转博申请

当前研究项目包括《**双层一体安全高性能区块链智能合约语言关键技术研究**》和《**非开源联盟链基础平台**》，重点开发领域特定语言CryptLang，并设计了策略隐私公平访问的去中心化访问控制系统，并发表了关于**去中心化访问控制、去中心化身份、自动密码任务代码生成和分片区块链无人机系统**的论文。博士阶段的规划包括**非开源联盟链平台的访问控制与密态计算、基于区块链的数据要素市场关键技术研究**，以及零知识条件支付。毕业论文研究了如何利用区块链和密码学技术实现**三层数据主权系统架构**，确保数据的安全性、隐私性和控制权。

1 现有工作

接下来将分别介绍目前研一和研二已完成的项目和论文工作。

1.1 项目工作

1.1.1 双层一体安全高性能区块链智能合约语言关键技术研究

我负责课题5《智能合约领域特定语言族(DSLs)及领域示范应用》中的民生领域DSL设计。

背景：本课题主要研究领域模型及智能合约领域特定语言族。采用分层设计的理念构建语言族框架。将DSL分为表达层、知识层、语义层。表达层简化智能合约编码过程，知识层抽象和封装领域知识，语义层提供不同领域的DSL可组合和扩展的模型，设计面向金融、政务、民生等领域的DSL。

任务：如下图所示，本课题主要包括三项子任务。各个子任务的主要研究思路与方案介绍如下。

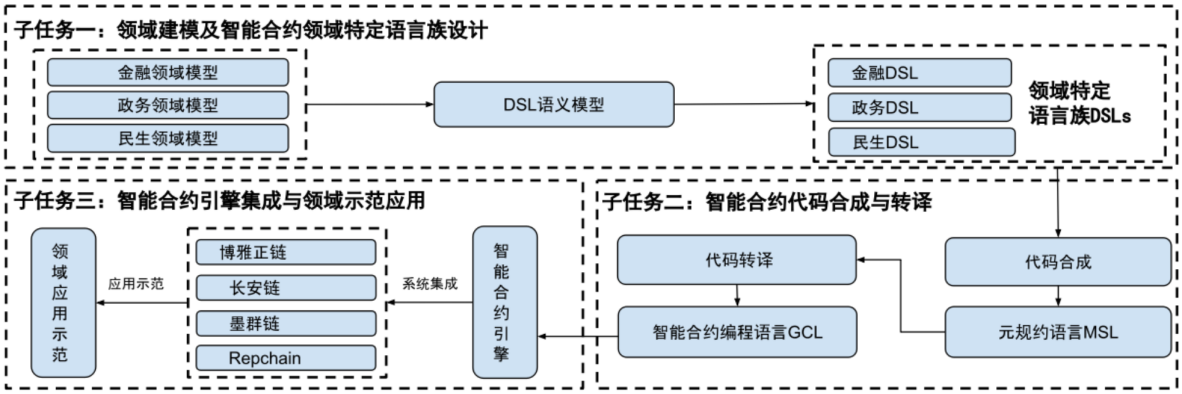


图 1 课题5子任务示意图

子任务一：领域模型及智能合约领域特定语言族设计

1. **DSL的语法应该易于理解和记忆，同时应该与领域相关的术语和概念保持一致。**

   我设计的民生领域DSL主要针对医疗隐私数据共享的应用场景：

   医疗数据共享应用针对当前医疗临床科研数据共享存在的“不愿、不敢、不能”问题，采用区块链+分布式统计、联邦学习等技术，将本地、中心式处理转为联邦学习、群体协作模式，达到在数据不出医院的前提下实现跨机构间数据安全共享协同分析，在保证数据隐私的前提下提高医院间的医疗数据安全共享能力，充分发挥医疗数据要素价值。

   虽然BPMN的图形界面能较为有效地满足上述应用的基本语义需求，但在涉及隐私数据共享及统计等核心问题时，如医疗数据共享和旅游信息管理，便需要依托密码学的相关技术，包括数字签名、承诺机制和零知识证明等，以确保其实现。鉴于开发人员可能缺乏深入的密码学知识，并未必熟悉以太坊提供的底层密码学API，涉及复杂密码学任务的构建过程可能面临潜在的安全风险。因此，本研究提出了一种专门针对密码学应用而设计的领域特定语言——CryptLang，旨在解决民生领域的隐私保护问题。该语言的设计初衷是为不太熟悉密码学的开发者提供一个简易的操作平台，帮助他们在民生领域轻松完成复杂的密码学任务。CryptLang可以被集成到BPMN的整体架构中，使用户能够在BPMN网页界面的侧边栏直接选择CryptLang任务框，并在其中进行编码以实现相应的密码学任务。值得一提的是，CryptLang不仅适用于民生领域，同样也能在政务或金融领域作为核心的密码学组件发挥作用。

2. **语法应该允许用户轻松地表达其意图，同时减少可能出现的错误。**

   CryptLang作为一种领域特定语言（DSL），专门为智能合约中的密码学任务而设计，旨在使开发人员能够轻松掌握并应用其内置的加密特性。当CryptLang集成到BPMN框架中后，它不仅拓宽了密码学操作的范畴，还能在民生领域中高效地处理隐私保护问题。

   为更直观展示其应用，以下是一个具体示例：CryptLang帮助医院等医疗记录存储机构快速生成验证签名并提供医疗记录的合约，具体代码如下所示：

```
// CryptLang
contract MedicalRecord {
    function permit(address _owner, address _hospital, uint256 _year) public
{
        @ECDSA with SHA3-256 (_owner, _hospital, _year);
        _search(_owner, _hospital, _year);
    }
}
```

   CryptLang的大部分用法和功能都与Solidity类似，只有用到“@”操作符时才真正使用到CryptLang的功能。建议在编写CryptLang程序时，使用的函数变量名以“_”开头，以便与代码生成后新添加的变量进行区分。

子任务二：智能合约代码合成与转译技术

1. **如何支持针对新领域设计的DSL编写的智能合约在区块链上运行。**

   下文将对CryptLang这一专为处理高级密码学任务而定制的工具进行深入探讨，并详尽解释其工作流程以及核心语法和语义。

   语法：CryptLang基于Antlr实现词法、语法分析、代码翻译与转义。下图展示了CryptLang的语法规则定义。

```
taskSymbol
 : '@' ;

addrSymbol
 : '#' ;

privateIdentifier
 : privateSymbol? identifier ;

privateIdentifierList
 : ( privateIdentifier? ',' )* privateIdentifier? ;

hashMethod
 : 'SHA3' | 'SHA2' | 'RIPEMD' ;

signatureMethod
 : 'ECDSA' | 'RSA' | 'BLS';

commitmentMethod
 : 'Pedersen' | 'Merkle' ;

proofMethod
 : 'Groth16' | 'PLONK' ;

signatureStatement
 : statementSymbol signatureMethod ('with' hashMethod)? '(' ('#'
identifier)? ( ',' identifier)* ')';

commitmentStatement
 : statementSymbol commitmentMethod ('with' hashMethod)? '(' (identifier?
',')* identifier? ')';

taskStatement
 : (signatureStatement | commitmentStatement) ';' ;

otherStatement
 : .+? ';' ;
```

