BitXHub 白皮书 V2.0

区块链跨链技术平台

趣链科技 HYPERCHAIN | BitXHub

2022 年 7 月

编写成员

研究撰写：

汪小益、李瑞阳、李若欣、孙昊、夏立伟、朱沛文、周蓉、徐才巢、江哲

排版设计：

龚巧琳、张驰

项目地址

BitXHub 跨链技术平台：https://www.bitxhub.cn/

中继链：https://github.com/meshplus/bitxhub

跨链网关：https://github.com/meshplus/pier

图目录

表目录

目录

第一章 概述

1.1 背景

当前的区块链应用和底层技术平台呈现出百花齐放的状态，但主流区块链应用中的每条链大多仍是一个独立的、垂直的封闭体系。在业务形式日益复杂的商业应用场景下，链与链之间缺乏统一的互联互通机制，这极大限制了区块链上数字资产价值的流动性，跨链[1]需求由此而来。

跨链指的是通过连接相对独立的区块链系统，实现不同账本的可信互操作。跨链依据其交换内容的不同可以大体分为数字资产交换和信息交换。在数字资产交换方面，当前资产交换主要依靠中心化的交易所来完成，中心化的交换方式既不安全、规则也不透明，业界也出现了去中心化的资产交换方式如 Uniswap、Curve、SushiSwap 等，但是当前的去中心化资产交易多数只能实现同一个区块链上不同合约资产的交换，对跨链数字资产去中心化交换仍不完善。信息交换涉及链与链之间的数据同步和相应的跨链调用，实现更为复杂，目前各个区块链应用之间互通壁垒极高，无法有效地进行跨链信息共享。

另一方面，区块链技术在单链架构下本身存在着性能差、容量不足等问题。单链受限于去中心化、可扩展性和安全性的权衡，难以支撑高交易吞吐量低延迟的商业场景应用。此外，随着区块链运行时间的增长，其存储容量也将逐渐增长，且这种数据增长的速度甚至会超过单链存储介质的容量上限。通过跨链技术实现多链协作的多层多链体系架构是解决区块链性能瓶颈的可取之道。

1.2 产品定义

趣链科技从跨链的需求出发提出了一种通用的链间消息传输协议，并基于该协议实现了同时支持同构及异构区块链间跨链交易的跨链技术示范平台 BitXHub，允许异构链间的资产交换、数据共享及合约调用。BitXHub 平台由中继链、应用链以及跨链网关三种角色组成，并链原生集成 W3C 标准的 DID 数字身份，依据场景导向可灵活组织部署架构，具有通用跨链传输协议、异构交易验证引擎、多层级路由三大核心功能特性，保证跨链交易的安全性、灵活性与可靠性。

BitXHub 致力于构建一个高可扩展、强鲁棒性、易升级的区块链跨链示范平台，为去中心化应用提供通信枢纽，支撑链上可信数据资产高效流动，服务区块链业务安全治理，为区块链互联网的形成提供可靠的底层技术支撑。

第二章 整体架构

2.1 平台架构

为了支持异构区块链之间交易的可信验证和可靠传递，我们设计了一种类似 TCP/IP 的链间传输协议：IBTP（Inter-Blockchain Transfer Protocol）。BitXHub 平台是一种支持 IBTP 协议的跨链综合服务的参考实现。

BitXHub 平台是一种由中继链、跨链网关和应用链所构建的跨层级链间互操作服务平台，其主要组成部分包括：

- **中继链**（Relay-chain）：中继链主要用于数字身份管理、应用链管理、跨链交易的可信验证与可靠路由以及跨链 Dapp 治理，是一种实现 IBTP 协议的开放许可链；
- **跨链网关**（Pier）：跨链网关担任着区块链间收集和传播交易的角色，既可以支持应用链和中继链之间的消息交互，也可以支持中继链与中继链之间的消息交互；
- **应用链**（App-chain）：应用链负责具体的业务逻辑，分为：（1）同构应用链（支持 IBTP 协议的区块链），具有类似的区块结构和交易数据存储格式，并具有相同的共识算法[2]和加密机制等，如趣链联盟链平台。（2）异构应用链，是指不直接支持 IBTP 协议的区块链，例如 Fabric[3]、以太坊[4]等。

为了适用不同的业务应用场景，我们的跨链组件可以通过灵活组合形成不同的架构，包括：链对链架构、主从链架构、中继链架构。我们把平台的这种架构称为“积木架构”。

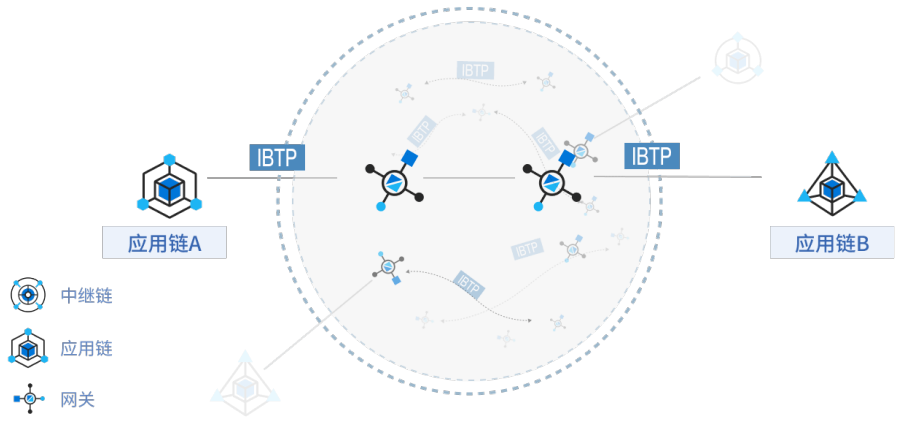

图 2-1 BitXHub 架构图（链对链架构）

链对链架构（如图 2-1）：在该种架构下，不同的链通过各自的跨链网关直接相连。对于跨链参与方有信任基础或者安全性要求不那么高的场景，采用链对链架构，可以大大降低跨链设施部署成本。

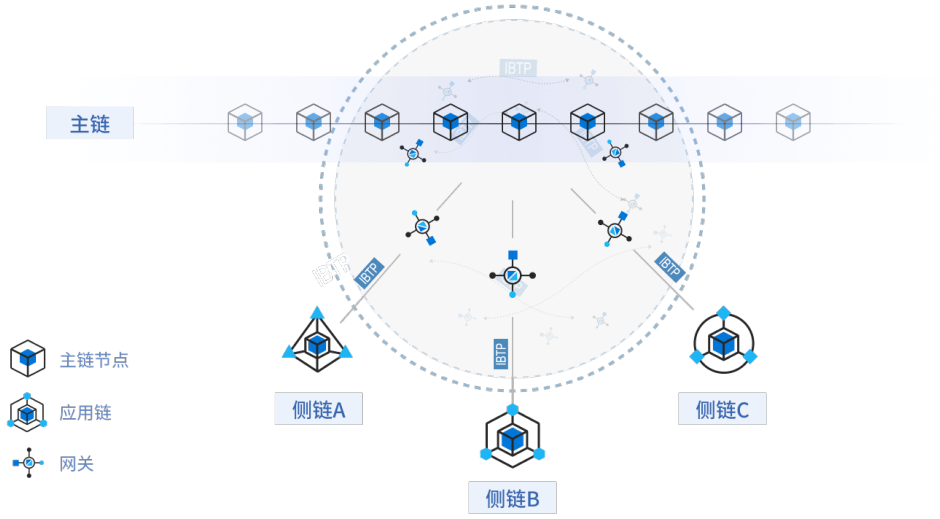

图 2-2 BitXHub 架构图（主从链架构）

主从链架构（如图 2-2）：此架构下存在一条主链，主链能够通过跨链的方式控制和管理从链上的部分功能和链上数据。该架构能够通过树状方式进行扩展，主链作为整个树的树根协调管理从链，比较合适的应用场景是存在权威中心或者希望通过多链提升业务性能。

