼容使用历史算法容器镜像，需要新设计的算法容器和接口具备向上支持、向下兼容的能力，这是算法容器设计的一个难点。为了满足以上设计需求，本子课题充分借鉴和吸取了从早期微软 COM 的体系设计方式、到近期行业内广泛流行的微服务设计理念，融入了类似注册中心的机制，尝试通过统一的键值对（Key-Value）形式对不同的接口服务进行统一分类和管理，再通过环境变量的方式给予到镜像内部，从而可以根据版本不同提供不同的接口，最大程度地提高了镜像系统的向下的兼容性。

(2) 算法容器需要尽可能地减少对镜像内部的限制要求，以便更好支持现有算法实现进行平滑迁移。由于算法实现往往基于不同厂商进行设计且不同平台算法实现方式大多存在差异，因此，如何在不对现有算法实现做较大修改的前提下，使算法容器镜像尽可能地兼容现有实现方式，这

是算法容器设计的另一个难点。基于此，课题组启动了算法容器镜像定义工作，通过对目前十多家厂商的实现方式进行充分调研和统计，以及采用整理需求、归结使用方式等手段和逐项排除的方法，确定算法容器镜像约束仅限制在镜像标签和环境变量上；此外，再通过镜像已有标准输出和错误码方式来获取运行状态，以及使用固定地址、去除接口等手段来减少约定项目，从而达到对镜像的低约束的目的。

(3) 算法容器工作流程应该尽可能简单，但同时又要尽可能覆盖绝大多数的应用场景。联邦学习算法原理本身具有多样性，在实现过程中也没有绝对的实现方式，这导致了不同实现算法的工作形态有着较大的出入。从容器生命周期角度来看，算法工作形态可以分成瞬时形态(即用即销毁)和常驻形态。通过对行业内实现场景的调研和统计，课题组发现虽然两种形态都有应用，但瞬时形态占大多数场景；同时出于适配场景和性能考虑，这两种形态虽在现实应用中都有存在，但瞬时形态相对更为简单，常驻形态则由于需要从更多接口和交互流程上进行定义，流程较为复杂。因此，综合应用广泛性和形态复杂性来看，在容器生命周期的选择上，宜以瞬时形态为当前首要设计方式，瞬时形态不仅有助于能够大幅降低的算法容器工作流程的复杂性，同时还能兼顾大多数的应用场景。

根据上述要求，课题组提出了一种简单的可以同时满足绝大多数应用需求的最小化约束算法容器方案：

- 容器内可包含一个或者多个算法组件；
- 容器以瞬时形态（即用即销毁）进行运行；
- 容器计算方式采用离线方式(两个算法间的计算数据流采用落地文

件的形式进行传递）；

● 系统与容器的交互以内存数据为主，落地文件为辅的方式进行。

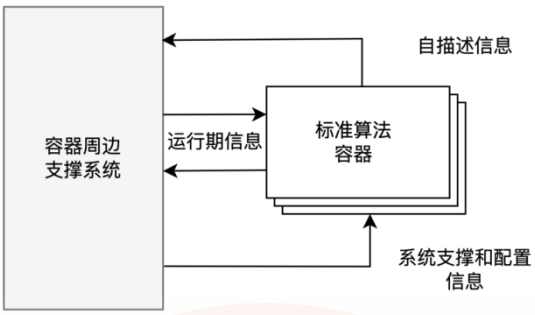

图36 最小化约束算法容器设计图

如图36所示，方案初步定义了算法容器标准与容器周边支撑系统的整体交互流程。其中，算法容器采用瞬时模式进行，且每次运行都会全新创建一个容器：

启动前，系统通过标准容器的自描述信息获取到算法容器内所包含算法组件以及调用方式；

启动时，通过环境变量或者文件的形式，将系统的支撑接口和配置信息传递给算法容器，用于算法容器的初始化和计算过程；

运行时，系统会通过内存的方式，将此次计算过程中所用到的运行信息（输入输出信息）传递给算法容器；

运行结束时，通过获取主进程的退出码作为组件返回值作为成功标志和标准输出作为运行日志。

其中环境交互包括以下内容：

(1) 自描述信息，描述了算法容器中所包含的内容以及使用方式，主要包含：版本、算法名称、参数及输入输出等信息；

(2) 系统支撑与配置信息，描述了周边的支撑系统所能提供的接口及环境配置参数；

(3) 运行期信息，描述了当前运行过程所依赖的动态信息。主要包含：使用的算法名称、参数及绑定的输入输出数据。

综上，最小化约束算法容器方案主要包括以下三个优势：

(1) 兼容性强。通过本方案，可以较好满足新老镜像对于不同版本接口的需求；

(2) 灵活性高。相比现有实现方案需要在镜像定义时固定输入接口的方式，本方案弥补了其不能适应多变接口需求的问题；

(3) 对镜像约定限制小。相比现有实现镜像定义，本方案支持使用最小化的约束来规范镜像的内容,可最大程度地释放了容器内算法实现的空间。

（七）隐私计算安全算子服务设计

目前业内对算法和算子的边界尚未形成明确的定义,算法和算子之间的关系，仍处于一种比较混沌的状态。在隐私计算互联互通的驱动下，业界开始要求隐私计算系统能够以一种松耦合的方式构建,并希望通过标准化、模块化来实现不同功能模块的能力迁移,使各功能模块得以独立发展,更好地突出其自身技术优势。

隐私计算安全算子服务设计拟采用算子服务化的方法将算法和算子进行解耦，更加突出隐私计算安全这一核心要求，以及满足互联互通中技术融合的需要。安全算子（简称算子），是指基于多方安全计算（MPC）

协议、同态加密等构建的最小必要计算算子操作。解耦后的安全算法可不与特定实现方式的算子绑定，灵活利用多种安全算子来构造安全算法。安全算子强调的是基于密码学构造的、采用安全计算协议实现的基础计算工具，例如基于多方的矩阵乘法、欧式距离、大小比较计算等；安全算法强调的是在基础计算之上，面向实际场景问题提出的具体算法，例如联邦逻辑回归、隐匿查询等。

安全算子服务化的方案，由安全算子服务接口和安全算子服务系统建设两个部分组成。安全算子服务化接口，通过定义算子服务各方面功能，约定了算法与算子的协同方式，圈定了算子的功能范围，从而形成了异步化、大数据处理机制等能力；安全算子服务系统建设指南，将在满足算子服务功能性要求的前提下，进一步在安全性验证、算力扩展等方面提供相应技术建议和要求。

1. 安全算子服务接口设计

(1) 算法与算子的交互

在接口设计上，首先需要明确算法与算子的协作方式，并对算法和算子的协作方式进行抽象和定义。以P_0表示隐私计算中的发起方，P_i表示参与方，$\mathcal{F}(x,y)$代表一个场景层面的隐私计算任务，由(P_0, P_i)协调完成，比如$\mathcal{F}(x,y)$可描述一个联邦的纵向逻辑回归或隐匿统计。其中$f_0(x,\dot{y})$代表着发起方P_0的算法；$f_i(\dot{x},y)$代表着参与方P_i的算法；$\dot{x},\dot{y}$表示是外部参与者需要保护的隐私数据，是一个逻辑上可使用但不可见的数据；x,y 表示本方持有的可用可见数据。

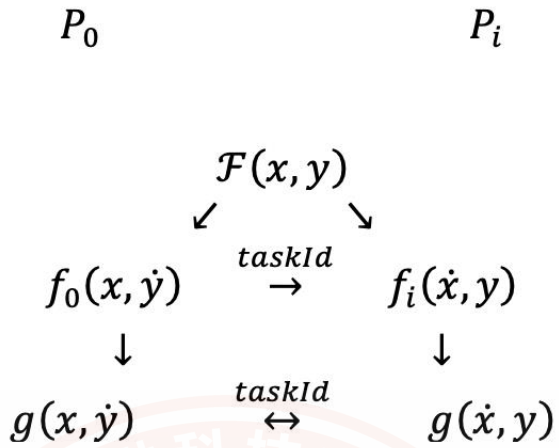

图37 算法算子协作示意图

一个隐私计算任务$\mathcal{F}(x,y)$实际上是由多方参与完成，需要多方进行协同计算（见图37）。用函数$f_0(x,\dot{y})$表示发起方P_0的算法；$f_i(\dot{x},y)$表示参与方P_i的算法；$g(x,\dot{y})$表示发起方P_0所使用的算子；$g(\dot{x},y)$表示参与方P_i所使用的算子。算子负责基于密码学的多方安全计算协议构建安全的计算操作。

P_0每次启动一个算法，将生成一个唯一的$taskId$，并通过算法间调用，将$taskId$分发给P_i。$taskId$的目的是能够使算子、算法之间实现计算的协同和对接，通过$taskId$对接算子、算法的上下文，实现相互识别和通信。

在不同参与方的算子服务接口定义上，所使用的算子服务接口是一样的，区别在于参数上，即 $g(x,\dot{y})$与$g(\dot{x},y)$，算子根据参数的识别构造计

算协同的过程。算法不关注底层算子协同计算的方式，在算子内部也能便于内聚计算逻辑，提供标准化服务。

(2) 算子服务安全性方案

随着MPC技术的快速发展，各隐私计算的科技公司为突出技术优势，提供了功能丰富的算子。在结合实际应用场景调研和分析后发现，业界高频使用的算子不多，主要以矩阵乘法、向量矩阵乘法、大小比较、欧式距离、隐匿求交等基础算子为主，其中虽然也涉及一些动态编译或解释执行的算子，但是这些算子的完备性和安全可验证性还有待进一步研究。

为确保算子使用的完备性和安全可验证性，拟采用满足当前主要隐私计算场景所需的部分高频算子作为研究对象，将矩阵乘法、向量矩阵乘法、大小比较、欧式距离、隐匿求交算子等作为服务化验证算子，并以枚举的方式定义算子功能，以及对这些算子进行规范化定义（表8给出了首批定义的高频算子，后续可继续根据算子性能及场景使用的需要逐步扩大算子列表）。通过此次算子规范化定义算子，不仅有助于更规范地使用算子服务，而且有助于算子服务的安全性开展更具针对性的定义、验证和评估。

表 8 高频使用算子列表

表达式	P_0 输入	P_i 输入	输出格式	描述
vds(x,y)	x 向量	y 向量	明文标量	向量与向量对应点乘 后求和，返回标量。

mvm(x,Y)	x 向量	Y 矩阵	明文矩阵	向量乘以矩阵，返回矩阵。用于逻辑回归中计算梯度。
greater(x,y)	x 向量	y 向量	明文向量	比较 x 是否大于 y
less(x,y)	x 向量	y 向量	明文向量	比较 x 是否小于 y
equal(x,y)	x 向量	y 向量	明文向量	比较 x 是否等于 y
distance(x,y)	x 向量	y 向量	明文标量	计算两个向量的欧式距离
matmul(X,Y)	X 矩阵	Y 矩阵	明文矩阵	计算矩阵 X*Y,其中 shape(X)!=shape(Y)
psi(x,y)	x 向量	y 向量	明文标量	双方隐匿求交

算子服务可提供两种服务形态，分别为无状态的算子服务和有状态的算子服务，二者在处理和安全性验证保障方式上存在一些区别。

当使用无状态算子服务时，对各方而言，算法不接触密文，密文操作也都封装在了算子内部，输入输出都是明文。这种构造使得从输出结果无法反算出他方的隐私数据，使用无状态的算子，能在算子服务级别保证安全。

当使用有状态算子服务时，算法能够接触密文和密钥，并能对密文数据进行一些额外的处理和扩展，比如加入噪声等。有状态算子服务通常会将算子拆分为三个阶段：第一阶段是准备阶段，第二阶段是计算阶