   语义：下文将详细介绍所提出的领域特定语言（DSL）的操作语义。在本节中，我们将专注于描述该语言核心的密码学语义，尤其是针对民生领域中的隐私保护功能，而其他功能则由外部的BPMN框架进行描述和管理。正式定义中，语义配置被明确定义为 $\mu := (vars, R, PoolM, Nc)$，其中 $vars$ 代表程序变量，R 是验证结果，$PoolM$ 包含了先前提交的链下证明集合（如签名、哈希前像），而 Nc 是签名者的nonce。

程序变量以元组 $vars$ = (M, H, I, Pr, Cid, Ad) 表示，其中 $M \in$ M 为密码学方法，$H \in$ H 为哈希方法，$I \in$ I 包含链上数据，$Pr \in$ P 是链下证明，$Cid \in$ C 代表验证合同的区块链标识符，而 $Ad \in$ A 是验证合同的地址。

符号 μ = [[a]] ⇒ μ' 表示在执行操作 a 后，系统状态从 μ 转变为 μ'。CryptLang的语法与其语义均已整合密码学类别。其中，验证结果 R 是决定后续操作的关键变量。同时，转变后的结果 R' 与众多变量有关，这包括密码学算法 M，哈希方法 H，链上输入 I，链下证明 Pr，池 $PoolM$，以及区块链标识符 Cid，地址 Ad 和 nonce Nc。

为了直观解释，验证函数 $V\ RFM$ 可被划分为两部分：首先验证链上输入 I 与链下证明 Pr 的一致性；其次，检查已提交的链下证明 Pr 是否已被使用。在签名过程中，nonce Nc 被嵌入到已签名的消息内容中，因此每次成功验证后需要递增相关签名者的地址nonce。在承诺处理中，更为高效的方法是利用$PoolM$ 作为存储先前提交的 Pr 的仓库。随后，验证过程会检查新的 Pr 是否已存在于 $PoolM$ 中。CryptLang编译器将自动执行上述流程。

2. **如何支持DSL和新型通用智能合约编程语言编写的智能合约交互。**

首先，开发人员使用CryptLang描述加密算法和链上输入数据。随后，编译器通过解析器对CryptLang代码进行分析，生成相应的抽象语法树（AST）。为确保生成的代码安全可靠，系统将参照权威公共存储库（如GitHub上的OpenZeppelin、RSA签名库和验证Merkle证明的库）中的标准密码模板和AST，生成对应的Solidity代码。最终，这些Solidity代码会部署到以太坊上，供用户提交并验证其签名。

下面给出一个示范应用：

通过在博雅领域智能合约编辑软件v1.0图形界面编辑基本业务流程和民生领域智能合约编程语言CryptLang，工作人员能够快速实现中医药临床数据中患者人口学信息、就诊信息、检查检验信息等各类医疗业务信息的隐私保护逻辑，并生成智能合约代码，简化了智能合约的开发流程，降低了系统研发成本，有效支撑中医数据标准化管理、智能合约授权、计算模型发布、科研任务联合分析等应用服务。

针对这一场景，本项目基于智能合约领域特定语言族，设计并实现下述智能合约，利用智能合约领域特定语言族中的CryptLang模块，帮助医疗公司在链上发布数据收集任务，并于拥有对应数据的医院科室进行对接。通过CryptLang语言快速实现链上的签名验证，零知识证明等密码学任务，帮助公司和医院间的数据隐私计算。该成果可以较好地和现有系统集成，使用期间运行正常，达到了预期效果。

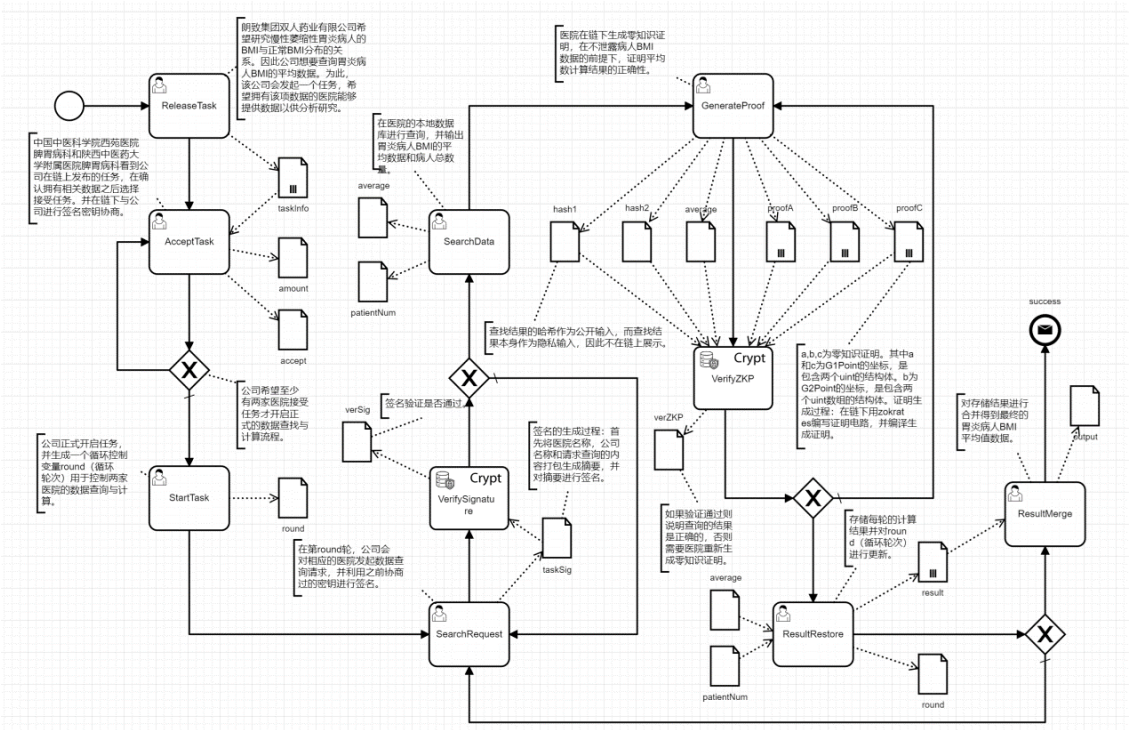


图 2 利用智能合约领域特定语言族编写的民生领域示范应用

4. **如何保证通过DSL编写的智能合约的正确性和安全性。**

   为确保生成的代码安全可靠，系统将参照权威公共存储库（如GitHub上的OpenZeppelin、RSA签名库和验证Merkle证明的库）中的标准密码模板和AST，生成对应的Solidity代码。接下来我们分别定义四类密码学方法：