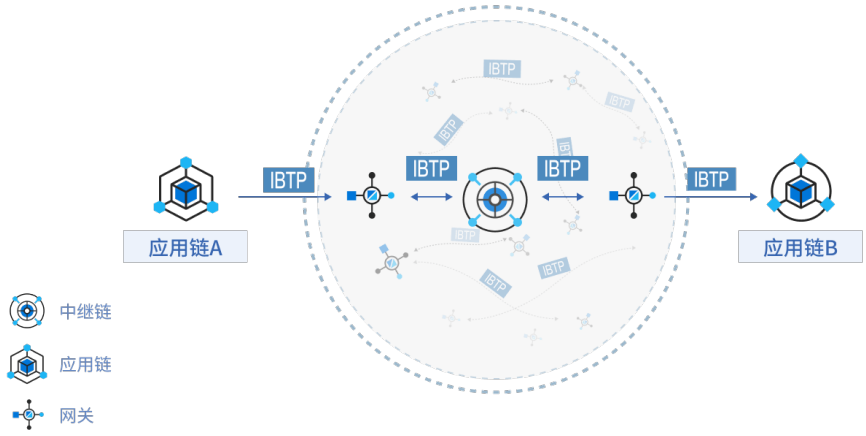

图 2-3 BitXHub 架构图（中继链架构）

中继链架构（如图 2-3）：该架构下需要有中继链来验证应用链发出的跨链交易，各个应用链以平等的身份加入到中继链中。这种情况下更适合地位平等的组织或者机构组成的跨链联盟，各个组织共同维护中继链，并让自己拥有的应用链接入到跨链系统中来。中继链架构下还能通过多层中继链互联形成中继链网络的形式进行横向扩展。

2.2 核心流程

2.2.1 单中继跨链架构

单中继链架构跨链交易处理核心流程如下：

1. 来源链发起跨链交易，记为 ct1；

2. 跨链交易 ct1 提交到来源链所关联的中继链；

3. 中继链验证跨链交易 ct1 是否可信，一是验证交易来源的可信，二是验证交易证明是否满足该应用链上对应的跨链验证规则；

   3.1 如果验证不通过，则执行步骤 4；

   3.2 如果通过，则执行步骤 5；

4. 非法交易，中继链返回失败的回执与相关错误信息到来源链，来源链携带正确的 proof 的跨链交易重新发送，或者应用链管理员对中继链发起验证规则更新，直到中继链验证到正确的交易；

   4.1 如果验证成功，执行步骤 5；

   4.2 如果验证失败，重新执行步骤 4；

5. 中继链判断 ct1 的目的链服务是否在其管理的应用链服务列表中，如果存在则执行步骤 6，否则执行步骤 7；

6. 提交 ct1 到目的链，目的链执行交易 ct1；

   6.1 如果执行成功，则执行步骤 10；

   6.2 如果执行失败，则执行步骤 11；

7. 中继链将交易 ct1 标记为无效交易，并通知来源链进行回滚；

8. 中继链提交无效交易到目的链，目的链进行回滚并返回执行失败的回执到中继链；

9. 中继链收到目的链失败的回执，将 ct1 标记为失败交易，进入步骤 12；

10. 中继链收到目的链执行成功的回执，将 ct1 标记为成功交易，进入步骤 12。

11. 中继链收到目的链执行失败的回执，将 ct1 标记为失败交易，并通知来源链回滚；

12. 交易结束。

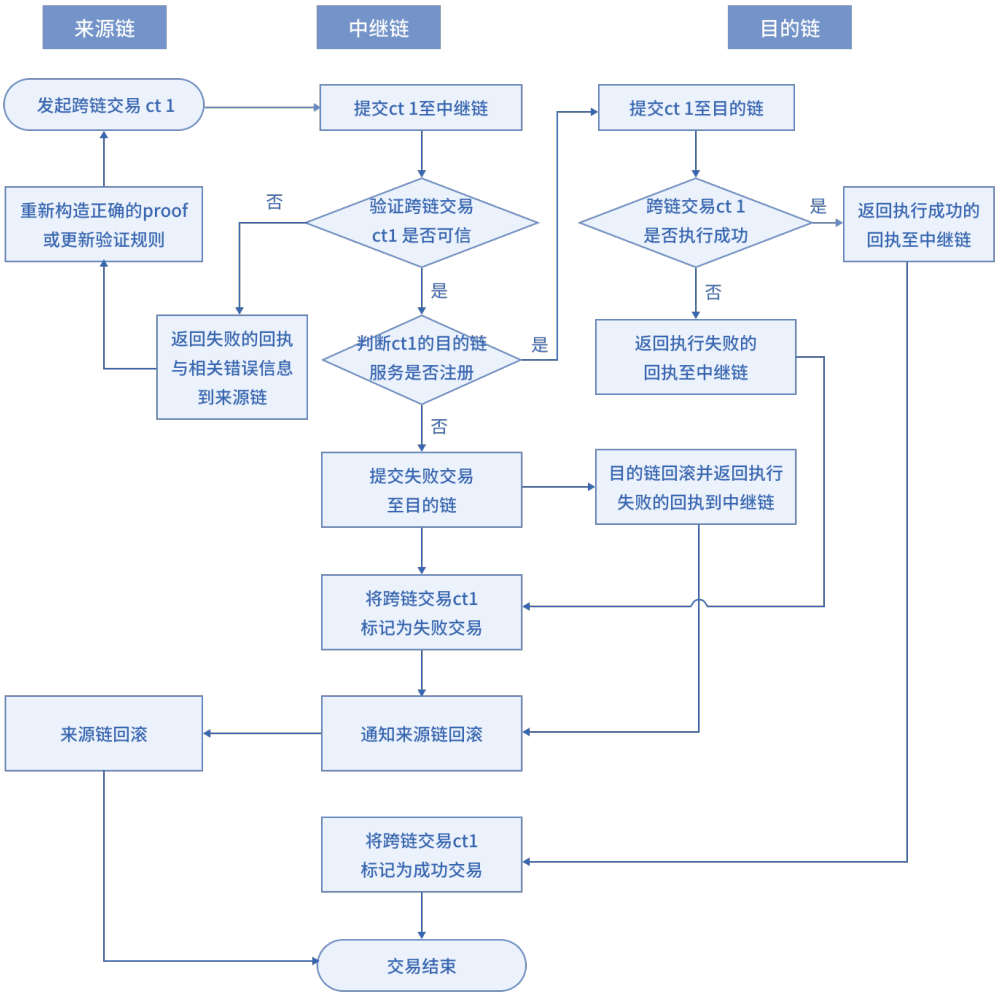

图 2-4 单中继链架构跨链交易处理核心流程

2.2.2 多中继跨链架构

由于单个中继链的处理能力有限，我们期望假设每个中继链支持不超过 64 个应用链的跨链管理，具体应用链的个数根据跨链业务的复杂度进行动态调整。因此在大的跨链生态中将会出现多个以中继链为中心的区块链互联网，那么跨链交易如何在整个跨链生态中进行流转呢？如图 2-5 所示为多中继链架构跨链交易处理核心流程：

1. 应用链 1 发起跨链交易，记为 ct1；

2. 跨链交易 ct1 被提交到应用链 1 所关联的中继链 A；

3. 中继链 A 验证跨链交易 ct1 是否可信，一是验证交易来源的可信，二是验证交易证明是否满足该应用链上对应的跨链验证规则；

3.1 如果验证不通过，则执行步骤 4；

3.2 如果通过，则执行步骤 5；

4. 非法交易，回滚，执行步骤 11；

5. 中继链 A 判断 ct1 的目的链服务是否在其管理的应用链服务列表中，如果存在则执行步骤 6，否则执行步骤 7；

6. 提交 ct1 到目的链，执行步骤 11；

7. 提交 ct1 到中继链 A 相关联的跨链网关 1；

8. 跨链网关 1 根据 ct1 的目的链地址，在跨链网关集群中通过分布式哈希表的方式进行查询，如果目的链所关联的中继链 B 的跨链网关 2 存在则执行步骤 9，否则执行步骤 4；