段，第三阶段是结果复原阶段。以下将以秘密分享和同态加密对有状态服务的三阶段进行说明：

- 同态加密

当使用同态加密作为计算协议时，第一阶段完成密钥生成、加密和密文的分发，输出密文和密钥；第二阶段在加密态完成计算，并根据算子表达式定义的功能，完成密文的计算，输出结果仍然是密文；第三阶段采用第一阶段生产的密钥，对第二阶段的数据进行解密，恢复出明文结果。

- 秘密分享

当使用秘密分享作为计算协议时，与同态的步骤类似，第一阶段完成数据的秘密分享；第二阶段，根据算子表达式定义的功能，在分享态完成计算；第三阶段，完成结果的合并。

有状态算子服务相比于无状态的算子服务更加灵活，在实际处理计算中，可以将多个阶段联合执行。例如第一阶段和第二阶段组合，或者第二阶段与第三阶段组合，或者第二阶段用到了多个一阶段的处理结果，这种可以组合的关系，造成了算子协议被算法协议的侵入，使得算子无法单纯从服务层面来保障其安全性，而必须与算法协议一起来联合保障。

(3) 算子服务扩展性方案

在对安全算子的范围约束的同时，本研究仍保留有一定的可扩展性和开放性。例如，接口设计方面，目前是通过枚举约束了表达式的定义范围，

后续可继续通过添加枚举来扩大算子的功能范围；此外，也可通过调整表达式的约束形式，逐步扩大算子功能的范围。

(4) 算子服务大数据传输方案

算子服务需要支持大数据的处理。服务化以后，为更好支持大数据处理，将对数据的传输与算子服务接口进行松耦合设计。除了提供直接通过接口传输数据的方式之外，还可借助存储引擎的方式进行大数据的传输，其中可由算法来指定数据的输入和输出方式。

算子服务需要处理的数据，均以数据块的方式进行传输；不同的编程语言，在数据类型上存在区别，算子服务化需要解决跨编程语言的数据传输问题。本研究将通过对数据块形状、数据字节序、数据类型进行规范化定义的方式，来实现跨语言的数据传输。

(5) 算子服务的异步化

算子执行的过程一般耗时比较长，因此在服务化后，可利用算法与算子的解耦化及计算资源的分离状态，为算法提供一种不需要阻塞等待的方法。例如，可在接口设计上提供异步化机制，使算法能够通过接口参数指定异步或者同步方式来获取算子计算结果。

(6) 算子服务间通信方案

算子需要依托通信传输模块来完成跨节点的算子服务间通信。算子服务在启动的过程中，需要将算子服务的本地容器地址向传输模块进行注册；在通信过程中，建立信道时也要及时将本地容器地址与信道关联，并在建立起跨节点的算法服务间会话后，复用节点间的信道与端口。

2. 安全算子服务系统建设指南

(1) 服务算子容器安全性

在互联互通框架设计中，安全算子服务层定义了需采用 Docker 容器的方式对安全算子进行封装和发布。容器化的方式一方面更利于异构安全算子的迁移、部署、扩展；另一方面由于容器化封装的方式，使得在实际应用中算子容器的安全性尤为重要。因此在安全算子服务层的定义中，对隐私计算平台互联互通做出以下几点安全性要求：

a. 在安全算子服务迁移时，隐私计算平台需能够核对镜像的签名和开发者发布的相关信息一致性，防止算子容器被篡改；

b. 隐私计算平台需能支持自开发或借助扫描工具对安全算子服务进行漏洞扫描，以保证服务的安全。

(2) 服务算子算力扩展

安全算子服务主要负责隐私计算算法中最小计算算子的计算任务，使得算子服务的计算速度对整个隐私计算任务的效率影响很大。在实际应用场景中，单个物理节点的计算性能上限无法满足大数据的计算需求，所以在安全算子服务的框架设计中也考虑了算子算力扩展的需求。具体的框架设计如图 38 所示。

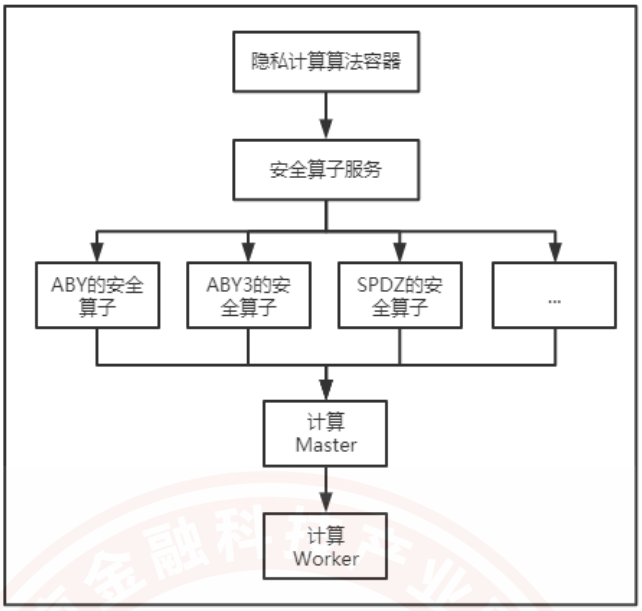

图 38 安全算子服务分布式计算框架图

该框架设计主要参考于应用较多的分布式计算框架，如图中所示，Worker 负责计算任务的执行并且支持多个 Worker 节点的扩展，Master 主要负责协调和管理多个 Worker 间的任务运行。

(3) 服务稳定高可用

在互联互通框架设计中，由于安全算子服务层承接了若干个算法组件层的大量算子计算任务，使得安全算子服务的稳定性对隐私计算任务能否正常运行起到了关键的作用。针对实际应用场景，我们设计了一套服务高可用的部署方案供参考，具体参考如图 39 所示。

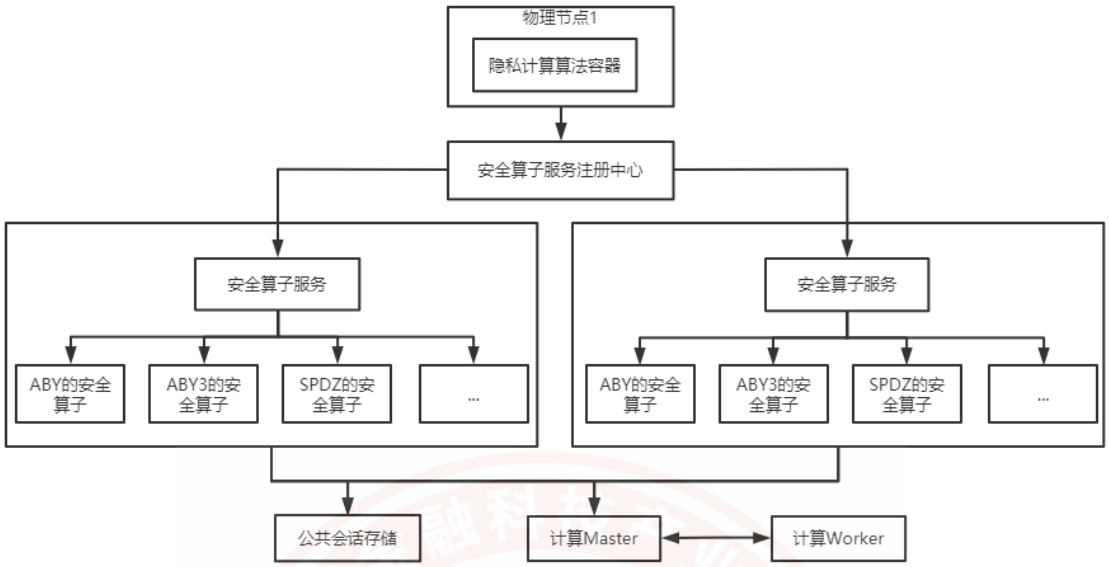

图 39 安全算子服务高可用

在实际使用场景中可以部署多个安全算子服务，并由统一的服务注册中心来管理算子服务。当接收到计算任务时，注册中心负责分配计算任务给某一个安全算子服务；即使某一个算子服务宕机时，注册中心也会将计算任务分配给可用的算子服务，不会影响到整个算法任务的运行。对于安全算子计算过程中的状态、中间结果等，可采用统一的公共会话存储的方式来解决。

（八）传输层同步异步兼容设计

在传输层的互联互通实践中，传输协议与报文结构是核心内容，这之中又可具体细分为同步传输和异步传输两种模式，我们意在设计一种同步异步兼容的传输层规范。为了实现这一目标，我们需要对“组网结构”“安全要求”“标准传输协议”几部分内容进行标准化定义。

1. 组网结构

不同隐私计算平台之间的连接，主要可分为直连模式和路由转接模式两种基础拓扑结构，基于两种基础拓扑结构的组合、扩展实现不同的互联互通组网。

(1) 直连模式

隐私计算平台直接建立连接，进行通信和交互。其拓扑结构如图 40 所示。

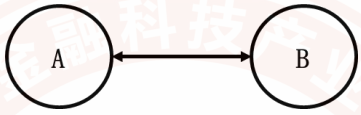

图 40 直连模式拓扑图

采用直连模式互联互通，平台间按照相同的技术方案，各自实现相同的接口或软件组件，建立点对点的相互连接。

(2) 路由转接模式

隐私计算平台间不直接建立连接，而是通过独立的第三方路由进行交互计算、格式转换。其拓扑结构如图 41 所示。

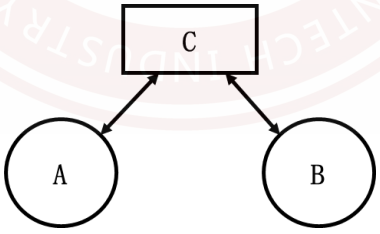

图 41 路由转接模式拓扑图

采用路由转接模式互联互通，可建立独立的软硬件隐私路由，各平台通过与隐私路由相连，实现在同一个隐私路由上的平台间组网。当新的隐私计算平台或节点要加入网络时，需要先与将发生互联互通的平台建立身份互认关系，再与互认的平台进行多方联合计算。另外，可通过

隐私路由间的链接，实现平台间跨路由的互联互通。从安全性上看，由于第三方路由参与了不同密文计算协议的数据格式转换，其安全性与相应的计算转换协议和安全假设有关。在这种模式下，传输层仅保证传输信道的安全性，第三方路由的安全性由相应的计算协议来确定。

2. 安全要求

跨平台间的互联互通需要建立在安全的基础上来进行,对于安全的要求可具体划分为传输安全和业务安全。

(1) 传输安全

a. 数据机密性的技术保障

数据机密性主要通过加密技术进行实现,在数据通过网络信道传输前使用相应的密钥及加密算法对原始数据进行加密,接收方在接收到网络信道中传输的密文后再使用相应的密钥及解密算法进行解密。

b. 数据完整性的技术保障

数据完整性主要通过哈希（Hash）或消息验证码（MAC）技术进行实现，在数据通过网络信道传输前，使用哈希或消息验证码技术对数据计算相应散列值或校验码,将数据与对应的散列值或消息验证码一起通过网络发送给接收方，接受方在接收到网络信道中传输的消息后，提取消息中的数据后采用与发送方相同的方式计算散列值或消息校验码,并与消息中提取的散列值或消息校验码进行比对。

c. 不可抵赖性的技术保障

不可抵赖性包括发送方不可抵赖性及接收方不可抵赖性,具体涉及身份不可抵赖及消息不可抵赖两方面。发送方不可抵赖性主要通过身份认证

及数字签名等技术进行实现，接收方不可抵赖性主要基于可信第三方来实现。

(2) 业务安全

a. 跨平台合作方识别

跨平台合作方识别可通过预制 token 等方式实现。隐私计算平台间所有通信请求均需携带相应 token 信息，并对请求内容进行签名校验，确保请求发送自真实的合作方。

b. 权限控制

针对隐私计算平台互联互通场景，权限控制主要包括数据权限控制及任务权限控制两大类。

数据权限控制主要包括数据查看权限控制、数据使用权限控制。数据查看权限控制指对隐私计算平台数据市场中可对合作方展示的数据源进行控制；数据使用权限控制指对互联互通的隐私计算具体任务可使用的数据源进行控制。