   哈希：SHA3-256即solidity中的keccak256，SHA2-256即solidity中的sha256，RIPEMD160即solidity中的ripemd160。在以太坊上，哈希函数被广泛应用于各种场景，包括数据完整性验证、智能合约地址生成、加密和安全性等方面。在以太坊中常用的哈希函数有SHA3-256(keccak256)，SHA2-256(sha256)和RIPEMD-160(ripemd160)。它们都可以由以太坊原生的密码学接口直接实现，keccak256可以通过调用以太坊的opcode，sha256和ripemd160可以通过调用以太坊内置的预编译合约来实现。

   签名：分别为ECDSA的签名验证和BLS的签名验证。签名验证是另一个在以太坊上的常见应用，可用于身份认证，授权和访问控制，安全性验证等等。签名验证是以太坊上保障安全性和可信性的重要机制之一，通过在交易和消息中使用数字签名，可以确保只有拥有相应私钥的用户能够执行特定操作，并防止网络中的欺诈行为。以太坊中最常用的签名方案为ECDSA，因此以太坊的第一个智能合约就是secp256k1曲线上的ECDSA签名验证预编译合约。通过调用合约，可以恢复出签名者的公钥相关联的地址，从而验证签名的有效性。然而ECDSA不能进行签名聚合，BLS签名可以进行聚合。BLS签名聚合是一种基于pairing的签名聚合方案，它可以将多个签名聚合成一个签名，从而减少交易的大小和成本。然而使用BLS签名聚合时，需要运算G_T上的乘法，目前以太坊并没有提供相应的预编译合约，但是有望在下一次升级中加入BLS-12-381曲线上的pairing预编译合约。

   承诺：分别为Pedersen承诺和Merkle承诺。承诺是密码学中的一个重要概念，通过承诺可以将某个值隐藏起来，同时保证该值不会被篡改。承诺可以用于实现零知识证明，可验证计算，可验证随机函数等等。区块链上的常用承诺协议有Pederson承诺和Merkle承诺。Pederson承诺是一个基于离散对数的承诺方案，可以通过调用以太坊的modExp预编译合约实现。Merkle承诺是一个基于哈希函数的承诺方案，同样可以通过调用以太坊的预编译合约实现。

   零知识证明：分别为Groth16证明和PLONK证明，均为zk-snark。零知识证明是密码学中的一个重要概念，它可以用于证明某个语句的真实性，同时不泄露任何关于该语句的信息。零知识证明可以用于实现隐私保护，可验证计算，可验证随机函数等等。区块链上的常用零知识证明协议有Groth16和Plonk。Groth16是一个基于pairing的零知识证明方案，可以通过调用以太坊的pairing相关预编译合约实现。Plonk是一个基于多项式的零知识证明方案，同样可以通过调用以太坊的预编译合约实现。

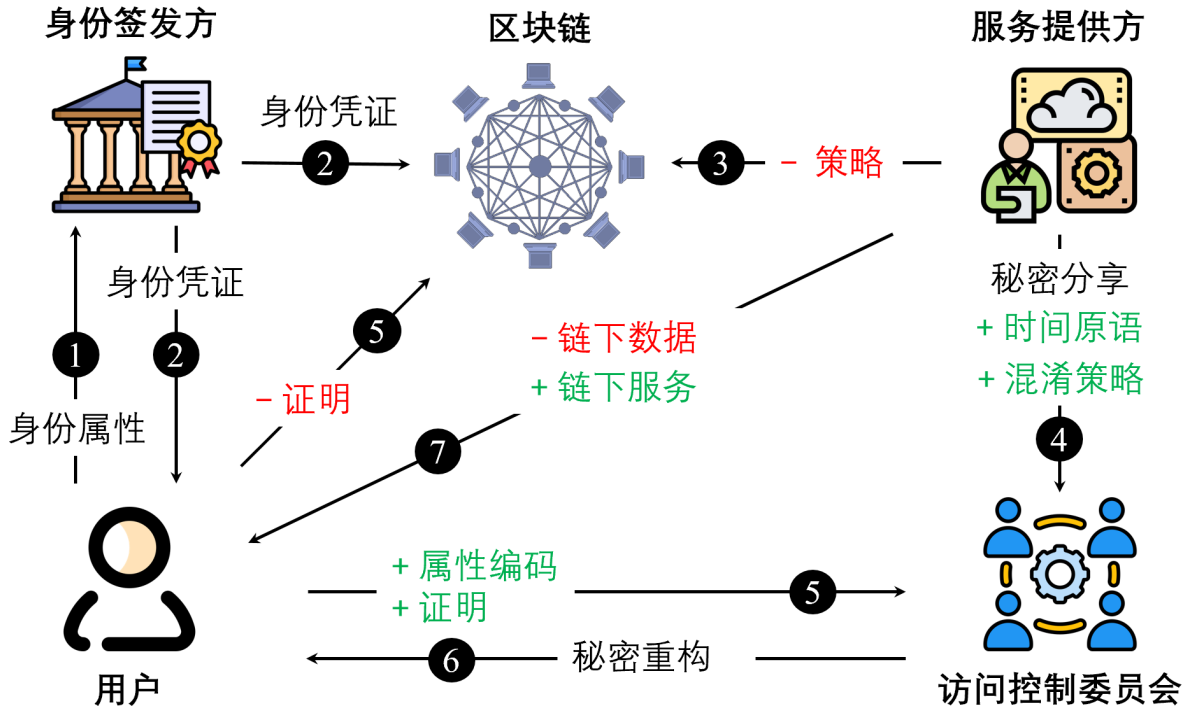
1.1.2 非开源联盟链基础平台

我负责课题3《面向区块链的多层次多维度安全和隐私保护体系》中的访问控制方案设计。

背景：围绕联盟链在国产密码应用、高隐私保护等方面的短板，研究密钥管理、身份认证、数据加密等基于国产密码的链内多维度安全保护技术方案，支撑国家核心领域应用的高安全和隐私保护要求。

任务：研究基于群签名的通用场景身份与密钥管理技术，支持用户的动态加入及恢复，保证用户的身份、用户属性以及场景要求设计不同粒度的数据访问控制，设计基于属性的身份管理技术，实现灵活的访问控制。