9. 跨链网关 1 将 ct1 发送给跨链网关 2，跨链网关 2 将其提交到目的链所关联的中继链 B；

10. 中继链 B 验证 ct1 是否通过前置中继链（即中继链 A）的验证背书，如果背书验证可信则执行步骤 6，否则执行步骤 4；

11. 结束交易。

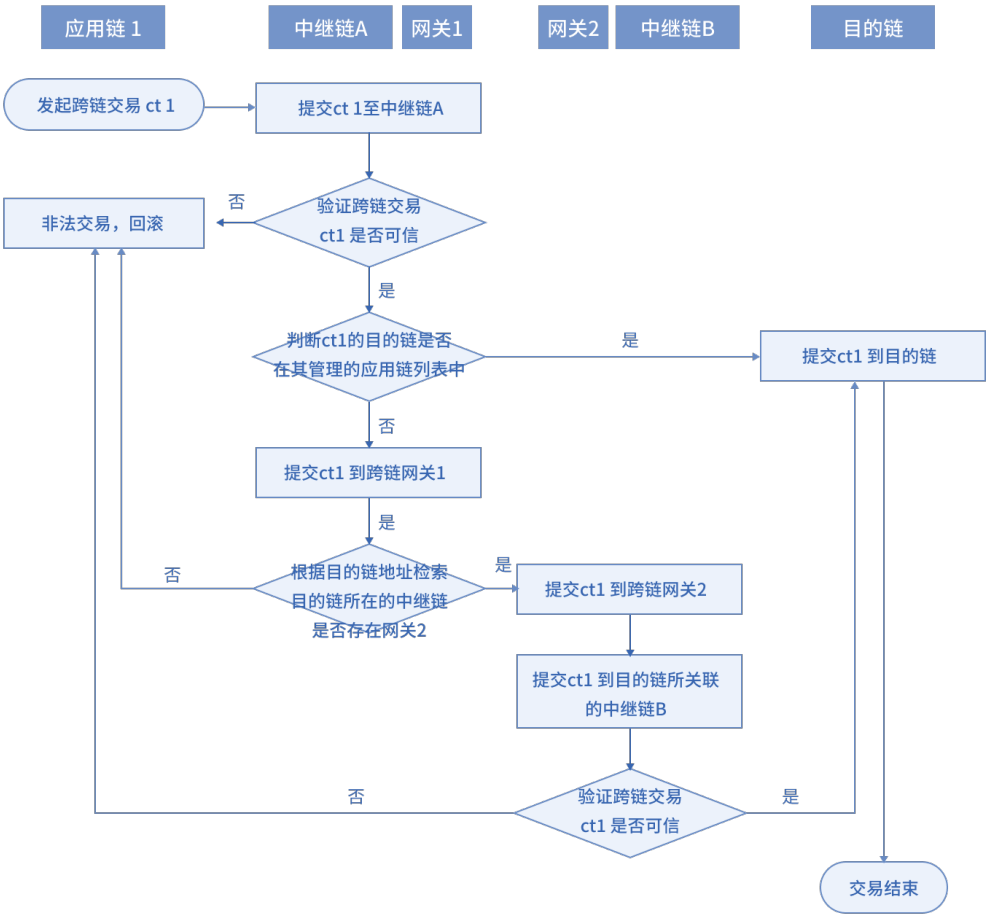

图 2-5 多中继链架构跨链交易处理核心流程

第三章 IBTP 协议

由于区块链只能维护自身状态，跨链应用场景中的区块链难以与外界进行沟通，为了中继链能够更方便地进行跨链消息的验证和路由，跨链网关能够更一致地进行跨链消息处理，BitXHub 设计了一种类似 TCP/IP 的 IBTP 通用链间消息传输协议。

跨链传输协议主要围绕 IBTP 数据结构展开，该数据结构在跨链交易之间的流转运行基于 IBTP 协议的几个关键特性：跨链服务、跨链消息证明以及信任树。

3.1 数据结构

IBTP 数据结构基于通用性和灵活性规范了一套跨链消息的主要数据字段，具体如表 3-1 所示：

表 3-1 IBTP 协议数据结构

参数	说明
From	来源服务 ID <BXH ID>:<AppChain ID>:<Service ID>
To	目的服务 ID <BXH ID>:<AppChain ID>:<Service ID>
Index	跨链交易索引
Type	IBTP 类型
Payload	跨链调用内容编码
Proof	跨链交易证明
TimeoutHeight	IBTP 在中继链上的超时块高
Group	一对多场景的跨链事务信息
Version	协议版本号
Extra	自定义字段

From 和 To 字段分别代表来源服务和目的服务的 ID，其格式为<BXH ID>:<AppChain ID>:<Service ID>，其中：

- BXH ID：来源/目的服务所在应用链接入的中继链 ID，如果来源服务和目的服务所在链连接了同一条中继链，该部分可以省略；
- AppChain ID：来源/目的服务所在应用链 ID，应用链向中继链注册时由应用链管理员给出（即应用链管理员自定义），同一条中继链上各应用链 ID 唯一；

- Service ID：来源/目的链上服务 ID，服务向中继链注册时生成，一般为服务的合约地址，同一条应用链上服务 ID 唯一。

Index 是跨链交易的索引，每一对 From 和 To 维护一个严格递增的索引值，它是中继链按序执行跨链交易的依据。Type 字段标识 IBTP 类型，来源服务发到目的服务的跨链请求类型为 IBTP_INTERCHAIN，目的服务发回来源服务的跨链回执类型可能为 IBTP_RECEIPT_SUCCESS、IBTP_RECEIPT_FAILURE 或 IBTP_RECEIPT_ROLLBACK，具体由跨链交易的执行状态决定。

Payload 字段是跨链调用的内容编码，支持定向加密，根据应用链的业务需求确定。

Proof 字段存储了跨链交易证明。跨链消息证明用于验证每笔跨链交易的有效性和存在性，其内容因应用链链类型而异。对于部分应用链跨链消息证明文件较大的情况，为了保证 IBTP 结构精简，Proof 字段也可存储跨链证明信息的哈希。

TimeoutHeight 字段为 IBTP 在中继链上的超时块高，仅对跨链请求（IBTP_INTERCHAIN 类型）的类型有效，该字段主要用于事务超时回滚，详见 4.4 节事务管理机制相关内容。Group 字段包含了一对多场景中同一跨链事务中其他 IBTP 的信息，关于一对多事务场景详见 4.3 节事务管理机制。VerSion 为跨链协议的版本信息。最后，IBTP 数据结构提供了一个扩展字段 Extra，可根据应用链的业务需求进行自定义扩展。

3.2 关键特性

IBTP 跨链协议的关键特性主要包括跨链服务、跨链消息证明和信任树。跨链服务是跨链交易的基本单元，通过 IBTP 跨链协议，可以实现不同应用链上业务服务级别的互操作。为了进一步保证跨链消息的有效性，跨链协议 IBTP 采用跨链消息证明及信任树机制，更好地实现对跨链交易的合法性验证。

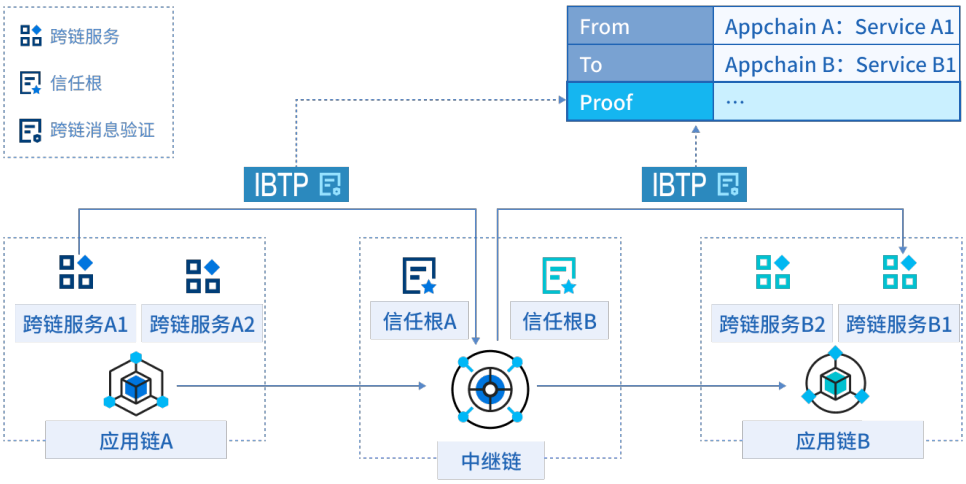

图 3-1 IBTP 跨链协议关键特性

3.2.1 跨链服务

跨链服务是指应用链要参与跨链的业务，该服务可以是来源链上发起跨链的服务，也可以是目的链上接受跨链调用的服务。跨链服务承载着不同应用链上的具体业务执行逻辑，跨链多方基于跨链服务的业务属性进行链间互操作协同。

1. 应用链注册

跨链系统具有审核准入机制，即参与跨链的多方需将应用链信息提交至中继链，并由中继管理员审核通过，该过程称为应用链注册。应用链注册需要提供的信息如表 3-2 所示：

表 3-2 应用链注册信息

字段	说明
AppChain ID	应用链 ID
Name	应用链名称
Type	应用链类型

TrustRoot	应用链信任根
Broker	应用链 broker 合约地址
MasterRuleAddr	应用链主验证规则地址
MasterRuleUrl	应用链主验证规则链接
AdminAddrs	应用链管理员地址列表
Desc	应用链描述