任务权限控制主要对隐私计算互联互通合作方可具体发起的任务类型进行控制，比如可以对已授权的数据进行安全求交、联邦建模、匿踪查询等计算任务。

3. 标准传输协议

跨隐私计算平台间的互联互通，需要依赖于一套标准的传输协议和报文结构。基于规范化的报文头结构，可以定义标准的同/异步传输协议。

(1) 标准报文头

报文头是开始并唯一标识报文的服务段。在该结构体中，我们需要对

协议版本号、厂商编码、认证令牌、发送和接受方的节点 ID 等内容进行规范化定义，具体内容见表 9。

表 9 报文头字段编码

头字段编码	头字段名称	头字段说明
version	版本	协议版本号
tech-provider-code	厂商编码	互联互通统一厂商编码
trace-id	traceId	用于追踪
token	令牌	认证令牌
source-node-id	发送方节点 ID	发送端的节点 ID
target-node-id	接收端节点 ID	接收端的节点 ID
source-inst-id	发送端机构 ID	发送端业务实体 ID
target-inst-id	接收端机构 ID	接收端业务实体 ID
task-session-id	会话 ID	用于需要建立有状态会话的通信时使用，一般情况下，算法和算子的通信，都需要建立会话，通过任务会话 ID 来建立计算执行容器的路由，除此之外，对 token 鉴权的时候，需要联合 task-session-id，对 token 的有效性进行验证

(2) 标准报文传输格式

标准的传输报文和具体的传输协议无关，包含了报文头和报文体，传输错误码和错误信息，报文编码需加入x-前缀。除此之外，还需要预留可扩展的元信息（见图42），例如，可以通过元信息，扩展异步通信所需要的message-topic和message-code。

```
syntax = "proto3";
package org.ppc.ptp;

// 标准协议头
enum Header {
  version = 0;              // 协议版本
  tech-provider-code = 1;  // Server端厂商编码
  trace-id = 3;             // 链路追踪ID
  token = 4;                // 认证令牌
  source-node-id = 5;       // 发送端节点编号
  target-node-id = 6;       // 接收端节点编号
  source-inst-id = 7;       // 发送端机构编号
  target-inst-id = 8;       // 接收端机构编号
 task-session-id = 9;         // 通信会话号，全网唯一
}

// 可扩展报文头字段名枚举
enum Metadata {
  message-topic = 0;       // 消息话题，异步场景
  message-code = 1;        // 消息编码，异步场景
}

// 通信传输层输入报文编码
message Inbound {
  map<string, string>  metadata = 1;  // 可扩展报文头，可选，预留扩展，序列化协议由通信层统一实现
  bytes payload = 2;                  // 报文，上层通信内容承载，序列化协议由上层基于SPI可插拔
}

// 通信传输层输出报文编码
message Outbound {
  map<string, string>  metadata = 1;  // 可扩展报文头，序列化协议由通信层统一实现
  bytes payload = 2;                  // 报文，上层通信内容承载，序列化协议由上层处理
  string code = 3;                    // 状态码
  string message = 4;                 // 状态说明
}
```

图 42 同步传输协议结构图

（九）异构平台开放算法协议设计

1. 握手原则

开放协议应当分为握手流程和算法流程两个步骤，握手流程应当具备如图 43 所示的能力：

(1) 通用算法类的参数算法对齐。例如隐私集合安全求交（PSI）有多种实现路径，但是不同的实现路径都应允许指定结果获取方是谁等和 PSI 实现方法无关的对齐。

(2) 安全原语相关的参数对齐。例如椭圆曲线迪菲-赫尔曼密钥交换（ECDH） 来做 PSI，需要对齐 ECC 曲线类型以及哈希函数。

(3) 为减少所使用的 RTT，上述握手在一次握手包里完成交互。

图 43 握手流程设计图

2. 同构算法的分类和设计

(1) 专有流程算法设计

专有流程的算法在协议设计时应当充分考虑不同需求的兼容性。一般来说，每个算法特点差异很大，而专有算法流程的设计应当满足足够强的拓展性和包容性。以 ECDH-PSI 为例，其中椭圆曲线和哈希函数的参数上应当优先考虑以国密为标准，支持 SM2 和 SM3 组合的 PSI 算法。ECDH-PSI 的算法流程如图 44 所示。

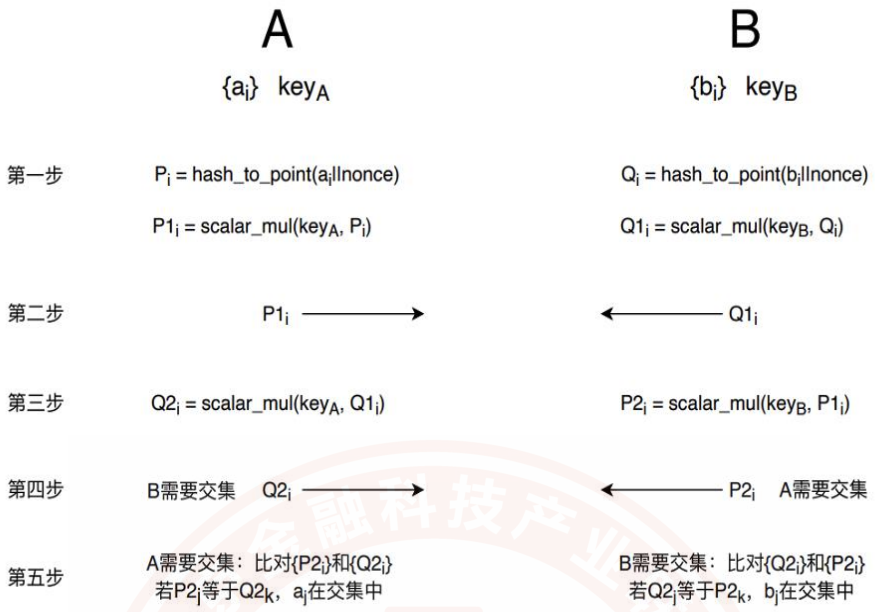

图 44 ECDH-PSI 算法交互流程图

(2) 基于秘密分享的通用多方安全计算算法互通框架

由于秘密分享是一种通用的隐私计算能力，所以，开展基于秘密分享的通用多方安全计算算法互通框架设计，有助于实现更彻底的算法解耦。

结合秘密分享，整体设计框架参考如图 45 所示。

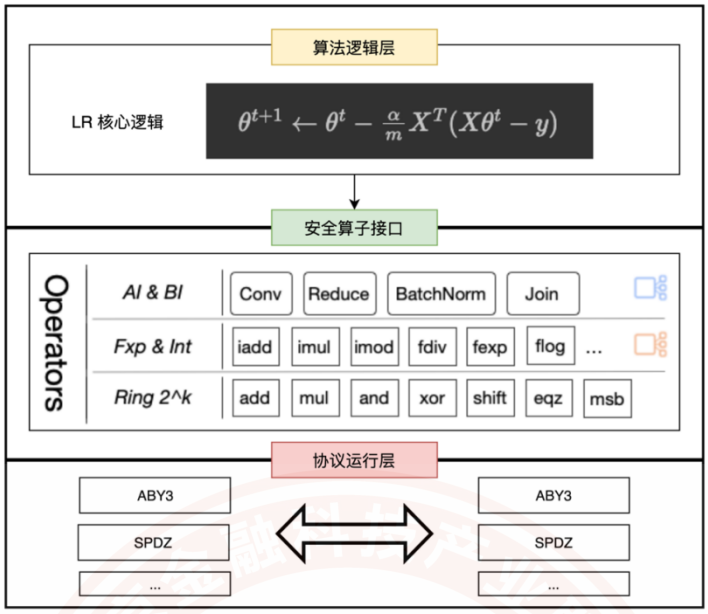

图 45 整体设计框架图

● **算法逻辑层。**算法逻辑层用于描述数学上算法应当如何被拆解为更细粒度的数学运算组合，且不应当与安全实现机理细节相耦合，例如不会特别指定采用 混淆电路（GC） / 同态加密（HE） / ABY3 或者其他具体协议机制。

● **安全算子接口层。**安全算子接口层应当具备扩展性，且有能力支持日渐复杂的算法表达式，尤其是对复合表达式的支持（复合表达式的支持可以减少算法逻辑层和安全算子层的频繁调用）。同时，更大粒度的表达式也给予了底层协议更多融合、折叠的优化空间。

● **安全原语协议互通层。**经过上述两层的解耦合抽象，安全原语协议互通层可以更好复用专有算法协议设计流程来实现互通。期间，握手和协议交互将按照上述框架原则设计完成。

3. 基于隐私路由的异构算法互联互通

基于隐私路由的异构算法互联互通研究，借鉴成熟互联网协议中关于自治系统（AS）的设计理念，首先将每一个隐私计算平台视为一个 AS，其次通过路由进行连接，进而实现跨平台数据面互通。参考如图 46 所示。

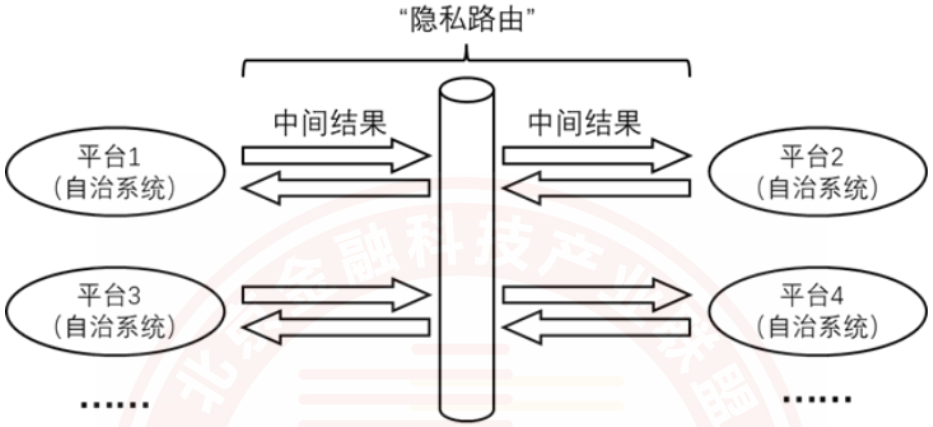

图 46 异构算法互联互通示意图

在该实现方案下，每个隐私平台无需关注其他平台的内部实现细节（如平台架构和技术类型），而仅需关注流通的中间结果即可。技术实现上，各平台可以通过特定的接口组件（路由客户端 Router Client，RC）来接入隐私路由，并重点关注于实现自身中间结果与某一种特定数据格式间的转化即可，无需再与所有其他平台进行互通关系改造。

该模式下，所指定的特定数据格式是关键。由于这种特定的数据格式仅在隐私路由器上出现，且仅针对桥接各种不同格式的中间结果，所以该方案对隐私计算平台的改造影响和侵入性很小，平台本身也不需要面向不同的平台进行多次改造和适配。

这种统一的数据格式应该尽量简单。一般而言，互联互通传递的中间结果代表着数据的使用价值，具有“可用不可见”的属性，因此，一种直

接简单的想法是，采用密码学算法技术来保护该中间结果的安全性，并将所有密码运算技术产生的中间结果统一转化为秘密分享的两个随机数分片。下面以“基于 SS3 秘密分享（三分片的秘密分享）的隐私计算平台和基于 HE 的隐私计算平台间互联互通”为例来具体阐述该技术方案的实现过程，如图 47 所示。方案主要涉及两个功能节点：一是部署在隐私计算平台侧的接入客户端 RC（Router Client）；二是具备数据转换功能的“隐私路由”服务器 RS（Router Server）。