我设计了一个能够实现**公平访问，策略隐私的去中心化访问控制系统。**



系统角色

- **用户**：拥有多个凭证的用户可以根据特定政策展示其资格，从而获得指定服务的访问权限。用户可能需要高质量的数据集和多样化的服务，特别是在涉及支付时。
- **身份签发方**：身份签发方是一个验证和存储用户身份的组织，同时负责凭证的颁发。
- **服务提供方**：服务提供方，可以是个人或组织，旨在通过设计访问政策来限制用户属性，以提供有价值的数据或服务以获取额外利润。对于重要的数据集，它倾向于仅分享计算结果，并将访问政策保密。
- **访问控制委员会**：由n个节点组成的委员会，有权向符合政策的用户披露服务提供者的秘密，但需得到t +1个成员的同意。由服务提供者选定进行秘密托管，他们通过参与秘密重构而获利。
- **区块链**：一个恢复已发行凭证列表和访问历史的平台。

协议构建

Setup(1^λ) :

1. Let $\mathcal{R}$ be a circuit with public input cred and EI asserting: EI = Ec$(e,$ attrs$)\wedge$ cred opens to (nk,attrs)
2. Compute $\text{crs}_\mathcal{R} := \text{G16.Setup}(1^\lambda, \mathcal{R})$
3. Let $\text{pp} := (1^\lambda, \text{crs}_\mathcal{R})$ and return pp

IssueReq$_U$(pp, attrs) :

1. Sample r_{cred}, nk // commitment nonce, pseudonym key
2. Commit cred := Com(nk, attrs; r_{cred})
3. Let $\omega :=$ (nk, attrs, r_{cred})
4. Send (ω, cred) to A and receive a proof θ attesting to cred $\in$ MT′

IssueGrant$_A$(pp, MT, cred, ω) :

1. Check cred = Com(nk,attrs; r_{cred})
2. Let MT′ := MT.Add(pk^U, cred) and $\gamma :=$ MT′.Root
3. Let $\theta :=$ MT′.Prove(pk^U) // θ attests to cred $\in$ MT′
4. Send θ to U and publish γ

SetPolicy$_P$(pp, S, $(f_\phi, param_\phi)$, T_1, T_2) :

1. Sample s // secret
2. Let $(\text{F}_\phi, (e^U, e^P), d) := \text{Gb}(1^\lambda, f_\phi)$
3. Let EP := Ec$(e^P, param_\phi)$
4. Let $\text{pp}_{\text{PVTSS}} := \text{PVTSS.Setup}(1^\lambda, T_1, T_2)$
5. Let $(\{c_i\}_{i\in[n]}, \pi_D) := \text{PVTSS.Sharing}(\text{pp}_{\text{PVTSS}}, s, \{\text{pk}^{C_i}\}_{i\in[n]})$ // locked encrypted shares $\{c_i\}_{i\in[n]}$ with time parameter T_1
6. Send (S, F_ϕ, EP, d, $\{c_i\}_{i\in[n]}$, π_D) to each party $C_i \in C$

VerifyShare$_C$(pp, pp_{PVTSS}, $\{c_i\}_{i\in[n]}$, π_D, F_ϕ, d) :

1. Check $\text{PVTSS.Verify1}(\text{pp}_{\text{PVTSS}}, \{\text{pk}^{C_i}, c_i\}_{i\in[n]}, \pi_D)$
2. Let $\{\hat{s}_i, \pi_i\} := \text{PVTSS.Recover}(\text{pp}_{\text{PVTSS}}, c_i, \text{pk}^{C_i}, \text{sk}^{C_i})$ // recover the decrypted share $\hat{s}_i$ together with proof π_i of valid decryption
3. Store (F_ϕ, d, $\hat{s}_i$, π_i)

Encode$_U$(pp, cred, attrs, S, ω) :

1. Request encoding information e^U form P and let EI := Ec$(e^U,$ attrs)
2. Let $\pi_\mathcal{R} := \text{G16.Prove}(\text{crs}_\mathcal{R}, (\text{cred}, \text{EI}), (\omega, e^U))$
3. Send (EI, cred, $\pi_\mathcal{R}$, S) to each party $C_i \in C$

Authenticate$_C$(pp, F_ϕ, cred, EI, EP, S, d, MT, $\gamma, \theta, \hat{s}_i, \pi_i$) :

1. Check $\text{G16.Verify}(\text{crs}_\mathcal{R}, \pi_\mathcal{R}, (\text{cred}, \text{EI}))\wedge\text{MT.Verify}(\gamma, \text{pk}^U, \text{cred}, \theta)$
2. Let encoded output EO := Ev(F_ϕ, (EI, EP))
3. Check the final output FO := De(EO, d)
4. If check pass and over T_1, send (S, $c_i, \hat{s}_i, \pi_i$) to U

AccessService$_U$(pp, pp_{PVTSS}, S, $\{c_i\}$, $\{\hat{s}_i\}$, $\{\pi_i\}$) :

1. Check $\text{PVTSS.Verify2}(\text{pp}_{\text{PVTSS}}, c_i, \hat{s}_i, \pi_i)$
2. Add the valid $\hat{s}_i$ to a set $\mathcal{S}$
3. If $|\mathcal{S}| > t$ and T_2 has not elapsed: let $s := \text{PVTSS.Pool}(\text{pp}_{\text{PVTSS}}, \mathcal{S}, T_2)$ and Result := S(s)

Revoke$_A$(pp, MT, pk^U) :

1. Let MT′ := MT.Remove(pk^U)
2. Return MT′

项目实现

1. **如何实现用户的身份隐私。**

   我们使用一种可监管隐私身份管理方案作为访问控制系统的前端，用于可监管的匿名凭证的生成。

   这种抗女巫攻击的可监管隐私身份管理方法，支持全面的监管治理机制（可追溯、可审计、可撤销）和细粒度的抗女巫攻击，同时提供强大的隐私保护能力，并具有去中心化（所有身份管理操作均由一个包含多个不同实体的委员会完成）、支持大规模假名追踪、高效率（大部分追踪计算都可以在单个节点上本地执行）、支持选择性链接（允许用户在不泄露身份相关信息的情况下选择性地披露其假名的链接性）等多个优秀关键特性。

2. **如何实现访问控制策略的隐私。**

   为了实现策略和数据保密性，我们利用混淆电路方案和零知识证明。服务提供者将混淆版本的政策发送给访问控制者（步骤4），而不是公开发布政策的明文版本（步骤3）。用户将其编码后的属性及其有效性证明发送给访问控制者，而不是证明其私人属性符合公开政策（步骤5）。