AppChain ID 为应用链在当前中继链网络中的唯一 ID，该 ID 对应 IBTP ID 中的第二部分；Name 为应用链在当前中继链网络中的唯一名称，支持中文，方便用户自定义使用；Type 为应用链类型。TrustRoot 为应用链信任根，不同共识算法的信任根类型不同，详见 3.2.3 节。Broker 为应用链上部署的 broker 合约地址。MasterRuleAddr 为验证该应用链跨链交易的主验证规则地址，可以是提前在中继链上部署好的合约地址，也可以是中继链为该类型应用链提供的内置验证规则地址；MasterRuleUrl 为验证该应用链跨链交易的主验证规则详情链接，如果 MasterRuleAddr 选择了内置验证规则地址，MasterRuleUrl 可为空。AdminAddrs 为应用链管理员列表地址，应用链可以有多个管理员地址，至少包含当前注册应用链的管理员地址。Desc 为应用链描述信息。

2. 服务注册

参与跨链的应用链业务必须将自己的业务信息提交到中继链，并由中继链管理员审核通过，该过程称为服务注册。服务注册需要提供的信息如表 3-3 所示：

表 3-3 服务注册信息

字段	说明
AppChain ID	应用链 ID
Service ID	服务 ID
Name	服务名称
Type	服务类型
Ordered	表示该服务是否需要按序调用
Permission	允许调用该服务的其他服务列表
Intro	服务功能简介
Details	服务调用详情，包含服务调用名称、参数等信息
PubKey	与该服务绑定的网关公钥，可选

AppChain ID 即为应用链注册时的 ID，Service ID 为服务在当前应用链上的唯一 ID，一般为应用链上的服务合约地址，对应 IBTP ID 中的第三部分；Name 为服务名称，支持中文，方便用户自定义使用。Type 字段为跨链服务类型，主要有以下几类：

- 合约调用类服务：通过目的链 Broker 合约进行调用，该类服务一般需要按序调用；
- 存证类服务：直接将来源链的数据发送到目的链指定地址，该类服务一般不需要按序调用；
- 数据迁移类服务：将来源链的交易转发到目的链，该类服务常用于同一机构两条链之间数据迁移，一般需要按序调用。

3. 服务属性

跨链服务具有以下几点属性：

- 唯一确定性：一个来自来源链服务的跨链消息对目的链服务有且仅有一次调用；
- 顺序性：一个定序的服务是指跨链消息必须按照发出顺序对目的服务进行调用，一个不定序的服务是指跨链消息无需按照发出顺序对目的服务进行调用；
- 许可性：跨链服务可以设置自己允许被哪些其他跨链服务调用。

3.2.2 跨链消息证明

应用链的异构性在跨链协议 IBTP 数据结构的差异主要体现在 Proof 字段。Proof 是跨链交易合法性证明的凭证，下面将从异构和同构两类应用链详细阐述 IBTP 结构的构建。

1. 同构应用链

下面以同构应用链为例简单说明 IBTP 的构造过程。其中，跨链交易组成的 Merkle Tree 结构[5]可以参见 4.1 节。

同构应用链所构建的 IBTP 结构如图 3-2 所示：

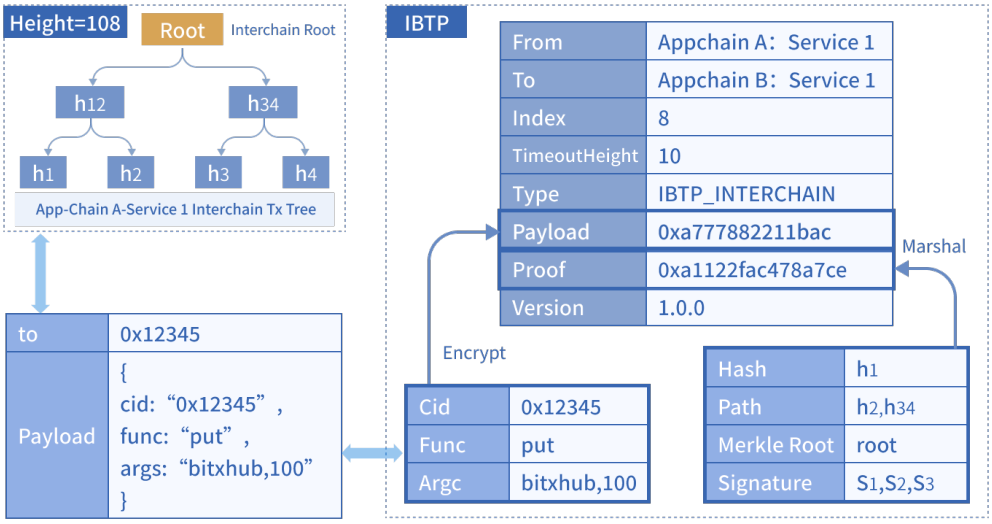

图 3-2 同构应用链构建的 IBTP 结构

以 108 号区块的第一个跨链交易为例，对于应用链 B 中地址为“0x12345”的合约，该交易调用合约方法“put”，参数为“bitxhub，100”。Proof 提供 Merkle 路径的信息，其中 Hash 是跨链交易相关内容的哈希，Path 是 SPV 路径哈希，MerkleRoot 是最终的根哈希，Signature 是对于根哈希的签名。

2. 异构应用链

异构应用链分为两种，一种是及时确认并且交易本身有验证节点作为背书的区块链，比如 Fabric 等；另一种是概率确认或者交易本身没有验证节点的区块链，比如比特币[6]、以太坊等。

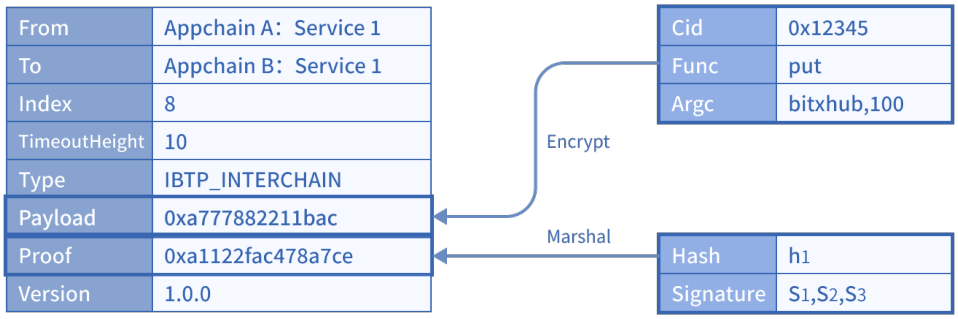

图 3-3 异构应用链构建的 IBTP 结构

第一种类型的异构应用链已满足 IBTP 协议的构造条件，因此跨链网关只需要调用应用链的 SDK 即可实现 IBTP 的构建。图 3-3 是以 Fabric 为应用链的 IBTP 结构，其中 Proof 字段包含的 Hash 字段是 Fabric 中 chaincode 执行结果的摘要，Signature 字段则是背书节点对 Hash 字段的签名数组，而对应的背书节点的证书信息[7]，即应用链信任根，已经提前存储到中继链中，详见 3.2.3 小节。

第二种类型以以太坊为例，其本身是概率性确认的区块链，且抛出的跨链事件未携带签名信息。因此跨链交易合法性证明需要由多个跨链网关协作构造，跨链网关的可靠性通过奖惩机制进行保证。

3.2.3 信任树

信任树是中继链上维护的关于应用链注册以来的所有信任根的更变信息。IBTP 协议要求，跨链交易场景中目的应用链需要验证跨链消息的有效性。根据 3.2.2 节内容可知，异构链跨链交易的验证需要结合 IBTP 的 Proof 信息和应用链的信任根信息，这个信任根信息就可以从该应用链的信任树中获取。

1. 信任树内容

信任树的具体内容由应用链共识算法类型决定。对于 PoW 共识的应用链，其信任树为应用链连续的区块头，信任根为某一个高度的区块头；对于 PoS 共识的应用链，其信任树为验证人信息的变化纪录，信任根则为某一时刻的验证人集合。

2. 信任树维护

应用链的初始信任根在应用链注册时存储到中继链上，后续信任根可能会发生变化，但跨链交易验证使用的信任根应当是交易发起时的应用链信任根，故 IBTP 协议需要中继链维护每条已注册的应用链信任树。