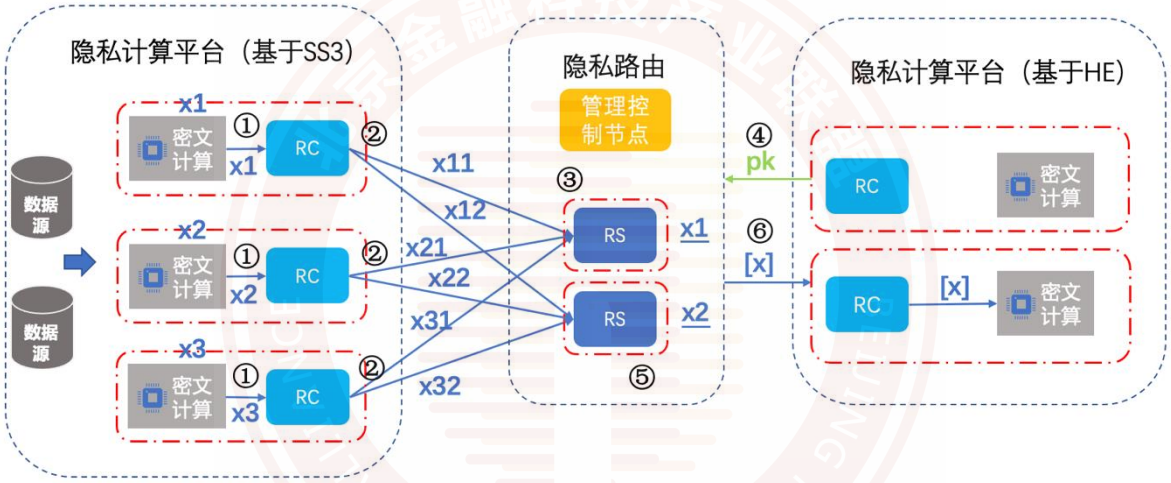

图 47 SS3 计算的中间结果转化为 HE 的密文示意图

其中，隐私路由承载了统一的数据格式：中间结果的两个分片。具体工作流程为：

(1) 左侧平台通过 SS3 秘密分享技术进行隐私计算，获得某个中间结果，该结果以 x1、x2、x3 三个分片的形式分布在不同的管理域上；

(2) 左侧三个接入节点分别对 x1、x2、x3 进行分片，每个拆分成两份后分别发送给隐私路由的两个转换节点；

(3) 两个转换节点分别对各自获得的分片（三份）进行合成，进而

得到中间结果的新分片：x1、x2；

(4) 右侧平台将公钥分发给隐私路由；

(5) 隐私路由的两个转换节点分别通过公钥对 x1、x2 进行加密，并在其中一个节点上对密文进行合成得到[x1]+[x2]，即[x1+x2]，也就是中间结果的密文形式[x]；

(6) 隐私路由将[x]发送给右侧平台的接入节点。左侧平台得到[x]后可进行本平台内的隐私计算。

(7) 需要注意的是，隐私路由两侧的设计实现是独立的：我们即使把左边的 SS3 替换为 SS4（基于四分片的秘密分享）或者其他 SS 协议，抑或是其他隐私计算的密码技术，都不影响右边平台从隐私路由取数的协议接口（即步骤(4)-(6)）；反之，左边平台发送中间结果的协议接口（上面步骤 2）也不受右边平台取数方式的影响。

（十）TEE 统一远程证明流程设计

TEE 互联互通关键技术点在于统一远程证明流程的设计，设计一套统一的 API 和报告格式封装，通过 TEE 平台和应用属性的统一抽象解决异构 TEE 远程证明的兼容性和互联互通问题。

统一远程证明流程设计主要涵盖以下关键内容的设计：

● UAR：统一的远程证明报告格式（Unified Remote Attestation Report）是对各种 TEE 方案中不同 RA Report 格式的统一抽象封装。

● UAP：统一的远程证明规则（Unified Remote Attestation Policy）主要包含用户如何对 TEE 平台和应用做校验的规则集合，这些规则项都是

所有 TEE 平台和应用属性的统一抽象。

● UAI：统一的远程证明报告接口（Unified Remote Attestation Interfaces）主要提供和 UAR 相关的两个接口： 报告生成接口和报告校验接口。

● UAS：中心化的远程证明代理验证服务（Unified Remote Attestation Service）：收敛所有远程验证差异化逻辑，使得其他 TEE 端借助证明服务完成与对端的远程验证,而在验证过程中 TEE 端无需额外兼容异构 TEE 的认证逻辑。

1. 统一远程证明报告格式（UAR）设计

统一远程证明报告格式（UAR）如下：统一的远程证明报告格式是为了屏蔽不同 TEE 方案中报告格式的差异,方便统一远程证明接口兼容不同的 TEE 平台。考虑到不同 TEE 系统中不同编程语言的兼容性，使用 JSON 格式作为统一远程证明报告格式，方便各种应用之间数据交互。具体实现也可以考虑利用 protobuf 定义数据格式，可以方便和 JSON 格式之间互转。

对于具体的 JSON 数据项键名，采用统一命名规则，数据值格式前缀_键名称。数据值格式前缀含义如表 10 所示。

表10 报文头字段编码

前缀	对应数据值格式
str	Normal string
hex	HEX string, using capital letter A-F
b64	Base64 encoded

pem	PEM string for key or certificate
json	JSON string
int	int
int64	int64
uint	uint
uint64	uint64
bool	"false" or "true" as string format
bin	binary data

统一远程证明报告格式（UAR）如图 48 所示。

```
//UnifiedAttestationReport
{
    "str_report_type":"...",
    "str_tee_platform":"...",
    "json_report":"...",
    // JSON serialized UnifiedAttestationNestedReports
    "json_nested_reports": "...",
}

// UnifiedAttestationNestedReports
{
    // The JSON serialized string of UnifiedAttestationNestedResults
    "json_nested_results" : "...",
    // The signature of json_nested_results
    "b64_nested_signature": "..."
}

// UnifiedAttestationNestedResults
{
    "nested_report_results": [
        // All UnifiedAttestationAttributes, 经过本地校验之后的次级TEE信息汇总
        { ... },
        { ... }
    ]
}
```

图 48 统一远程证明报告格式

此外，为了能够正确解释市场上不同架构体系 TEE 平台的报告，子报告中详细列举了常用 Intel SGX1、Intel SGX2、蚂蚁 HyperEnclave、华为鲲鹏以及海光 CSV TEE 平台远程证明报告具体数据结构。

2. 统一远程证明校验（UAP）规则抽象

校验规则和统一抽象的可信应用程序可以被远程证明校验，实现相关属性集合对应，校验方可以根据平台选择对应的校验属性。

属性集合是所有TEE平台定义的可信应用程序属性的合集，相应属性集合如图49所示。

```
// UnifiedAttestationAttributes
{
    "str_tee_platform":"...",          // TEE平台标识字符串，对应UAR中str_platform字段
    "hex_platform_hw_version":"...",   // TEE平台硬件关联版本号
    "hex_platform_sw_version":"...",   // TEE平台软件关联版本号
    "hex_secure_flags":"...",          // TEE平台或者可信实例的安全属性标志位
    "hex_platform_measurement": "...", // 硬件平台相关组件度量值
    "hex_boot_measurement": "...",     // 启动阶段相关组件度量值
    "hex_ta_measurement":"...",        // TA的静态度量值
    "hex_ta_dyn_measurement":"...",    // TA的动态度量值
    "hex_signer":"...",                // TA的签名信息
    "hex_prod_id":"...",               // TA的产品编号
    "str_min_isvsvn":"...",            // TA的软件版本
    "bool_debug_disabled":"...",       // TA是否关闭了debug模式
    "hex_user_data":"...",             // 用户自定义数据
    "hex_hash_or_pem_pubkey": "...",   // 报告中绑定的用户公钥或者其HASH，对应公钥一般用于业务数据加密
    "hex_nonce":"...",                 // user_data之外，独立的保证freshness的随机值
    "hex_spid":"..."                   // 第三方ServiceProvider请求标识，比如SGX1中使用
}
```

图 49 统一远程证明校验属性集合

根据Unified Remote Attestation Attributes我们可以指定对应的校验规则(UAP)，对应格式如图50所示。

```
// UnifiedAttestationPolicy, both main or nested submodule attester support multi-instances.
{
    // 假定一个公钥和一份报告绑定，可以再这里统一指定公钥。
    // 或者在UnifiedAttestationAttributes里面精确指定每个公钥HASH值
    // 如果指定，这里的公钥和UnifiedAttestationAttributes里面公钥hash都会被用于校验报告中实际公钥hash
    // 如果报告中存在submodule，那么一定需要在这里指定公钥用于校验
    "pem_public_Key": "...",
    // 主TA的所有可能校验规则，比如多个版本，多个signer都认可
    "main_attributes": [
        {...}, // UnifiedAttestationAttributes
        {...}
    ],
    // multi nested submodules, and each submodule
    // support multi UnifiedAttestationAttributes
    "nested_policies": [
        {
            "sub_attributes": [
                {...}, // UnifiedAttestationAttributes
                {...}
            ]
        },
        { ... } // next submodule
    ]
}
```

图 50 统一远程证明校验格式

3. 统一的远程证明报告接口（UAI）抽象

为了兼容更多其他编程语言，统一远程证明报告接口（UAI）涉及的API封装（见图51和图52）都使用C-ABI语法定义。

- 报告生成接口

```
/// @brief C API for unified attestation report generation
/// @param tee_identity: The identity of TEE or TA instance
/// @param report_type: Type of report, "BackgroundCheck"|"Passport"|"Uas"
/// @param report_identiy: The identity string for report instance
///                        which is cached inside TEE. It's optional
///                        and usually used in Asynchronous processes.
/// @param user_data_buf: The user data will be included by the report.
/// @param user_data_len: The user data buffer length.
/// @param report_josn_buf: The output serialized JSON string of AttestationReport.
/// @param report_josn_len: The maximal JSON report buffer size as input,
///                         and the real JSON report string size as output.
///
/// @return 0 means success or other error code
///
int UnifiedAttestationGenerateReport(const char* tee_identity,
                                     const char* report_type,
                                     const char* report_identity,
                                     const char* user_data_buf,
                                     const unsigned int user_data_len,
                                     char* report_json_buf,
                                     unsigned int* report_json_len);
```

图 51 报告生成接口

- 报告校验接口

```
/// @brief C API for unified attestation report verification
///
/// @param report_json_str: The serialized JSON string of UnifiedAttestationReport.
/// @param report_json_len: The length of serialized JSON string of UnifiedAttestationReport.
/// @param policy_json_str: The serialized JSON string for UnifiedAttestationPolicy.
/// @param policy_json_len: The length of serialized JSON string for UnifiedAttestationPolicy.
///
/// @return 0 means success or other error code
///
int UnifiedAttestationVerifyReport(const char* report_json_str,
                                   const unsigned int report_json_len,
                                   const char* policy_json_str,
                                   const unsigned int policy_json_len);
```

图 52 报告校验接口

4. 中心化的远程证明代理验证服务设计（UAS）

统一证明服务本身也需要基于TEE实现，以此保证其自身验证逻辑的正确性。而各个TEE端在交互通信前，需要向对端发起远程验证请求，以获取统一证明报告发送给统一证明服务。远程验证由统一证明服务代理完成，验证结果由统一证明服务进行背书确认，并且也封装为UAR格式，方便TEE端做其他额外校验。