   在系统中，我们通过在编码阶段巧妙地修改混淆方案以包含零知识证明来实现策略保密性。在SetPolicy函数中，服务提供者将安全参数1λ和布尔电路$f\phi$作为混淆算法Gb的输入。该算法输出混淆电路Fϕ，即ϕ的加密表示，编码信息e，使得输入attrs到ϕ的函数可以与Fϕ兼容地编码，以及解码信息d，用于稍后解释混淆电路的输出。然后，服务提供者将Fϕ和d发送给访问控制者，并将e分成eU和eP。服务提供者使用eP对策略参数$param\phi$进行编码为EP := Ec(eP, $param\phi$)，并将eU留给用户编码其属性attrs。然而，如果对手同时获得eU和编码输入EI := Ec(eU , attrs)，他可以计算出用户的attrs。为了解决这个问题，服务提供者可以为相同的布尔电路预生成多个实例的混淆电路，每个实例仅用于一次。

   在Encode函数中，用户利用Groth16方案证明attrs的有效性和从服务提供者接收到的编码输入EI的正确性。使用编码函数，用户的attrs被编码成由服务提供者通过盲传输（OT）选择的相应密钥，而不会泄露attrs和eU。系统的设置提供了另一个限制，即用户应该使用ZKP向访问控制者证明attrs的有效性。为了实现这两个任务，用户需要使用ZKP证明OT过程（在用户和服务提供者之间）的正确性以访问控制者。然而，对于用户来说，在不知道服务提供者的证人eU的情况下构建ZKP是不切实际的。牺牲一点隐私，我们让服务提供者正常传输编码信息eU给用户。这样，用户可以使用公共输入cred、EI和证人attrs、eU生成证明，证明cred = Com(attrs) ∧ EI = Ec(eU , attrs)。或者，我们也可以使用安全多方计算（MPC）作为黑盒来解决这个问题。具体来说，公共输入包括用户的凭证cred，即attrs的承诺。私有输入包括服务提供者的编码信息eU和用户的属性attrs。只有访问控制者接收MPC的输出Ec(eU , attrs)。尽管这种方法实现了更好的隐私，但需要所有各方在线并承担高计算和通信开销。我们建议在未来的工作中探索这一点。

3. **如何实现访问控制数据的安全和隐私。**

   服务提供者可以使用对称密钥加密数据，并将这些加密数据存储在诸如IPFS之类的去中心化存储服务中[5]。认证后，符合政策的用户能够重建对称密钥并获取明文数据。

4. **如何实现公平的访问控制。**

   为了实现公平访问的目标，我们在系统中采用了公开可验证的定时秘密共享协议。在秘密共享阶段（步骤4）中加入时间原语，密码学上保证了在特定时间间隔内秘密重构过程的正确性（步骤6）。下限时间$T1$保证了在$T1$之前没有计算能力有限的对手能够在访问控制者之间发生任何潜在腐败的情况下得知秘密。此外，上限时间$T2$保证了在$T2$之前秘密重构的正确性。

   具体来看，我们在SetPolicy、VerifyShare和AccessService函数中集成了PVTSS方案，算法的功能如下。在SetPolicy函数中，服务提供者指定访问策略ϕ，并将其与随机生成的秘密结合，使所有符合策略的用户都能获取该秘密以访问服务。随后，提供者将秘密分成份额，用公钥加密后，通过时间锁定协议（TLP）锁定，并分发给访问控制者。接收到份额后，控制者使用Reed-Solomon码和DLEQ证明通过VerifyShare函数验证份额，然后开始解决谜题。一旦达到$T1$，他们用有效解密的SLP解密份额，并将其转发给符合策略的用户。在$T2$到期前，这些用户必须验证份额的正确性，并通过AccessService函数重构原始秘密以访问服务。

1.2 论文工作

1.2.1 Heimdall: Decentralized Access Control Scheme Enabling Fair Access and Policy Confidentiality.

分布式访问控制的概念:

- 分布式访问控制（DAC）是在没有单一控制实体的情况下管理访问权限。
- 基于区块链的DAC利用区块链的分散和透明特性，对个人数据进行可验证的控制。

现有方法的局限:

- 现有的方法主要关注智能合约的访问控制，未能实现公平访问和策略机密性。

Heimdall方案的提出:

- 提出了一种新的基于区块链的DAC方案，名为Heimdall，实现了公平访问和策略机密性，并提供面向服务的访问控制。
- Heimdall利用公开可验证时间秘密共享（PVTSS）来规范秘密共享和重构的过程，解决公平访问问题。
- 采用混淆电路方案和零知识证明来实现策略机密性，同时保证用户身份的机密性。

实现与功能:

- Heimdall使用功能加密、可验证加密和基于令牌的身份验证分别实现计算、验证和授权服务。
- 实现了Heimdall的原型，并展示了其实验结果，证明其具有实际性能。

未来研究方向:

- 借鉴匿名凭证方案中的问责机制，将问责功能引入Heimdall。计划利用门限公钥加密，使访问控制者在怀疑服务提供者存在恶意行为时能够重构策略。
- 改进属性编码的效率，利用分布式零知识证明，减少复杂混淆电路的证明时间。

1.2.2 CRYPTCODER: An Automatic Code Generator for Cryptographic Tasks in Ethereum Smart Contracts.

问题背景：以太坊为智能合约提供了一系列系统级别的加密API，以便进行各种加密操作。然而，由于缺乏加密领域专业知识，开发者在使用这些低级API时常常遇到难题，从而产生不安全的代码。

CRYPTCODER的介绍：为解决这个问题，研究者们引入了CRYPTCODER，一个自动代码生成器，用于桥接低级加密API和高级加密任务之间的差距。开发者可以使用CRYPTLANG轻松且安全地实现签名和承诺等加密任务，并利用CRYPTCODER将其自动转换成Solidity代码。

CRYPTLANG的特性：CRYPTLANG是一个基于任务的语言，支持三种以太坊中常用的加密类别：承诺、签名和摘要。它允许开发者专注于加密任务的高级规范，定义所使用的加密方案。CRYPTCODER能够自动将这些高级规范转换为Solidity实现。

CRYPTCODER的组件：包括解析器（用于解析输入的CRYPTLANG代码）、辅助模板（提供必要的代码输入以保证输出代码的功能性）以及集成器（将抽象语法树与辅助模板合并，自动生成Solidity代码）。

功能评估：对CRYPTCODER进行的评估显示了其在生成Solidity代码方面的功能性，以及相对于参考代码仅增加了4%的平均气体成本的可接受开销。

结论和未来工作：论文指出，CRYPTCODER为开发者有效处理高级加密任务提供了帮助，虽然增加了一定的气体开销，但仍是可接受的。未来的工作将包括扩展CRYPTCODER支持的加密代码类别范围，并增强其包括自动生成链下JavaScript代码的能力。