不同的应用链信任树通过各自的信任树智能合约来维护，即由跨链网关获取应用链信任树的变化信息并提交到中继链上，每一条已注册的应用链在接入的中继链上有唯一对应的信任树。

第四章 中继链

中继链主要用于数字身份管理、应用链管理、跨链交易的可信验证与可靠路由以及跨链 Dapp 治理，是一种实现 IBTP 协议的开放许可链，中继链与中继链间还可进一步实现跨链交易证明的跨层级信任传递[8]。

4.1 关键存储结构

中继链本身需要提供跨链交易的合法性证明, 所以其存储结构比传统区块链多了一个由跨链交易构成的 Merkle Tree。中继链的 Merkle Tree 构造如图 4-1 所示，每个区块中相同目的链的所有跨链交易组成一个新的 Merkle Tree，其中叶子节点是跨链交易的哈希，它们的 Merkle Root 和非跨链交易组成的 Merkle Root 再算出一个新的 Merkle Root。验证者会对最终的 Merkle Root 进行签名。

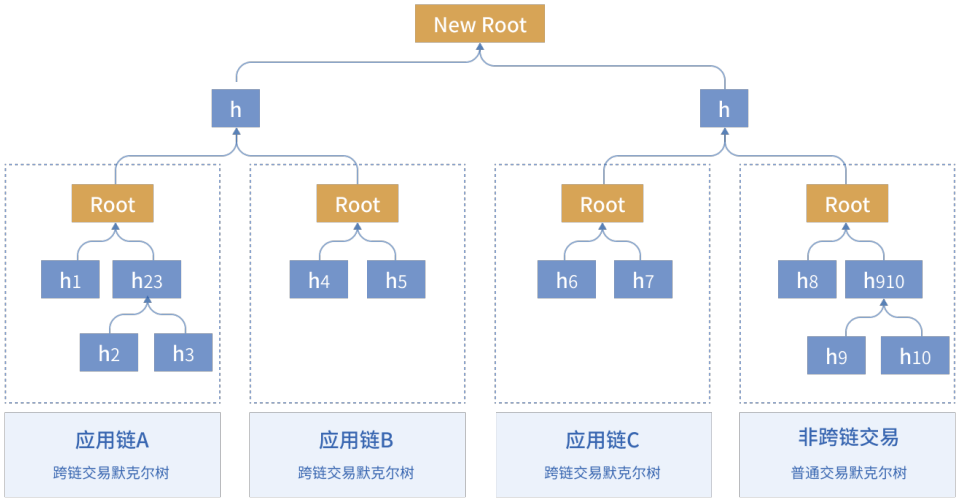

图 4-1 中继链存储结构示意图

4.2 验证引擎

我们为中继链设计了一种高效可插拔的验证引擎，用于支持应用链所提交跨链交易的可信验证，基于动态注入的验证规则对相应应用链提交的证明进行验证。在应用链接入之前，首先需要进行验证规则的编写和注册，验证规则由中继链审核后才能部署到验证引擎。

在跨链平台中，对于每一笔跨链交易，中继链需要对其进行校验，防止交易被伪造或篡改。验证引擎通过智能合约的方式管理多种验证规则，对不同区块链的交易进行合法性检验，并支持验证规则的在线升级和改造。

验证引擎的工作流程主要分为以下三个步骤：

- **协议解析：**验证引擎内部对跨链交易的解析。由于所有跨链交易都遵循 IBTP 协议，通过该步骤可以解析出交易的来源链信息和验证证明信息作为后续验证引擎的输入；
- **规则匹配：**验证引擎根据上述步骤解析出的来源链类型去匹配对应的验证规则脚本；
- **规则执行：**验证引擎的核心，主要通过 WASM 虚拟机[9]动态加载规则脚本，然后对跨链交易的 Proof 字段进行校验，从而确定交易的合法性。

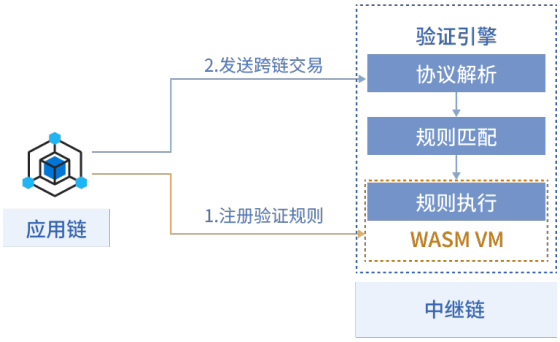

图 4-2 中继链交易验证流程示意图

综上所述，中继链的异构交易验证引擎具有诸多优势：

- 高效：通过 WASM 虚拟机保证了验证规则执行的高效性；
- 更新：验证规则依据不同区块链规则的变化快速、低成本的热更新；
- 全面：满足各类型区块链的验证体系；
- 便捷：应用链的业务人员可以直接管理验证脚本中的校验规则；
- 安全：对 WASM 虚拟机设置安全限制，只能调用引擎自身所允许的函数和库。

4.3 交易路由

中继链除了需要对跨链交易的合法性进行验证，同时还肩负着跨链交易路由的职责。路由对象包括两个方面：一是跨链交易，二是具有回调的跨链交易回执。

一个区块中所有要被路由的跨链交易会被构建成如图 4-3 所示的 Merkle Tree，其中 Merkle Root 会经过验证节点的签名。（AC1 代表的是从 A 发到 C 的 1 号跨链交易）

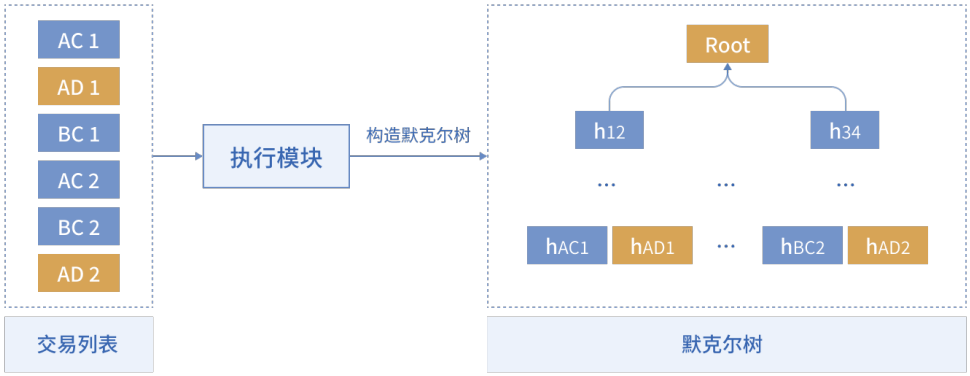

图 4-3 跨链交易路由

中继链执行完一个区块后会根据目的链 ID 进行交易分类，如图 4-4 所示路由模块根据 C 和 D 两个目的链把跨链交易分成两部分。Pier C 和 Pier D 会从中继链同步与自己相关的跨链交易和跨链交易证明。如果跨链交易是跨层级的，Pier 还需添加该条跨链交易来自中继链的证明信息，以供目的中继链进行验证。

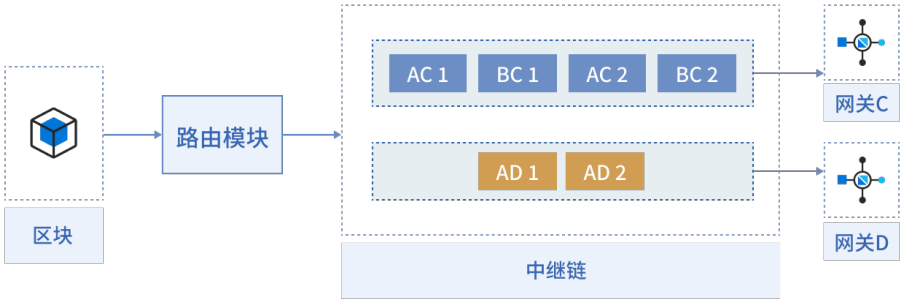

图 4-4 跨链交易分类与执行

跨链网关 Pier 对于不同来源链的交易可以并行执行。执行完的回执分为两种，一种是没有回调的跨链交易对应的回执，在返回中继链后完成跨链。另一种是有回调的跨链交易对应的回执，经过中继链的排序打包后返回来源链。

4.4 事务管理机制

跨链需要保证跨链交易的原子性和一致性，即来源链和目的链上的交易要么都成功，要么都失败回滚。为此，中继链提供了事务管理机制，通过内置的事务管理合约，来保证不同业务场景下跨链交易的事务性。针对非资产交换类业务场景，中继链提供了基于多链消息表事务管理方案；针对资产交换类业务的事务管理方案比较复杂，将在第七章中详细阐述。