通过引入基于中心化的证明服务，可以将N（N个节点）*N（N套远程验证机制）的工程复杂度压缩到N（N个节点）*1（1套远程验证机制）。

(1) 统一验证服务的TEE节点注册流程

统一证明服务的TEE节点注册流程是为了实现TEE节点向UAS注册，获取访问凭证。两种参考流程如下：

参考流程一：

- TEE节点生成表征自己身份的一组公私钥对：ak.pub/ak.priv
- TEE节点将自己必要的元信息与ak.pub发送给UAS

● UAS返回自己的根证书以及签发的ak.crt给TEE节点 （为了保证ak.crt安全，UAS作为第二级CA，由第三方CA对UAS颁发中间证书）

参考流程二：

● TEE节点提前获取用于验证UAS签名的公钥，内置于可信代码中。（UAS签名公私钥对可以通过安全KMS保管）

● TEE节点开发者或者挑战者都可以通过UAS管理界面注册元信息，获取访问凭证AccessKey（类似Intel IAS做法）

● 后续TEE节点或者挑战者都可以通过AccessKey+UAR请求UAS提供的代理验证服务

(2) 统一证明服务的TEE节点远程认证流程

引入中心化的统一证明服务的TEE节点远程认证流程：

a)护照模式报告转换为UAS格式报告（见图53）：

图 53 护照模式报告转换为 UAS 格式报告

b) 背调模式报告转换为UAS格式报告（见图54）：

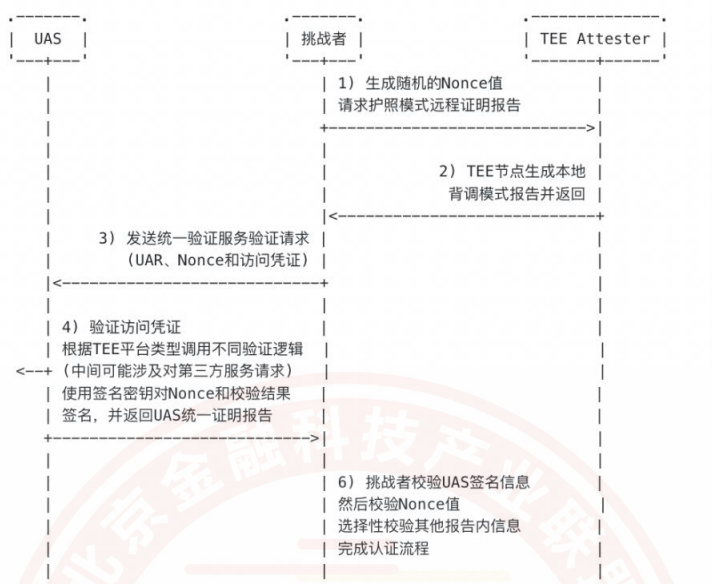

图 54 背调模式报告转换为 UAS 格式报告

(3) 统一证明服务的报告格式

统一验证服务的返回报告格式也是Unified Attestation Report格式。其中"str_platform"="Uas", "json_report"为图55所示格式的JSON序列化字符串。

```
message UasAttestionResult {
    optional int64 int64_result_code = 1;
    optional string str_tee_platform = 2;
    optional string hex_nonce = 3;
    optional string b64_quote = 4;
}

message UasReport {
    // The JSON serialized UasAttestionResult
    optional string str_uas_result = 1;
      // the UAS signature of str_uas_result
    optional string b64_signature = 2;
}
```

图55 统一证明服务的报告格式

五、隐私计算互联互通生态展望

隐私计算互联互通生态研究建立在本子课题所提出的隐私计算互联互通的技术框架之上，探索研究当隐私计算互联互通标准形成后，所可能产生的生态模式，从而更好地保障整个互联互通体系的运转。

（一）生态建设目标

通过研究互联互通生态，助力面向金融行业发展需要的隐私计算互联互通框架的落地与推广，促进隐私计算互联互通形成良性生态，推动隐私计算在标准化体系下的进一步发展，为数据要素市场发展贡献行业力量。

1. 形成数据流通网络

构建基于统一标准和接口的隐私计算互联互通生态，有利于打破数据流通技术壁垒，创新跨主体数据安全共享模式，形成数据流通网络，为实现跨机构、跨地域、跨行业数据资源有序共享和综合应用提供有效支撑，使更多资源投入聚焦于金融行业务创新，充分挖掘数据价值，赋能实体经济发展，发挥数据和技术双轮驱动作用。

2. 构建互通新业态

通过隐私计算互联互通生态研究与建设，在保护知识产权的同时促进技术共建，推动金融与其他领域系统互联和信息互通，不断拓展金融行业数据要素广度和深度，实现数据综合应用与多向赋能，助力服务实体经济、创新驱动发展等工作更好落地。通过隐私计算贯通各领域数据资源，形成一个完整、安全的数据共享体系，构建一个兼容开放、互利共赢的互联互通新业态，从而将数据要素的市场化红利惠及各个行业。同时，作为一种

创新的协作方式，隐私计算互联互通有助于打通各领域数据融合壁垒，形成各行业汇聚合力，提升数据流转效率，为数字文明时代技术创新降本增效。

3. 延伸生态逐步实现生态循环

通过隐私计算互联互通生态延展加快推动生态参与方之间的技术兼容性和项目的协同性，一方面通过协同整合各方力量，节省技术资源投入，聚合各自生态资源，扩大各自项目影响；另一方面，加快基础设施的建设，打造互联互通相关市场，提供增值服务，助力数据要素安全有效流通，从而实现数据流通业务场景的商业闭环。

（二）生态蓝图

隐私计算互联互通生态将在各参与方商定的治理机制和规范的引领下共同演进。通过各参与方间的有效互补与合作，逐步完善治理机制与准则，加快实现价值共创。

1. 生态参与方的角色定位

互联互通生态的基本参与方主要包括数据提供方、技术输出方、业务需求方、监管审计方和其他相关方（见图 56），各参与方在某些业务场下又可能同时身兼不同的角色。

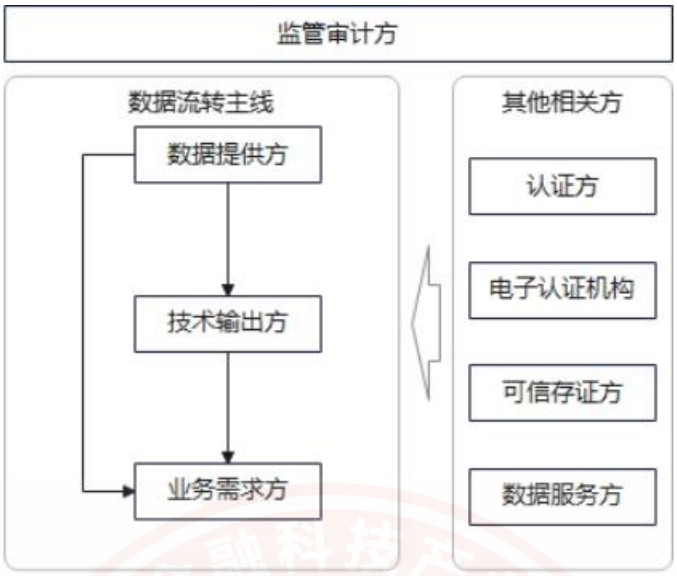

图 56 隐私计算互联互通生态参与角色图

(1) 数据提供方

数据提供方是指提供隐私计算所需私有数据的参与方。典型的数据提供方有政务大数据局、数据交易所、通信运营商、征信公司、互联网企业、金融机构、医疗机构等。

在产业生态中，数据提供方根据自身拥有数据的情况，在确保数据安全的前提下对外共享数据要素，并获得预期收益。在推动互联互通时，数据提供方可重点从数据管理层面主导互联互通探索，明确包括数据目录、数据描述、数据样例、数据授权等管理流程和标准接口规范。

(2) 技术输出方

技术输出方是提供隐私计算算法模块，建设互联互通业务流程，主要贡献技术研发及应用能力的参与方。技术输出方可以是专业的隐私计算技术服务商，也可以是具备一定技术禀赋的数据应用领域的其他主体，如数据交易所、大型互联网企业、大型商业银行等。隐私计算互联互通目标下，

技术输出方承担着相较独立隐私计算厂商更为复杂的角色定位。互联互通隐含的交互标准化、算法模块化、部署复杂化、数据产品化等新特征，需要在技术输出方的综合技术服务中予以落实。

(3) 业务需求方

产业生态中的业务需求方是指通过发起隐私计算任务获取多方数据计算结果的参与方，典型的业务需求方有金融机构、医药厂商等。

业务需求方从本领域的业务需求出发，具备大量的用数场景，且对于业务应用流程有较为明确的功能要求。在推动互联互通时，业务需求方需明确对隐匿查询、隐私求交、联合统计、联合建模等服务流程标准定义，主导项目管理、任务定义、任务执行、任务监控等要求，探索异构平台同类应用服务的流程标准化。同时业务需求方从具体需求场景出发，以利用数据创造业务价值为目标,具有较强驱动力,更能推动互联互通各方合作。

(4) 监管审计方

监管审计方是指产业生态中负责对跨机构隐私计算应用中的业务合规、数据安全、风险事件等进行监督、管理、仲裁、审计等职责的有权机构，确保跨机构隐私计算互联互通生态发展合法有序。监管审计方可由相关行业主管部门或监管部门担任。

监管审计方的职责分为两方面：一是对隐私计算所涉金融业务合规性进行监督管理，基于《个人信息保护法》[9]、《征信业务管理办法》[10]等法律法规，处罚不合规隐私计算业务；二是对隐私计算互联互通中出现的信息窃取、数据泄露、网络攻击等风险事件进行调查、审计和仲裁，确保隐私计算互联互通行业生态行稳致远。

(5) 其他相关方

1）认证方

认证方是指产业生态中负责对隐私计算产品进行技术评估与认证，确保隐私计算产品满足生态对互联互通能力要求的机构。认证方应由检测机构、认证机构等第三方中立组织担任。

2）电子认证机构

产业生态中的电子认证机构是指作为权威的证书签发，且具有合法性、权威性和公正性的第三方信任机构，包括但不限于数字证书的审批和安全管理职责。

3）可信存证方

产业生态中的可信存证方是指将生态建设业务合作的过程中对于各参与角色（不局限于技术输出方、数据提供方等）的隐私相关场景合作，对于平台底座、节点资源、存储资源、通信网关、算法协议、接口和任务等维度，通过特定的技术进行电子存证的机构。该机构可以是技术输出方、数据提供方、监管审计方中的一方或者多方。

4）数据服务方

产业生态中的数据服务方包括不限于作业调度方、数据商和第三方服务机构等能够促进和满足数据要素合规高效、安全有序流通和交易需要的机构。作业调度方是根据互联互通隐私计算流程，通过调度各参与方完成隐私计算作业并同步计算过程中各方任务状态的参与方；作业调度方不提供隐私计算流程中需要的算法、算力、数据资源。数据商为数据交易双方或多方提供数据产品开发、发布、承销等服务，提高数据交易质量和效率。

第三方服务机构是开展数据集成、数据经纪、合规认证等第三方专业服务机构，可提升全流程服务能力。数据服务方也可能承担技术输出方等角色。

2. 生态发展蓝图

互联互通整个生态的构建和发展应建立在比较健全的标准和治理体系之上（见图 57），通过在产品、检测和应用阶段不同生态参与方之间的配合，完善和带动互联互通的生态治理体系，促进互联互通持续健康发展。