1.2.3 DIDAPPER: Practical and Auditable On-Chain Identity Service for Decentralized Applications.

现有DID系统的局限性：许多现有的DID系统缺乏完整的可审计性和与去中心化应用的互操作性，这限制了它们的实际应用。

DIDAPPER的解决方案：为了解决这些限制，DIDAPPER提出了一个实用的链上身份服务。它利用群签名为链上用户和去中心化应用提供匿名且可追踪的凭证。

DIDAPPER的目标：系统旨在确保匿名性、可审计性和服务化。重点在于在保障用户隐私的同时确保符合监管要求的可追踪性，并且高效灵活地服务广泛的去中心化应用。

系统参与者：论文概述了DIDAPPER系统中不同类型的参与者，包括DIDAPPER服务、群管理者、去中心化应用（Dapp）和用户。

工作流程和协议：详细讨论了诸如群创建、认证、密钥分发、凭证使用与验证以及凭证追踪等详细的工作流程过程。

性能和功能性评估：论文评估了DIDAPPER在以太坊平台上的性能，强调了链上验证的效率以及为实现匿名性和可审计性所引入的开销的可接受性。

1.2.4 A Sharding Blockchain-based UAV System for Search and Rescue Missions.

区块链与无人机的结合：论文讨论了将区块链应用于无人机（UAV）搜索与救援（SAR）任务中的优势。特别强调了区块链在提高系统安全性和去中心化决策方面的作用。

区块链可扩展性的挑战：尽管区块链提供了多种优势，但其在可扩展性方面存在限制。随着网络参与者数量的增加，区块链的吞吐量可能会快速下降。

分片技术：论文提出分片技术作为解决区块链可扩展性问题的方案。通过将区块链网络分成独立运行的小分区（分片），使得系统能够支持大量的搜索与救援无人机。

动态分片机制：除了分片带来的可扩展性优势外，系统还通过动态创建可配置的、专门用于任务的分片来提高适应性，并通过支持不同分片间智能合约的调用来提高互操作性。

系统实现与评估：论文还描述了该系统的实现原型，包括对提高适应性和互操作性的分析，并进行了性能评估。结果表明，该系统能够实现目标，克服了基于区块链的无人机系统在搜索与救援场景中的弱点。

2 读博规划

接下来将按照时间顺序依次介绍博三、博四和博五的项目和论文规划。

2.1 博三

2.1.1 非开源联盟链基础平台 （访问控制任务）

基于上述访问控制系统进行具体的代码实现。

2.1.2 非开源联盟链基础平台 （数据密态计算）

我负责课题3《面向区块链的多层次多维度安全和隐私保护体系》中的密态数据计算与分析。

背景：围绕联盟链在国产密码应用、高隐私保护等方面的短板，研究密钥管理、身份认证、数据加密等基于国产密码的链内多维度安全保护技术方案，支撑国家核心领域应用的高安全和隐私保护要求。

任务：研究基于秘密分享和零知识证明的密态数据计算技术，实现链上数据的密态发布，在保护用户数据隐私前提下，对数据进行分析、执行，提升密态内容的计算和带宽效率并充分发挥数据的价值优势。

我准备设计一个**高性能，抗女巫攻击，抵抗恶意用户恶意注入的隐私聚合系统**，这个系统可以在不暴露任何单个设备数据的情况下计算总体统计数据。目前已经有如苹果、谷歌和Mozilla等公司部署了隐私聚合系统。

项目实现：

1. **如何在不暴露任何单个设备数据的情况下计算总体统计数据 （如何实现可用不可得）。**

   使用秘密共享，多方计算和零知识证明等密码学技术来实现。

   1. **数据分割和秘密共享**

   多方计算（MPC）是一种加密技术，允许多个参与者在不泄露各自私有数据的情况下共同计算一个函数。客户端使用秘密共享技术将数据分割成若干份，并将这些份额发送到不同的服务器。

   具体过程如下：

   数据分割：

   - 客户端将其私有数据 x 分割成 n 份，每份数据 xi 被发送到不同的服务器。
   - 这些分割的数据满足： $x=x1+x2+\cdots+xn$
   - 这种分割方式确保每个服务器只持有数据的一部分，无法单独还原出原始数据。

   秘密共享方案：

   - **加法秘密共享**：一种常见的秘密共享方法。客户端生成随机数 r1,r2,...,rn−1并计算最后一份 rn 使得所有份额之和等于原始数据 x 。

     具体为： $rn=x-(r1+r2+\cdots+rn-1)$

   - 将r1,r2,...,rn 分别发送到不同的服务器。

   2. **零知识证明**

   为了确保数据分割的正确性和数据的有效性，客户端需要提供零知识证明。零知识证明是一种加密证明方法，使证明者能够在不泄露实际数据的情况下，证明某个陈述的真实性。

   具体过程如下：

   生成证明：

   - 客户端生成一个零知识证明，证明其数据分割是正确的，即 $x=x1+x2+\cdots+xn$。
   - 这个证明是基于数据分割的过程，客户端生成的每个份额都能验证整个数据的完整性和有效性。

   验证证明：

   - 服务器在接收到数据份额后，通过零知识证明验证数据的有效性。
   - 验证过程确保每个服务器收到的数据份额是合法的，不含恶意修改或错误数据。

   3. **数据聚合**

   在所有服务器验证数据的有效性后，开始进行数据聚合。

2. **如何提升密态计算的效率。**

   使用静默可验证证明，模拟多方计算，和证明批量验证来实现。

   初始化阶段：

   - 每个客户端生成其私有数据 x 的秘密共享份额 x1,x2,...,xn，并分别发送给不同的服务器。
   - 客户端生成一个静默可验证证明，证明其数据分割是正确的，并将证明的每一部分发送给对应的服务器。

   证明生成（Gen）：

- 证明生成过程包括两个部分：公共部分和私有部分。
- **公共部分（π_pub）**：包含所有验证者共享的信息。
- **私有部分（π_i）**：每个验证者持有的独立信息。
- 具体过程如下：
  - 客户端模拟所有参与者（证明者和验证者）的行为，生成一个模拟的协议执行转录（transcript），记录所有消息交换。
  - 客户端将这些转录发送给每个验证者。

证明验证（Eval 和 Ver）：

- 每个验证者在本地验证接收到的证明。
- **Eval**：验证者根据自己的私有信息和接收到的公共信息，计算出验证标签（verification tag）。
- **Ver**：验证者交换验证标签，并检查这些标签的线性组合是否为零。若是，则证明通过验证；否则，证明失败。