4.4.1 多链消息表

如图 4-5 所示，以一对多的跨链事务场景为例，多链消息表方案将一个跨链事务拆分成针对不同应用链的多个子事务。来源链首先执行子事务，执行成功后向中继链发送跨链事务消息，该消息中包含各目的链的跨链请求、跨链成功后的来源链回调信息和跨链失败后的来源链回滚信息。中继链事务管理合约根据跨链事务消息生成全局跨链事务 ID 和不同目的链的子事务 ID，并初始化各个事务 ID 的状态。之后，中继链将各个子事务路由到多个目的链。目的链执行完子事务后根据执行结果向中继链反馈子事务状态，中继链事务管理合约将进行相应的更新。只有所有的子事务都成功，跨链事务才成功；一旦有一个子事务失败，该跨链事务便失败。各个应用链（包含来源链和目的链）的跨链网关可以获取中继链的全局事务和子事务的状态，来对应用链进行业务上的“回调”操作或“回滚”操作。

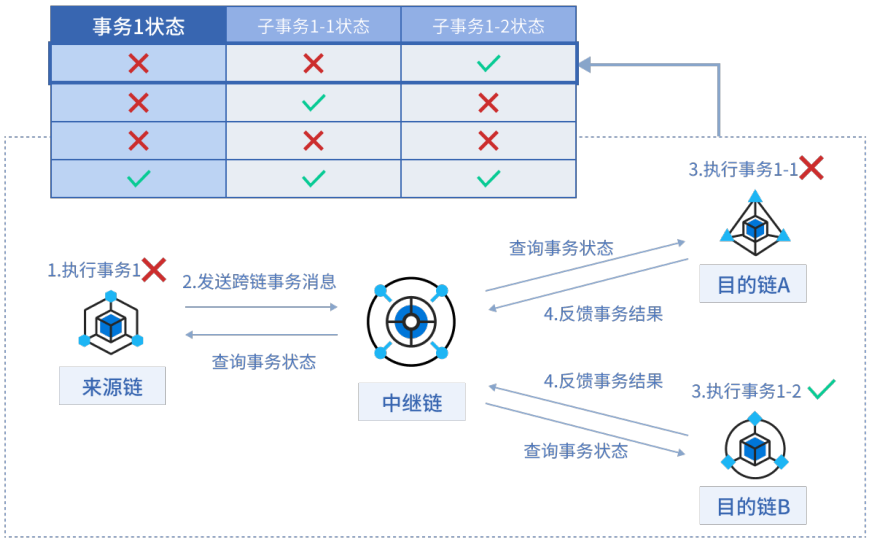

图 4-5 多链消息表方案

多链消息表方案相对于传统二阶段事务方案的主要优势在于不需要过多地改造业务合约，降低了业务合约编写的复杂度。

4.4.2 事务回滚机制

事务管理机制的回滚包含失败回滚和超时回滚两种类型。

失败回滚即目的服务执行跨链交易失败，目的网关向中继链提交失败的交易回执，中继链验证后，来源链网关对来源服务进行回滚。

超时回滚即跨链交易长时间无响应，需要通过回滚来保证跨链双方状态统一。

超时回滚如果发生在来源链服务上，如来源网关因网络原因连接不上中继链，则来源网关直接向来源服务进行事务回滚的调用，同时向中继链提交一个无效的跨链交易来递增中继链上的跨链交易 index，进而通知目的服务将跨链交易 index 递增。

图 4-6 来源链超时回滚示意图

超时回滚如果发生在目的链服务上，即中继链在指定的超时块高内没有收到目的服务的跨链交易回执（超时块高由 IBTP 结构的 TimeoutHeight 字段决定），中继链则将该跨链事务设置为回滚状态。来源链网关可以获取需要回滚的跨链交易和证明信息，并对来源链服务进行回滚；当目的链网关向中继链提交将目的链服务的回执时，中继链会返回该事务已经回滚的状态信息，目的网关得到该信息后对目的链服务执行回滚操作。

图 4-7 目的链超时回滚示意图

4.5 跨链自治

BitXHub 平台的中继链提供了完善有效的跨链治理机制。中继链自身节点的构成是联盟自治的基础，通过丰富的治理服务实现全方位的治理管控，此外，中继链还提供规范的提案模型、灵活的投票策略和科学的评价体系，充分保证成员规范工作、系统健康升级、联盟稳态发展。

图 4-8 治理机制

4.5.1 中继链节点构成

中继链的节点由创世节点、普通节点和审计节点三种构成，每个节点都有一个管理员。创世节点和普通节点管控跨链联盟内的主要治理业务，初始状态下只有创世节点，联盟成员可以持有普通节点申请加入中继链参与协同治理。审计节点由监管部门持有并申请加入中继链，审计节点的管理员主要对链上数据进行审计监管而不参与业务治理。

4.5.2 治理服务

中继链上的治理服务涵盖了对跨链平台上各种行为的管控，包括中继链节点管理、应用链管理、验证规则管理、链上服务管理、统一身份管理、跨链权限管理、数据同步管理、治理业务管理、投票策略管理等。

- 中继链节点管理：管控中继链自身节点的准入、冻结、解冻等业务；
- 应用链管理：管控应用链的注册、更新、冻结、激活和注销等业务；
- 验证规则管理：管控应用链跨链交易验证规则的注册、更新等业务；
- 链上服务管理：管控链上服务的注册、更新、冻结、激活和注销等业务。

4.5.3 提案模型

当跨链平台上产生治理服务范围内的业务时，中继链会自动生成一个由业务发起人提交的相应类型提案，以供中继链管理员进行投票治理。

治理提案模型包括提案的结构和生命周期。提案的结构如图 4-9 所示。

图 4-9 提案结构

提案 ID 与提案发起人账户地址绑定，作为提案的唯一标识；提案类型标识提案所属的治理服务类型，如应用链管理、验证规则管理等；事件类型标识提案中发生事件的类型，如注册、更新等；提案状态与提案的生命周期对应，有 proposed、approved、rejected 等状态；投票信息包含了该提案当前收到的投票信息、计票信息、超管投票信息等，便于推进提案的投票计票流程；选民信息包含了该提案面向的选民列表以及当前可用选民数、最低投票选民数等信息，便于推进提案的投票计票流程；治理对象信息包含了治理对象的 id 及其提案发起前状态，记录发起前状态是为了便于投票拒绝时的状态恢复；状态转换说明包含了提案的提交理由、结束理由、撤回理由等信息，便于相关人员了解提案的状态转换原因；提案锁定信息与提案优先级策略相关的锁定信息；扩展项可附加提案需要的相关信息，如更新提案可以添加更新前后对象的信息对比明细。