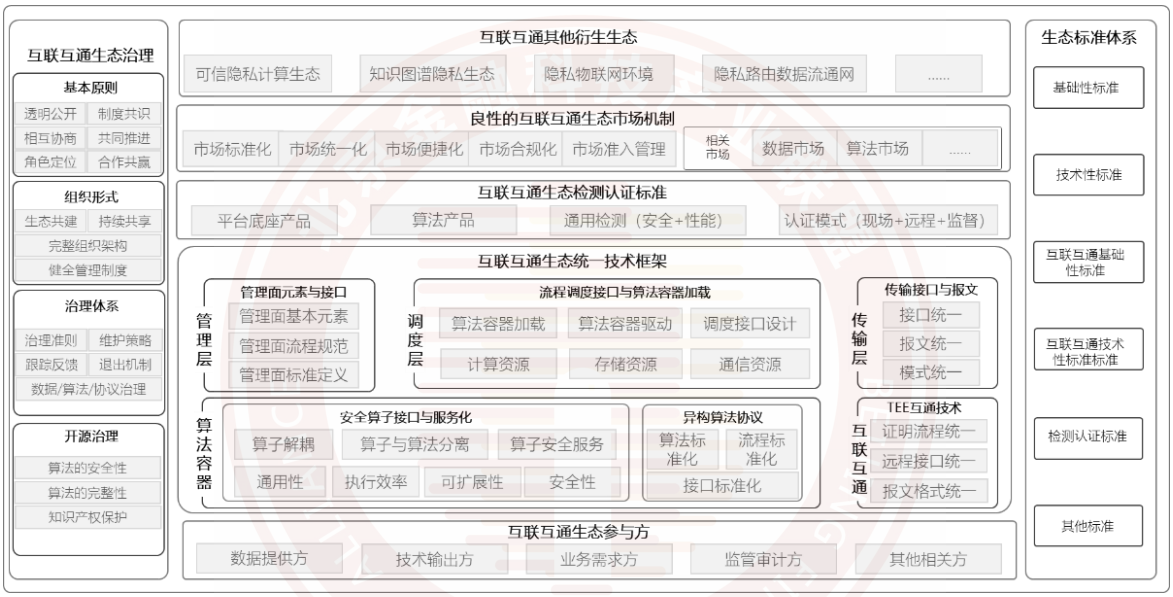

图 57 互联互通生态发展蓝图

如图 57 所示，结合主流的隐私计算技术路线，本子课题从管理面和数据面提出了整体互联互通技术框架实现方案以及具体的落地实现技术路线，推动包括通用型场景、隐私计算技术场景、不同行业等不同颗粒度数据服务朝着互联互通方向发展。

同时，各生态参与方之间，持续在互联互通的框架和标准下优化彼此的职责和范围，促进整体互联互通生态良性的市场发展，反哺标准体系的

完善、技术框架的迭代更新和相关检测认证的逐层优化，形成一个生态共赢的整体良性发展势态，从而带动周边其他衍生生态的配合，逐渐形成具备隐私计算+区块链、知识图谱隐私互联、隐私物联网环境等多技术互联互通的循环生态局面。

(1) 健全的标准支撑

标准化是促进互联互通生态的重要保障，相关部门已发布了一系列隐私计算技术的国家和行业标准，在标准化建设方面取得了长足的进步，但在互联互通层面仍然相对缺乏顶层的规划。互联互通生态标准体系将有机连接各个环节，并为彼此间的协同工作提供技术参考，是实现互联互通、资源开放、数据共享、生态协调的基础（见图 58）。

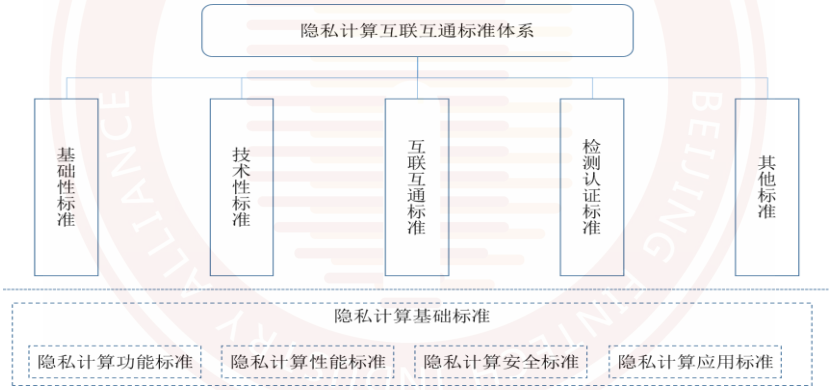

图 58 互联互通生态标准体系组成

(2) 统一的技术框架

通过调研主流的隐私计算技术路线，研究互联互通整体框架实现案例及具体的落地实现技术路线，并验证一些隐私计算产品互联互通，进一步提炼统一技术框架，归纳各子课题间的关系。

针对联邦学习、多方安全计算这类纯软件技术路线的异构平台互联互

通，可自顶向下基于管理面元素与接口、流程调度接口与算法容器、异构算法协议、安全算子、传输接口与报文实现。首先，管理面元素与接口研究对于整个隐私计算互联互通技术研究起到基础性和框架性作用，能明确隐私计算系统中管理面的基本元素，统一管控互联互通的隐私计算平台内外部各元素，维持平台正常运作；其次，流程调度接口与算法容器加载研究是整个隐私计算互联互通技术研究的核心部分，用于加载和驱动底层算法容器，同时统一调度通信、计算、存储等资源；通过异构算法协议研究和安全算子接口与服务化研究，将底层算法解耦成应用算法和通用算子，以容器化实现可独立运行算法和算子服务，使大多数厂商有能力提供典型算法、算子标准接口和流程；最后通过传输接口与报文研究，在异构的隐私计算平台之间建立一套算法和算子相互协作全链路的通信相关规范和参考实现模式。

针对 TEE 这类软硬结合技术路线的异构平台互联互通，通过 TEE 互联互通技术研究，形成统一的远程证明流程和接口，抽象出统一的远程报告格式和 Enclave 属性，实现异构 TEE 平台互认。

(3) 配套的检测认证

隐私计算互联互通生态配套检测及认证项目将依据生态标准体系中的《流程调度接口与算法容器技术规范》等系列检测标准，对有意加入互联互通生态的产品进行系列检测和认证并出具相应检测及认证报告。适用范围包括平台底座产品和算法产品，通过检测和认证的产品便可被认定为从技术上达到了进行互联互通的要求，具备与生态中其他产品进行互联互通的能力。

对于初次加入生态的产品，依据产品形态性质检测内容可有所不同，但总的来说检测和认证需求较为普遍。而对于已经加入生态的产品，检测的内容应仅覆盖该产品在上次检测后发生变更的内容，对于未变更的内容不需要重复进行检测和认证，对于已变更内容需确认其技术上依然满足互联互通要求。

(4) 良性的市场机制

隐私计算领域市场的形成，同样也需要行业标准与消费机制。算法的动态上架与下架、数据的定价交易等，需要先获得由有资质的检测认证机构执行的安全认证，从而保障隐私计算的安全性与机动性。对于市场的内容，需要予以版权保护，保障市场生态与秩序，促进良性竞争；对于市场的生态模式，需研讨符合市场利益的收费机制，确保市场活力与商业模式的长期发展。

(5) 完善的生态治理

隐私计算技术能够互联互通需要基于各个参与方对于生态治理的共识。在接口规范、资源发现协议、身份认证模式等各类基础性的互联互通形态能够共同建设、共同协商、共同修改的基础上，形成一套统一且具备共识的治理章程与规则，才能够保证互联互通生态的持续发展。以联盟团标为基础，确定接口规范（管理面、数据面）、资源发现协议和身份认证的基础版本作为互联互通生态治理共享基线，并就此基线取得生态建设参与方的基本共识，成为生态治理的提案基础。完善的生态治理至少包括算法的安全性、算法的完整性和知识产权保护。

开源治理是隐私计算跨平台互联互通如果要做到具有规模化的生产能力的实现方案之一：

a. 算法的安全性：隐私计算对于算法的安全性尤为重视，因此进行隐私计算跨平台合作的多个参与方，都需要信任加载算法包的安全性；算法安全性验证是一个复杂且专业的事情，其实现代码需要专业的审查，通过多方审议或检测的算法的安全性更容易受到广泛的认可；

b. 算法的完整性：为了实现隐私计算更广泛意义的互联互通，其能够实现和联通的算法一定是足够完整的，闭源产品较难开放全部的算子，开源隐私计算平台框架成为获取更为完整的算法能力的重要方式。

c. 知识产权保护：在开源实现互联互通的同时，注重全面加强知识产权保护工作，促进互联互通体系良性发展，激发生态参与方创新活力，营造优良的交易和治理秩序，为隐私计算相关方知识产权的研发、申请、利用和保护提供保障，推动构建新发展格局。

(6) 其他衍生生态

a. 隐私计算+区块链=可信的隐私计算解决方案

区块链在联邦学习上的应用可以夯实数据拥有者数据的隐私性和系统的安全稳定性。区块链可以确保计算过程和数据可信，而隐私计算实现数据可用不可见，两者相辅相成，实现更广泛的数据协同。隐私计算与区块链技术相融合，为实现数据价值共享提供了一套更加完整与严密的解决方案。

b. 知识图谱隐私互联

知识图谱旨在采用图的结构来建模和记录世界万物之间的关联关系，并沉淀关于万物的知识。知识图谱的相关技术已经在搜索引擎、智能问答、语言及视觉理解、大数据决策分析、智能设备物联等众多领域得到广泛应用，被公认为是实现认知智能和智能互联的重要基石。基于知识图谱实现隐私互联的联邦知识图谱将通过社区和群体的联邦式协作进行构建，具备知识互联、去中心化和知识的可信等三种能力。

c. 隐私物联网环境

物联网的发展，将网络连接和计算能力延伸到了计算机以外的物体、传感器和日常物品，使得这些设备可以在较少人类干预的情况下生成、交换和消耗数据。如今，物联网呈现出规模化（联网设备数量持续增多）、亲密性（可穿戴设备和植入人体的设备等）、普遍性（无处不在、始终联网）、智能化等发展趋势，这可能冲击个人隐私保护，使得个人可以被更容易地识别、追踪、画像和影响。加强物联网相关的个人隐私保护，国际互联网协会提出了四个方面的建议：第一，加强用户对 IoT 设备和服务的有意义控制，从而实现物联网数据管理；第二，提升用户数据收集使用的透明度；第三，隐私立法和政策跟上技术发展步伐；第四，加强多利益相关方的参与。

d. 基于隐私路由的数据流通网络

作为数据流通网络基础设施中的枢纽节点—隐私路由，其作用将举足轻重。从安全性上看，隐私路由需要具有密码学上的安全保证；其次，面对大数据量流通，隐私路由应具有数据快速转换和分发的功能；最后，隐

私路由需要兼容不同技术类型（特别是密码技术）的中间结果转换，提升自身的使用效能。同时注意到，隐私路由的运营方需要具备中立、可信等属性。隐私路由的开发、实现、运行维护将成为未来数据流通相关产业不可或缺的一部分。

（三）生态主要建设路径

互联互通生态的建设不是单纯的技术问题，互联互通生态主要由市场化主体参与，涉及各方面利益，需要通过路径的规划，才能把生态建设的理念转化为实践，能够更好地处理建设中涉及的生态参与方的复杂关系。隐私计算技术和应用发展已取得一些阶段性的成果，互联互通生态建设可在已有互联互通形态试点推广的基础上，在联盟互联互通技术框架内进一步进行技术验证，完善互联互通相关标准和政策，进行算法市场、数据市场等相关市场建设，开展更多业务层面落地应用等方面实现生态的良性发展。

1. 已有形态试点推广

互联互通生态建设实现的形态包括黑盒互通、白盒互通、灰盒互通、基于隐私路由和中间件的互通等。这些形态需要扩大试点推广，形成作为探索数据流通新模式的突破口。

(1) 节点内纵向互联互通统一管控平台

互联互通是打破“技术孤岛”实现数据流通的必经之路，业务需求方可以通过互联互通统一管控平台（见图 59）将多个厂商的产品接入互联互通统一管控平台，进行统一操作管理。