批量验证：

- 验证者可以批量验证多个证明，仅需交换一次验证标签的线性组合。
- 通过这种方式，验证者可以在极低的通信成本下验证大量客户端的提交。

3. **如何减少服务器的存储空间。**

利用小空间素描（Sketching），用于在存储空间有限的情况下近似计算统计数据。

1. **线性素描（Linear Sketches)**

线性素描通过对数据进行线性变换，将高维数据投影到低维空间。常用的线性素描方法包括Count-Min Sketch和Count Sketch。

- **Count-Min Sketch**：
  - 维护一个二维数组，每一行对应一个独立的哈希函数。
  - 每个元素通过多个哈希函数映射到数组中的位置，并在这些位置上进行计数。
  - 频率查询时，返回所有哈希函数映射位置中的最小值作为估计值。
- **Count Sketch**：
  - 与Count-Min Sketch类似，但在计数时会根据哈希函数的符号进行加减操作。
  - 频率查询时，返回所有哈希函数映射位置中的中位数作为估计值。

2. **高效近似算法**

小空间素描使用高效的近似算法来处理数据，这些算法能够在有限的时间和空间内，提供接近真实值的统计结果。

- **随机哈希函数**：通过使用随机哈希函数，可以将数据映射到较小的空间，从而降低存储需求。
- **线性变换**：通过线性变换，可以在低维空间中进行统计计算，减少了计算复杂度。

4. **如何实现抗女巫攻击特性。**

为了防止女巫攻击，系统首先需要一个强大的注册和认证机制，以确保每个客户端的唯一性和真实性。

- **唯一身份标识**：系统要求每个客户端在注册时提供唯一的身份标识，可以通过政府颁发的证件或其他可靠的身份验证方法来确保每个客户端的身份唯一且不可伪造。
- **认证协议**：使用强大的认证协议，如OAuth、SAML等，确保每个客户端在注册和登录过程中都能被验证其身份的真实性。

具体可以结合王师兄的Hades实现：

应用程序初始化抗女巫攻击实例，用户使用假名提供访问令牌和零知识证明。

详细步骤：

- 应用程序创建抗女巫攻击实例，设置实例ID ζ 和 nonce 限制。
- 用户生成访问令牌 φ=Hash(sku||ζ||nonce)。
- 用户提供审计串 ψa 和 Pedersen 承诺 At 的零知识证明。
- 应用程序验证访问令牌和证明的正确性，检查令牌是否有重复。

5. 如何抵御恶意用户恶意注入数据以干扰总体统计数据。

2.1.3 2024.12 S&P Private Analytics

创新点：

1. 目前的隐私聚合系统只支持简单的计算（求和、平均、方差、标准差、最小值/最大值、频率计数、近似频率、与、或、线性回归），对于复杂的计算只能得到近似的结果。我打算加入混淆电路，实现更复杂的计算。
2. 对于批量验证零知识证明，难以抵御恶意用户故意提交错误证明的攻击。一旦发现错误证明，系统需要有效处理以防止恶意用户继续攻击。我打算使用Merkle树，服务器可以高效验证大批量数据的完整性，并识别出错误证明。可以结合Hades的可追溯特性，追溯恶意用户信息，限制其进一步操作或采取其他惩罚措施。

2.2 博四

2.2.1 基于区块链的数据要素市场关键技术与示范应用

我负责课题1《基于区块链的数据要素流通交易模型与数据要素市场技术体系研究 》中的隐私链上数据要素交易。

背景：数据要素具有异构多态、权属复杂的特性和点对点的典型交易模式，集中式的大数据交易所难以满足持续发展的数据要素流通需求，区块链去中心化、可编程的特性展现了构建数据要素交易市场的应用潜力，业界尚未形成基于区块链的数据要素交易流通的基础理论体系和总体技术架构。

任务：针对区块链体系中缺乏底层规则支持的现状与复杂数据要素链上流通技术的要求，建立包括链上链下协同的数据要素交易规则、支持隐私保护的链上数据要素交易规则、跨链数据要素交易规则等在内的、可扩展的区块链数据要素核心交易规则。

我准备设计一个**高性能，满足交易原子性的数据要素隐私交易系统。**

项目实现：

1. **如何在保证数据要素隐私的前提下进行公平交易。**

   我们使用commit-and-prove non-interactive zero-knowledge (CP-NIZK) 来实现隐私数据的公平交易。

   CP-NIZK的构造基于承诺方案和零知识证明系统。以下是CP-NIZK的一般构造步骤：

   - **承诺数据**：卖家对数据x生成承诺cx = Commit(pp, x)。
   - **生成验证证明**：卖家生成证明π，证明x满足条件φ，即π证明了cx包含的数据x满足φ(x) = 1。
   - **生成交付证明**：卖家加密数据x生成z = Enck(x)，并生成证明πz，证明z是通过k加密cx包含的数据x得到的，同时h = H(k)。
   - **验证证明与数据交付**：买家验证π和πz的有效性，确保数据的有效性和一致性。买家在验证交付证明后，完成交易并支付款项。

   协议在区块链上运行，利用智能合约作为公正的仲裁者。智能合约负责验证零知识证明和交付证明，确保交易双方的公平性和数据隐私。通过智能合约的自动执行，可以有效避免人为干预和欺诈行为。

2. **如何实现交易的原子性。**

   ZKCPlus将交易分为两个主要阶段：验证阶段和交付阶段。

- **验证阶段：**在验证阶段，卖家生成一个零知识证明（Proof π），证明数据满足买家的条件。买家在验证该证明后，生成一个包含支付信息和哈希锁（hash lock）的交易（tx），并将其发送到区块链上的智能合约。这个交易表示只有揭示哈希锁的预像（preimage）后才能支付。
- **交付阶段：**在交付阶段，卖家加密数据x生成密文z = Enck(x)，并提供一个交付证明（Proof πz），证明加密的数据与之前承诺的数据一致。卖家将这个证明和加密数据发送给买家，随后揭示哈希锁的预像k。智能合约验证πz和揭示的预像k是否匹配，如果匹配，则执行支付，将款项从买家转移到卖家。买家使用密钥k解密密文z，得到原始数据x。

3. **如何提升系统的性能，以交换更大规模的数据要素。**
    1. **改进加密和解密过程**
        - 使用高效的加密算法：选择高效的加密算法，如MiMC-p/p分组密码在CTR（计数器）模式下进行数据加密，可以有效提升加密和解密的速度。
        - 加密过程并行化：将加密过程并行化处理，可以显著提高大规模数据的处理效率。例如，使用CTR模式进行加密时，可以对每个数据块独立加密，从而实现并行处理。
    2. **批处理技术**
        - 批量交易处理：将多个交易打包成一个批次进行处理，可以减少交易的整体开销和延迟。例如，可以将多个小规模交易合并成一个大规模交易进行处理和验证。
        - 批量证明生成：通过批处理技术，可以将多个零知识证明打包生成一个批量证明，从而减少证明生成和验证的开销。
4. **如何给出具体实际的示范应用。**