提案创建后一般会经历 proposed、approved、rejected 三个状态，特殊情况下还可能出现 pause 状态。其生命周期具体如下图所示：

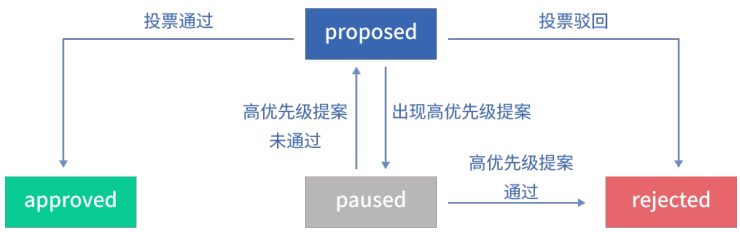

图 4-10 提案生命周期

4.5.4 提案管控流程及投票策略

治理提案提交后需要由中继链管理员进行投票，中继链上的投票管控流程如图所示。

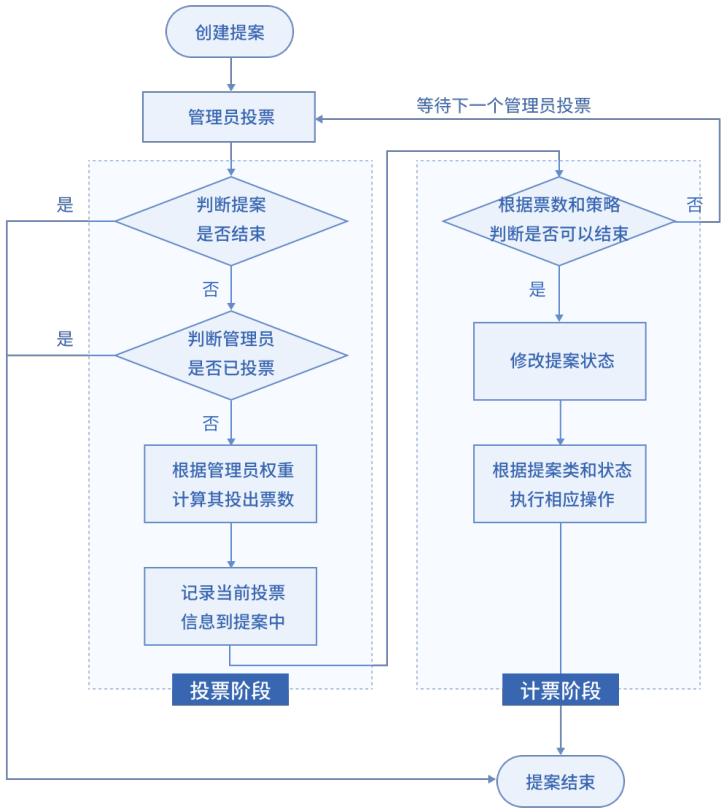

图 4-11 提案管控流程

投票管控流程主要分为投票和计票两个阶段。投票阶段由各个中继链管理员进行投票，管理员只能对未结束状态（即 proposed 状态）的提案进行投票，每个管理员只能投一次，但实际投出票数根据管理员的权重进行计算。计票阶段中继链会根据提案的当前投票情况和投票策略判断该提案是否达到结束要求（即通过或拒绝）。每次投票后都会进入计票阶段，若计票时发现提案达到结束要求将会触发相应的业务操作，反之，该提案将继续等待下一个管理员的投票。

提案的投票策略支持多种策略灵活适配，即不同的治理服务可以设置不同的投票策略，投票策略主要包括以下三种类型：

- 简单多数制：若当前赞成票数多则通过提案，反之则拒绝提案；
- 绝对多数赞成制：在收到票数较少时，需要大量赞成票才可以通过；
- 绝对多数反对制：在收到票数较少时，需要大量反对票才可以通过。

绝对多数赞成制和绝对多数反对制支持投票策略的自适应调节，即投票率越低，需要的赞成或反对票数越少，当全员参与投票时，这两种策略均收敛为简单多数制。这种自适应调节可以有效保证提案通过与否的结果代表了群体多数意愿。

4.5.5 治理评价体系

为了提高联盟自治效率、保证联盟健康发展，中继链提供科学的评价体系。中继链将根据链上账户行为或链上服务情况等信息对中继链管理员、联盟成员以及链上服务等对象进行评价。

联盟支持基于评价结果的奖惩制度，从而推动中继链管理员积极参与治理、抵制成员非法作恶、激励高质量链上服务生态，进而打造优质跨链联盟体系。

第五章 跨链网关

在跨链场景下，应用链如何便捷地接入跨链系统以达到良好的扩展性，对于激发跨链系统生态的活力至关重要。跨链网关 Pier 是一个能够满足应用链接入便捷性和支持跨链网络多层级扩展关键。

本质上来说，Pier 是一种连接不同区块链系统的交互组件，在 BitXHub 中可以充当两种角色：

- 连接应用链和中继链。在单中继链的层级中，Pier 作为一个中间部件来简化区块链接入跨链系统的过程，增强接入跨链系统的便捷性；
- 连接不同的中继链。在多中继链形成的区块链网络中，Pier 通过 P2P 组网的方式在多个层级中起到“路由器”的作用。

为了实现上述的便捷性和可扩展性，Pier 在应用链的适配和核心功能的实现上做了灵活设计。

5.1 插件机制

现有的区块链平台在架构设计上存在着较大的差异，如何将其快速、便捷地接入跨链系统是一个亟待解决的问题。

插件机制的设计将 Pier 中和应用链交互的模块与跨链网关自身核心功能模块进行解耦，从而方便更多的应用链加入跨链系统。在 Pier 运行时，通过动态加载插件的方式完成应用链的接入。为了提升与应用链的交互能力，具体应用链插件需要根据不同区块链的特性实现具体的接口，交互接口需要满足以下几个功能：

- 监听相应区块链上的跨链事件并传给核心模块进行处理；
- 执行来自于其他区块链的跨链请求；
- 能够查询相应区块链上已收到和已执行的跨链请求状态。

5.2 核心模块

5.2.1 跨链交易收集

Pier 在对接具体的应用链时，不仅要监听交易，而且需要保证跨链交易的有序性和可验证性。为此，Pier 需要拥有应用链一定的权限，例如收集 Fabric 跨链交易时，需要有权限获取背书信息。

由于应用链的不同，Pier 收集跨链交易的策略也应视情况而制定。下面从同构应用链和异构应用链两个角度进行分析。

对于同构应用链，Pier 在监听到跨链事件后，只需要根据区块存储结构获取该跨链消息的合法性证明，即可构造 IBTP 结构。

对于异构应用链，以以太坊为例进行分析。以太坊采用基于概率性确认的 PoW[10]共识机制，所以 Pier 收集到一个跨链交易时，并不能立即确认其是否为有效交易。在这种情况下，Pier 需要设定一个等待阈值（一般为 20 个区块后），使得该跨链交易有极大概率被确认后，再将其转发到中继链。

考虑到单个 Pier 可能产生恶意行为，如谎称交易已经确认等，需要采用 Pier 集群的方式来增强跨链网关的可信度。如图 5-1 所示，每个 Pier 由权威机构进行背书，并在中继链上引入对作恶网关的惩罚机制，通过经济激励的方式防止多个 Pier 联合作恶。另外以太坊本身未对跨链交易签名，所以需要多个 Pier 确认跨链交易的存在并对其进行签名，然后转发到中继链上。

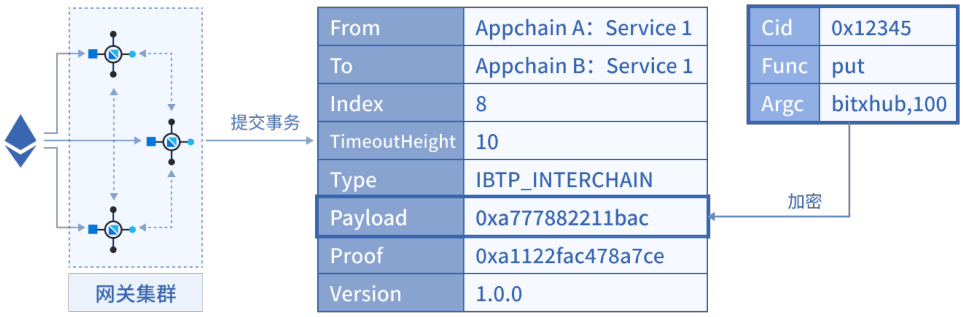

图 5-1 公链跨链交易构建

5.2.2 同步与执行

Pier 提交跨链交易之后，中继链会对跨链交易进行打包。各个 Pier 节点会自动同步中继链上的跨链交易并通过合法性证明确认跨链交易的可信。

之后，Pier 使用与交易来源方协商的对称密钥解析跨链交易，得到具体的 IBTP 结构，并构造出跨链事件。Pier 根据具体应用链的特定规则，选择相应的插件处理。

5.3 网关直连模式

在跨链参与多方信任基础较高，对跨链交易传输安全性要求不那么高的场景，BitXHub 的积木架构可以灵活切换为链对链的网关直连架构，在该种架构下，不同的链通过各自的跨链网关直接相连，具体可以参考直连架构设计。

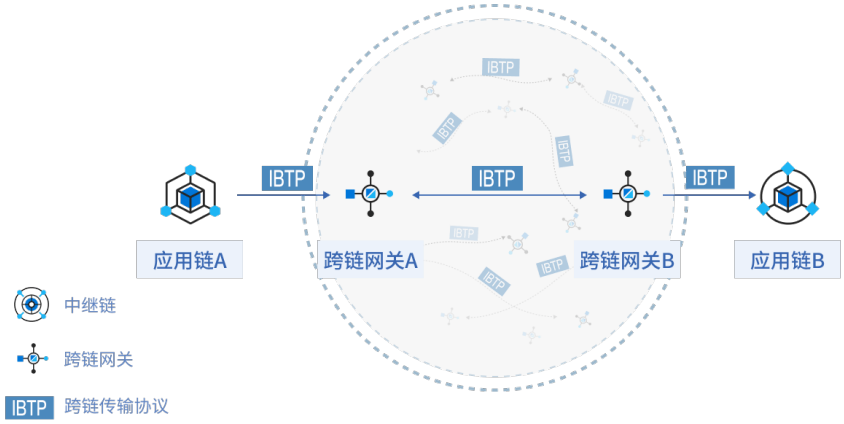

图 5-2 网关直连模式

网关直连架构下，由于跨链多方没有中继链作为公证人来进行跨链管理，需要在各自应用链部署事务管理合约来实现链间跨链交易的事务管理。事务管理合约只能通过 BitXHub 在各应用链部署的跨链管理合约 Broker 进行合约调用，并且仅管理由该链发出的跨链交易的事务状态。