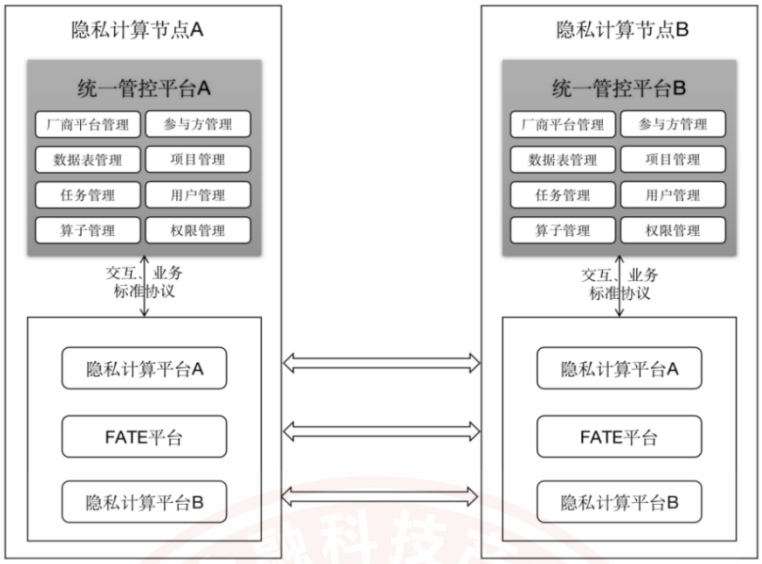

图 59 统一管控平台互联互通示意图

互联互通统一管控平台具有的特点包括统一管控平台凌驾于各家平台，形成事实接口标准；满足多个平台降低运营成本需求；确定性强，短期落地可行性高；易于理解，客户可接受性高；共荣生态，构建一个兼容开放、互利共赢的互联互通生态。

(2) 基于隐私路由的跨平台互通技术

基于隐私路由的互联互通实现方案保留各隐私计算平台的内部自治性（如采用了哪一种隐私计算技术和底层密码运算技术），各平台互联时仅需要接入隐私路由并实现自身的中间结果与某一种特定数据格式的转化即可，无需与所有其他平台两两构造互通关系。“隐私路由”的实现方案充分保证平台自治性，隐私计算平台无需修改已有的计算协议，也无需额外增加支持其他密码算法、协议，或者增加相应的安全假设；同时提供了灵活的扩展性，可兼容各类技术体制下的隐私计算平台动态接入或退出互通网络。

(3) 基于灰盒的异构平台互通技术

异构隐私计算平台之间的灰盒互通技术验证，是从异构平台的隐私计算算法层面实现互通。区别于黑盒互通、白盒互通的思路，灰盒互通是区分异构隐私计算算法及算法协议的设计实现原理和过程，以及算法源码级内容。其中灰盒互通技术验证聚焦于异构平台间隐私计算算法原理和实现过程的统一开发流程说明，实现源码不在其他平台间公开。对比白盒互通实现方法，异构平台间算法在非开源的前提下实现互联互通；对比黑盒互通实现方法，异构平台间仅公开了隐私计算算法原理和实现过程，从算法实现流程上加深互联互通参与方的合作深度，便于开展具体算法协议间的技术验证。

(4) 基于中间件异构平台互通技术

异构隐私计算平台间的基于中间件异构互联互通技术验证方式，参考了互联网数据传输互通实现方法。隐私计算异构平台间的互联互通可以划分成底层通信、中间层传输及顶层应用等层次，通过逐层定义系列标准共识内容，实现规范互联互通。

具体地，异构平台间采用轻量化的中间件方式，分别基于消息队列进行组件化改造、利用中间件实现任务事件转发、贯穿隐私计算各个交互阶段。中间件主要实现的功能包括：算法数据报文重构、任务事件转发、任务状态同步。基于中间件的异构平台互通方式，可将对各平台原生框架的改动量降到最小，便于扩展为多种隐私计算平台对接，具有较好的通用性。

(5) 基于区块链智能合约的异构平台互通技术

区块链是分布式数据存储、点对点传输、共识机制、加密算法等计算

机技术的新型应用模式，具有防篡改、追踪溯源、公开透明的特性。而智能合约则是基于链上可信的不可篡改的数据执行模式，可自动化执行预先定义好的规则和逻辑。

基于区块链技术的智能合约机制，链上交互标准统一、基础设施完善、合约复杂度较低，能够较快完成跨平台互通。结合区块链的特性，可以有效实现隐私计算任务的链上审计和溯源。

2. 技术验证

技术验证是在已有互联互通技术框架的基础上进一步验证技术的实用性和可靠性，为大规模应用奠定基础。技术验证的重点在于对已有互联互通形态互联互通接口的验证，定义标准化的互联互通接口，成为构建行业互联互通方案的关键技术基础。因此，开展互联互通技术验证，可以围绕互联互通框架相关功能模块及接口，从以下四个方面实施：

(1) **确定可量化互联互通技术指标：**是确定技术验证目标的基础。互联互通的能力，是可以被明确定义和拆解的，在各个能力点上量化互联互通能力，为技术验证提供了明确的方向。

(2) **具象化检测对象：**标准化的接口带来了互联互通系统各个模块细粒度的划分，明确了模块化职责，提升了技术验证的有效性和针对性，更加聚焦验证的关注点。例如通信模块关注协议信道打通与稳定性，算法与算子模块关注计算性能与安全性，管理模块关注功能的完备性，调度模块关注其标准满足情况及其兼容性等。

(3) **采用多元化技术验证方法：**对不同的模块，针对性和关注点不一样，采用的验证方法也会多样性，除了原有的人工功能检测外，更多使

用面向接口的自动化检测：对于面向容器化的框架基座，引入镜像扫描的标准性检测的方法；对于通信模块，可引入鲁棒性检测，协议兼容性检测等验证方法。

（4）**研发复用化技术验证工具：**面向标准化通用化的接口，可以开发出能够复用和持续完善的验证工具，对不同的产品，不同模块，分别建立验证用例集与评价指标体系。

隐私计算互联互通技术验证在实践中根据业务需求和业务形态有不同的实现形式。以跨平台互通技术在联盟联通、数交所联通模式为例，可采用实验室网络模拟方式在联盟（或其代表机构）、数交所部署隐私计算路由以使其成为各隐私计算平台技术验证环境的重要组成部分，互通网络结构如图 60 所示。

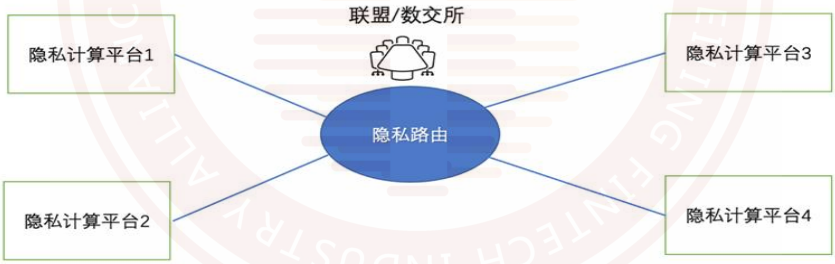

图60 互通网络结构示意图

在该互通网络结构中，各隐私计算平台为被验证对象，隐私路由为陪测环境的一部分。在两个隐私计算平台上分别部署不少于一个数据源，并结合自动化检测方法和工具，充分验证平台间的互联互通效果。

3. 标准和政策建设

针对不同需求，一方面由行业联盟牵头制定完整的标准体系，编制相

关标准体系表，并组织编写相关标准。另一方面由互联互通生态参与方根据不同角色定位，在国际标准、国家标准、行业标准、团体标准或地方标准各层面完善适合自身特点的标准体系。

其中，已完成的团体或地方标准等一方面可以争取向上申报，尝试在应用实践的基础上升为行标等扩大标准影响力，另一方面根据技术发展情况和市场需求对标准体系表进行维护和更新。

政策方面，可通过组织编写研究报告和技术白皮书等方式向上级领导单位反映互联互通对于政策方面的需求和发展建议。

4. 相关市场建设

在隐私计算互联互通命题下，推进相关的算法、数据等市场建设，除了可能收获上述共性价值外，还对隐私计算的商业化实践、数据要素的多维度流通、技术创新的场景化应用、知识产权的有效保护具有推动作用。因此，在隐私计算互联互通命题下，需要重点讨论数据市场、算法市场的前置建设。

(1) 数据市场建设

公开市场数据交易是数据要素市场化配置、参与社会生产、贡献要素价值的最主要方式。在既往实践中，个人隐私保护、数据利益保护、数据安全要求等几项基本前提逐步确立，也引申出数据可用不可见、用途可控可计量的隐私计算技术目标。

(2) 算法市场建设

目前，算法市场主要有三种模式，一是头部企业所搭建的开放平台和AI 市场，提供开发工具、通用模型、数据集等全量支持；二是垂直领域的

AI 公司，利用自身的业内资源和口碑而建立的行业算法市场；三是创业公司搭建的独立算法市场。隐私计算算法市场建设，除了在交易规则、信息匹配、分润机制等方面的一般考虑外，还需着重关注标准适配、场景分类、知识产权有效保护等特殊需求，打造相较场外市场的效率性、完整度优势。

a. 标准适配与场景分类

结合现有隐私计算技术框架，算法市场需要依托（或推动制定）协议标准、接口标准、实现标准、审计监督标准、安全性度量标准，对场内算法进行索引和评价，从而保障算法模块在不同隐私计算框架的流通。

b. 算法知识产权保护

算法知识产权保护方法的延续性、适应性、专业性是实现有效保护的主要挑战。算法市场主体信用体系可以使用联盟链方式构造，确保技术厂商、业务厂商、机构用户、鉴定机构等主体在平等、安全、自治的前提下共建可信市场。将知识产权鉴定机构引入算法交易市场，提供标准化、可追溯、全链路的认证服务能力，帮助事后更好识别算法特征和权利主体。在算法交易记录加密链上存证基础上，探索算法开发权限的 token 化授予和流转，提升算法知识产权保护的延续性。

隐私计算互联互通生态市场化发展，除数据市场和算法市场外，还可能出现模型市场、算力市场等进一步细分的市场。各类市场的有机联动将共同推动互联互通生态的进一步快速发展。

5. 业务层面落地

(1) 互联互通的不同业务形态和相应落地方式

当前业内主要有三种互联互通的业务形态，对应的落地方式也不同，各机构需要根据实际场景灵活选择。**一是监管类、数交所或数据流转平台模式**，该模式由监管机构、数交所或行业中立机构主导的一对多参与机构，场景一般由牵头单位制定互联互通的各项要求，参与机构必须按照要求加入合作中来，适用于有一家主体单位主导的有强监管需求的场景，或者是根据部分行业特性与需要出现的由行业中立机构主导的数据流转平台场景；**二是大行自有生态模式**，该模式一般会由一家金融机构牵头，再将与其合作的外部机构按照各方达成共识的互联互通方式引入网络生态，该模式的优点在于生态中的各节点在与牵头金融机构合作的同时，均可以自由合作，网状结构便于场景快速扩展；**三是点对点的业务需求模式**，该模式适合少数几家机构达成互联互通方案共识后的快速落地试点，但不适合大范围扩展应用。图 61 展示了通过组建由多机构组成的联盟的方式，提供防范多头借贷的新措施。