    应用示范1：交易训练好的CNN模型

    - **场景描述：**在机器学习即服务（MLaaS）中，一个计算能力有限的客户端可以将训练机器学习模型的重任务委托给拥有充足计算资源的不可信服务器。服务器在给定的模型架构和超参数下进行训练，并将训练好的参数（如权重和偏置）作为数字商品出售。客户端只有在模型表现出足够高的准确率时才愿意支付。

    应用示范2：SQL查询支付

    - **场景描述：**在云数据库系统中，客户端将数据库外包给不可信服务器，然后服务器响应客户端的查询。为了保证公平，客户端只为正确的查询结果付费。

2.2.2 2025 CCS Zero-knowledge contingent payments (ZKCP)

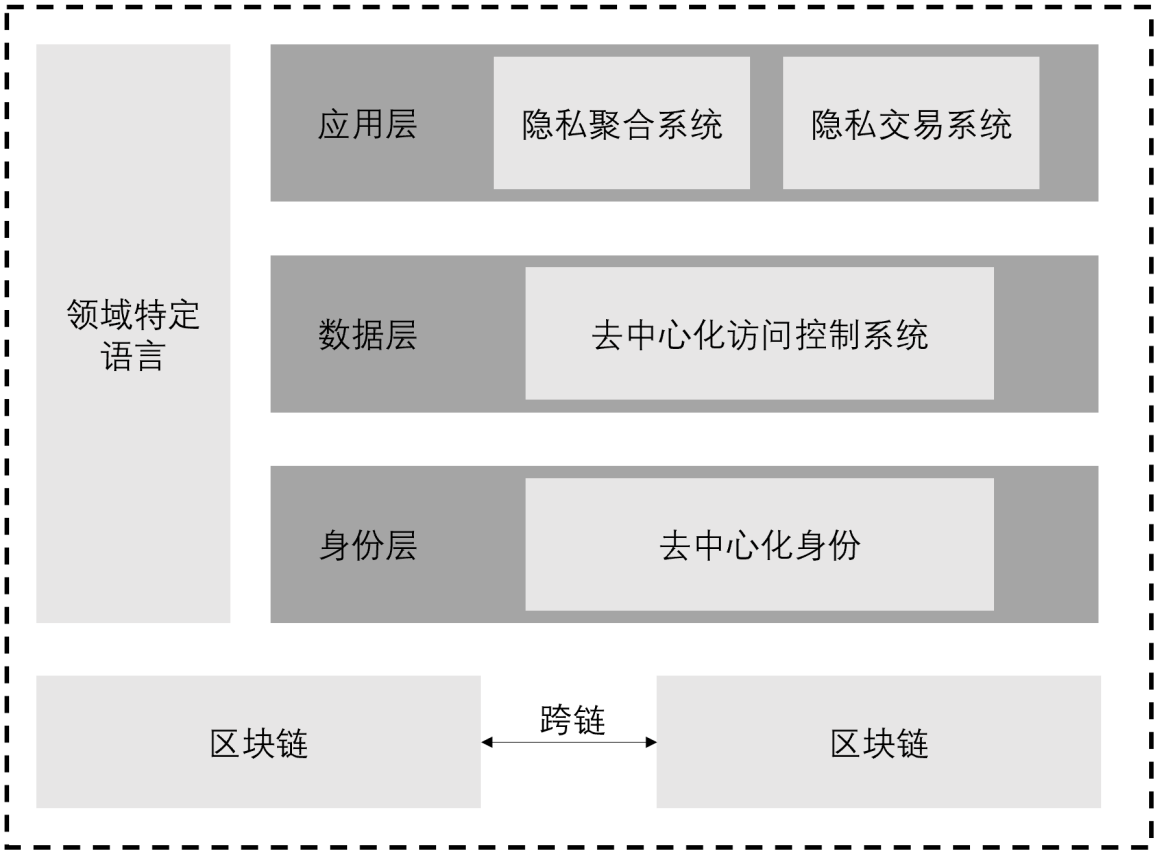
创新点：

1. 目前的工作主要集中在提高证明者生成零知识证明的效率，却忽略了验证者验证零知识证明的时间与gas的开销。我希望通过批处理技术，可以将多个零知识证明打包生成一个批量证明，从而减少证明生成和验证的开销。
2. 目前的工作只支持简单的谓词，并不支持复杂谓词证明的高效聚合。我希望利用Groth16关联证明，证明多个 Groth16 证明共享一些隐藏的公共输入，在保证隐私的同时，能够验证这些证明的互相关联性。

2.3 博五

2.3.1 毕业论文：利用密码学技术实现三层数据主权的系统架构

数据主权



1. **区块链**

   区块链是整个系统的最底层，其去中心化，不可篡改，可编程特性为数据主权的实现提供了坚实的基础。

2. **身份层**

   去中心化身份系统旨在去除集中化实体，确保用户无需依赖第三方即可管理和验证其数字身份。该系统主要由去中心化标识符（DID）和可验证凭证（VC）构成。具体可以使用我设计的**Didapper**身份管理系统或者兼容王师兄设计的Hades身份系统，在保证用户身份隐私的同时实现身份的可追溯、可审计、可撤销、抗女巫攻击等性质。

3. **数据层**

   去中心化访问控制系统旨在去除用户对集中化访问控制器的依赖，使用户能够验证地控制谁可以访问其个人数据。具体可以使用我设计的**Heimdall**去中心化访问控制系统，在保证数据机密性，匿名性的同时还能实现策略机密性和公平访问。

4. **应用层**

   本系统具体实现了两个应用分别是**隐私聚合系统**和**隐私交易系统。**

   - 隐私聚合系统可以在不暴露任何单个设备数据的情况下计算总体统计数据。本系统通过混淆电路实现更复杂的计算，通过Merkle树实现高效验证大批量数据的完整性，并识别出错误证明。可以结合Hades的可追溯特性，追溯恶意用户信息，限制其进一步操作或采取其他惩罚措施。
   - 隐私交易系统可以保证数据隐私的前提下进行公平交易。本系统通过批处理技术，可以将多个零知识证明打包生成一个批量证明，从而减少证明生成和验证的开销。系统还能利用Groth16关联证明来支持复杂谓词证明的高效聚合。

5. **领域特定语言**

   通过设计领域特定语言实现访问控制策略和隐私聚合函数的高效编写。