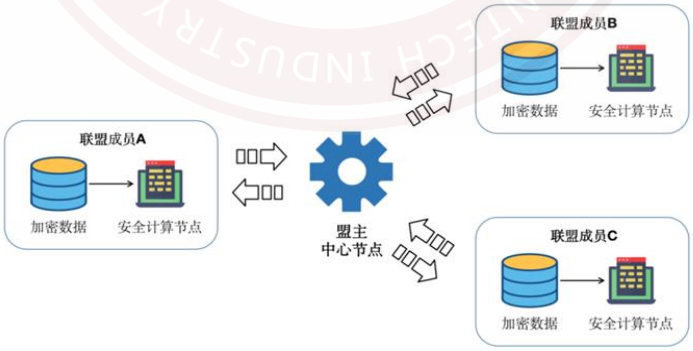

图 61 金融领域多头借贷多机构联盟示意图

(2) 互联互通场景驱动

作为数据流通产业的基础建设，隐私计算互联互通在业务场景应用中能更好地平衡安全和效率，典型的业务应用场景包括营销场景、风险场景以及监管场景等。

a. 营销场景驱动

营销场景以精准定位目标客户，提升营销收益为目标。金融机构自有用户数据和特征有限，联合跨机构、跨行业的数据，可以拓宽数据样本量、丰富数据维度，从而实现高效触达存量客户、精准纳新拓客的效果。营销场景下的数据提供方主要是金融机构、互联网平台、运营商、征信机构、政府部门等，这些数据提供方数据一般都与自身的隐私计算节点绑定输出，因此跨机构、跨行业的联合营销场景具有较强的互联互通驱动力。

b. 风险场景驱动

风险控制具有信用评级、贷款风控、授信额度、黑名单查询等广泛的场景需求，可以控制金融机构风险、降低不必要的损失、提高经济效益。基于隐私计算互联互通的联合风控，可以整合隔离在不同领域和行业的数据，在各方原始特征不出域、计算环境不改变的前提下，建立高精准度的风控模型，提升风控质量。对于金融机构和公安部门，风控场景具有较强的内生驱动力，互联互通联合风控场景需求也多产生于金融机构。

c. 监管场景驱动

随着网络生态的发展，网上欺诈和跨境犯罪的形势越来越严重，中国人民银行、国家金融监管管理总局等监管部门出台了多项治理措施，所属金融机构积极行动，在反洗钱、反欺诈识别和检测可疑交易等场景下，驱动隐私计算互联互通的推广落地。

六、总结与建议

（一）总结

金融行业隐私计算互联互通，既是金融机构实施“以用户为中心”服务转型过程中对标监管方关于合规化数据要素流通要求的行动方向，也是金融行业积极落实中国人民银行印发的《金融科技发展规划（2022-2025年）》中关于“探索建立跨主体数据安全共享隐私计算平台”要求所必须开展的工作。隐私计算 1.0 时代已破解了以机构为主体的数据孤岛问题，然而 1.0 时代所依托的异构隐私计算平台在发展的同时也催生了高昂的数据要素合规流通成本，不利于行业生态持续、良性的发展；基于此，金融行业隐私计算互联互通工作开始被提上议程，并被金融机构、互联网平台企业、金融科技公司、金融检测认证机构、隐私计算开源社区等多个产业方所密切关注。

本课题以金科联盟平台为依托，经过51家金融机构、互联网平台企业、金融科技公司、金融检测认证机构、隐私计算开源社区等参与单位协同攻关，针对隐私计算特点解耦了隐私计算系统架构，实现了“管理面与数据面解耦”及“隐私计算系统底座与算法解耦”，并将该架构进一步拆解为“管理面元素与接口研究”“流程调度接口与算法容器加载研究”“安全算子接口与服务化研究”“传输接口与报文研究”“异构算法协议研究”“TEE互联互通技术研究”6个部分。随后，在陆续攻关了各部分所涉及的“管理面最小必要元素设计”“多方资源访问控制协同策略”“DAG & CONF的通用化设计”“流程调度互通设计”“组件容

器化加载方案”“基于最小化约束的算法容器设计”“隐私计算安全算子服务设计”“传输层同步异步兼容设计”“异构平台开放算法协议设计”“TEE统一远程证明流程设计”等技术难点问题后，本课题还从互联互通生态建设目标、生态蓝图、生态主要建设路径展望了行业生态的发展前景，为后续的互联互通接口定义与对接验证奠定了坚实的基础。

（二）发展展望

隐私计算互联互通的目标立足于行业级互联互通，旨在实现“行业化、平台型、生态化”的愿景。在强监管的主基调下，金融业隐私计算互联互通技术有望在未来一段时间内成为合规的重要基础能力，并在金融科技的创新侧和监管侧发挥重要作用。

课题组认为当前这项工作尚处于小范围试点阶段，距离大规模推广乃至成熟的商业模式尚需作出不懈的探索。总体而言，下一步规划应考虑如何在满足合规要求下，充分调动隐私计算互联互通生态各方的积极性，营造良性互联互通行业生态。

一是需要配套的政策环境引导。政策面上应积极推动基于隐私计算互联互通技术层次化解耦的标准与认证工作，并以行业应用为抓手明确隐私计算互联互通的能力成熟度，使更多机构有动力加入隐私计算互联互通生态构建工作中，引导行业逐步从自治试点向规范、有序隐私计算互联互通过渡。

二是需要积极探索隐私计算互联互通关键技术的实施方案，保持必要兼容性，避免因出现新的门槛而增加各方改造成本。在充分尊重主流异

构隐私计算平台技术路线差异基础上，从南北向和东西向两个维度综合评估隐私计算互联互通框架解耦后每层接口的不同实现方案，并优先解决紧迫性应用需求。

三是需要加强业务需求方与技术提供方的联动，重点解决场景和平台互联互通技术验证。安全与效率的平衡一直以来是隐私计算技术发展的难点，不能脱离具体场景需要而独立存在。不同业务场景的数据流通频次、密度、体量等往往有所差异，作为场景应用基础设施的平台在技术路线兼容方面也有所差别，因此，需选择代表性场景对隐私计算平台进行系统性测试，以便更好贴近金融行业数据要素流通需求。

（三）政策建议

金融行业隐私计算互联互通的发展具有强烈的合规属性，因而政策导向在守正创新方面具有举足轻重的作用。当前，隐私计算互联互通处于发展的关键阶段，良性生态建设刻不容缓，“如何推动隐私计算技术与数据要素市场建设协同发展，构建政府、企业、社会多方协同治理模式”成为生态参与方的普遍关注点。然而，由于隐私计算相关技术应用仍在不断实践演进中，关于各类业务场景如何落实创新数据治理，还需要更多实施细则和配套政策进行指导，以便更好明确改造落地的实施导向和推进标准布局、产业激励等方面工作，加快助力数据要素在合规前提下寻找安全与效率的平衡点，深化数据融合，赋能实体经济。

1. 加强政策引导

(1) 建议监管部门推进互联互通相关规范落地，鼓励金融行业积极

采用隐私计算互联互通框架开展多方数据合作相关业务。在合规前提下提升多方隐私计算合作效率，并在实践中逐步夯实互联互通框架对金融行业场景应用的支撑力度。

(2) **建议建立行业级数据安全产品认证检测体系，形成产品质量保障机制。**循序推进隐私计算产品、服务和流程建立认证认可体系，根据行业实践积累经验，逐步推动相关行业标准立项，规范相应产品研发和检测认证流程。既要为隐私计算技术服务商提供透明、科学、客观、可验证的技术测评，不断提高产品和服务质量，也要在产品选型和业务运营上为金融机构提供评估验证和指引，有效降低碎片化评估成本，促进产业的健康快速发展。

(3) **建议建立统一的认证认可法律体系、制度体系、组织体系、标准体系和监管体系，构建起“统一管理，共同实施”的行业认证认可体系。**建议认证认可流程由独立第三方机构具体实施，政府部门负责制订政策、建立制度并实施监管，社会各方采信评价结果。统一规范市场准入要求和市场评价规则。

2. 推动联盟建设

(1) **建议组织推动建设适应金融行业发展需要的隐私计算互联互通联盟。**长期推动互联互通规范的演进与生态建设发展，从数据流通全生命周期运维高度规划隐私计算互联互通体系，逐步建立隐私计算互联互通运营机制及评审报备制度。

(2) **建议使优质参与方有更多的合作机会，促进良性循环。**建立资源贡献度评估和激励机制，允许将算力折算为资源加入互联互通生态，激

励参与方积极地参与到生态中，优化联合建模与推理，继而为互联互通生态的各参与方提供优质模型和系统服务，促进良性循环。

3. 完善知识产权保护

建议强化场景算法和安全算法的知识产权保护力度，促进商业隐私计算与开源隐私计算社区协同发展及各自优势发挥，严格保护商业隐私计算算法知识产权和隐私计算技术服务商权益，完善技术产业生态，引导业界基于互联互通基础框架形成技术演进与数据业务相互促进的商业模式。

4. 规范市场建设实施路线

(1) **建议推动隐私计算相关市场建设，明确运营主体**。构建涵盖数据提供方、技术输出方、业务需求方、监管审计方在内的生态化产业体系。

(2) **建议监管部门在管理层次上，先推荐、再鼓励，最后形成统一标准。**统筹应用方、数据方、平台供应商等相关方，遵循逐步渐进的方式，先从小范围试点开始，逐步推进；同时，充分用好监管沙箱机制，完善出箱后再落地的衔接制度，最后形成全市场统一的格局。

附录：重要术语

术语	定义
节点	隐私计算生态中的抽象功能单元，用来指代由机构或组织部署的隐私计算平台。
数据集	可供参与隐私计算的数据资源。
项目	面向特定目标的，提供一项独特产品、服务或成果的隐私计算方案，该实体为可选项。
组件	独立执行隐私计算任务的模块单元，其经过封装、符合开放接口规范、可以完成某个特定计算或算法，可独立部署。
流程	采用DAG结构定义的、可编排的隐私计算作业运行模板。
作业	一个隐私计算流程通过配置运行参数后的运行实例。
任务	组件运行实例的载体，每个运行实例通过任务来管理。
模型	指通过隐私计算技术训练完成，可用于进一步推理的模型，或是已有的现成模型。

参考文献

[1] 中共中央国务院. 关于构建数据基础制度更好发挥数据要素作用的意见 [EB/OL][2022-12-19].http://www.gov.cn/zhengce/2022-12/19/content_5732695.htm.

[2] 中共中央国务院. 关于坚持和完善中国特色社会主义制度推进国家治理体系和治理能力现代化若干重大问题的决定 [EB/OL][2019-11-05]. http://www.gov.cn/zhengce/2019-11/05/content_5449023.htm.

[3] 中国人民银行. 金融科技发展规划（2022-2025年）[EB/OL][2021-12-29]. http://www.pbc.gov.cn/goutongjiaoliu/113456/113469/4438627/index.html.

[4] Stepehn Hardy, Wilko Henecka, Hamish Ivey-Law, et al. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. [EB/OL]https://arxiv.org/abs/1711.10677.

[5] Chaochao Chen, Jun Zhou, Li Wang, et al. When homomorphic encryption marries Secret Sharing: secure large-scale sparse logistic regression and applications in risk control. [EB/OL] https://arxiv.org/abs/2008.08753.

[6] Dung Bui, Geoffroy Couteau. Improved private set intersection for sets with small entries. [EB/OL] https://eprint.iacr.org/2022/334.

[7] Vladimir Kolesnikov, Ranjit Kumaresan, Mie Rosulek, et al. Efficient batched oblivious PRF with applications to private set intersection. [EB/OL] https://eprint.iacr.org/2016/799.pdf.

[8] Kewei Cheng, Tao Fan, Yilun Jin, et al. Secureboost: a lossless federated learning framework. [EB/OL] https://arxiv.org/abs/1901.08755.

[9] 全国人民代表大会. 中华人民共和国个人信息保护法[EB.OL] [2021-08-20].http://www.npc.gov.cn/npc/c30834/202108/a8c4e3672c74491a80b53a172bb753fe.shtml

[10] 中华人民共和国中央人民政府. 征信业务管理办法[EB/OL][2021-09-27]. http://www.gov.cn/gongbao/content/2021/content_5654782.htm.