5.3.1 事务状态转换

应用链上 Broker 合约在抛出跨链事件的同时，在事务管理合约中记录该笔跨链交易的相关事务信息，在收到目的链回执后更改相应事务状态。事务状态转换图如图 5-3 所示：

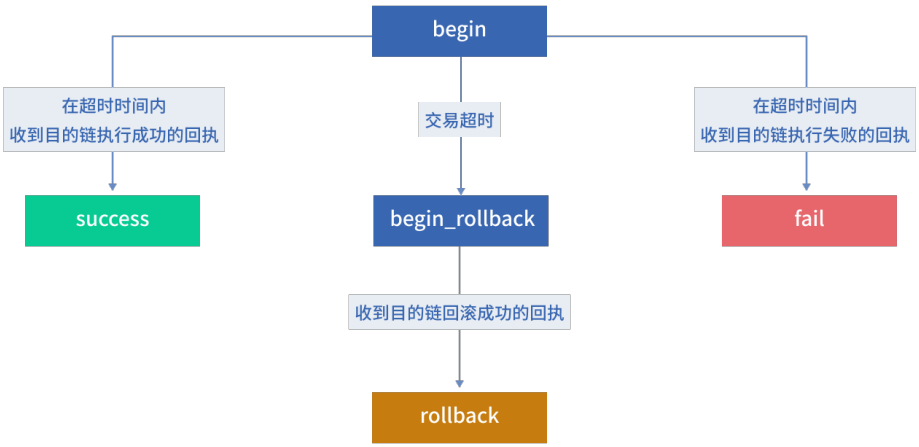

图 5-3 事务状态转换图

Begin -> Success 流程：

正常的跨链交易的事务状态转换流程如下：

1） Broker 合约抛出跨链事件的同时，在事务管理合约中将该笔跨链交易的事务状态设置为 Begin，并记录该笔跨链交易的发起时间；

2） 来源链网关收到该笔跨链交易后转发至目的链网关，目的链网关将该笔跨链交易提交至目的链执行；

3） 目的链执行成功后返回 IBTP 类型为 IBTP_RECEIPT_SUCCESS 的跨链交易回执至来源链网关；

4） 来源链 Broker 合约收到 IBTP 类型为 IBTP_RECEIPT_SUCCESS 的跨链交易回执后，通知应用链上业务合约执行相应回调操作并将事务管理合约中该笔跨链交易的事务状态设置为 Success。

Begin -> Fail

当目的链执行失败时，来源链 Broker 合约将根据跨链交易回执执行相应回滚操作，具体的事务转换流程如下：

1） Broker 合约抛出跨链事件的同时，在事务管理合约中将该笔跨链交易的事务状态设置为 Begin，并记录该笔跨链交易的发起时间；

2） 来源链网关收到该笔跨链交易后转发至目的链网关, 目的链网关将该笔跨链交易提交至目的链执行;

3） 目的链因为各种原因执行失败后返回 IBTP 类型为 IBTP_RECEIPT_FAILURE 的跨链交易回执至来源链网关;

4） 来源链 Broker 合约收到 IBTP 类型为 IBTP_RECEIPT_FAILURE 的跨链交易回执后, 组织业务合约执行相应回滚操作并将事务管理合约中该笔跨链交易的事务状态设置为 Fail。

Begin -> BeginRollback -> Rollback

每笔跨链交易将根据具体的业务场景携带一个超时时间信息, Pier 会根据事务管理合约中记录的跨链交易事务状态以及发起时间，当跨链交易超时时触发超时回滚操作。具体参考超时回滚机制。

5.3.2 事务回滚机制

跨链需要保证跨链交易的原子性和一致性，即来源链和目的链上的交易要么都成功，要么都失败回滚。为此，采用网关直连跨链场景中，提供了超时事务回滚机制。

在网关直连场景下, 来源链在发起跨链交易时可根据具体的业务场景设置超时回滚时间, 由跨链网关根据事务管理合约中记录的事务信息协调跨链双方执行超时回滚操作。

具体实现流程描述如下:

1) 来源链业务合约抛出跨链事件，该事件中包含一个超时时间;

2) 来源链跨链网关捕获到跨链交易，首先计算该笔跨链交易是否超时;

(i) 若超时，则说明来源链与来源链网关这条链路之间网络出现故障，触发超时回滚，执行步骤 5;

(ii) 若未超时，则来源链网关将该笔跨链交易发送至目的链网关，继续执行步骤 3;

3) 目的链网关收到跨链交易后, 将跨链交易提交至目的链执行, 目的链执行结束后, 生成相应的跨链交易回执发送至来源链网关;

4) 来源链网关收到跨链交易回执后，计算该笔回执对应的跨链交易是否超时;

(i) 若超时, 则说明来源链网关与目的链网关或目的链网关与目的链的链路之间网络出现故障，触发超时回滚，执行步骤 5;

(ii)若未超时，则来源链网关将该笔跨链交易回执提交至链上执行，执行结束后更新事务状态，该笔跨链交易结束。

5) 触发超时回滚后，来源链网关首先通过跨链管理合约 Broker 组织来源链业务合约进行回滚，回滚成功后更新该笔跨链交易在事务管理合约中的事务状态为 BeginRollback，生成类型为 IBTP_RECEIPT_ROLLBACK 的 IBTP 发送给目的链网关；

6) 目的链 Broker 合约收到 IBTP 类型为 IBTP_RECEIPT_ROLLBACK 的跨链交易后，组织目的链业务合约进行回滚，并生成类型为 IBTP_ROLLBACK_END 类型的 IBTP 发送给来源链网关；

7) 来源链网关收到 IBTP 后提交至来源链 Broker 合约，此时会更新事务管理合约中的事务状态为 Rollback，该笔跨链交易结束。

5.4 跨链路由网络

在大规模多层级的区块链网络中，需要由跨链路由网络进行跨链交易的传递。跨链路由网络是一个由多个 Pier 组成的 P2P 网络。中继链加入跨链网络之前，需要由一种具有跨链路由功能的 Pier 负责连接中继链和跨链网络中的其它 Pier，该 Pier 的身份代表中继链在跨链路由网络的标识，同时其它中继链的管理员会向中继链注册新加入的中继链的相关验证信息，从而通过网络标识和验证信息可以快速便捷地传递可供验证的跨链交易。

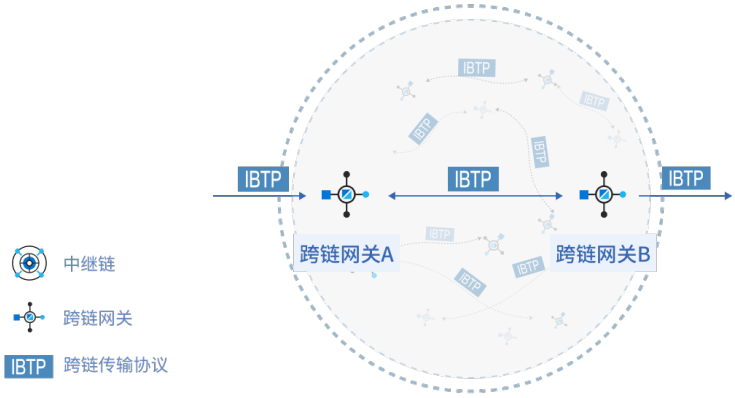

图 5-4 跨链路由网络拓扑图

首先每个 Pier 会维护一个全网的分布式哈希表，记录应用链和中继链的关联关系。Pier 在加入路由网络时，需要广播所属中继链管理的所有应用链信息，其他 Pier 会根据广播信息进行分布式哈希表的更新。

当 Pier 传递跨链交易时，首先中继链各个节点会对跨链交易进行多重签名，随后通过路由算法查找跨链交易的目的中继链对应的 Pier，并通过跨链路由网络进行传输。目的 Pier 验证跨链交易来源的真实性，通过验证后再将跨链交易发送到相应的中继链，中继链验证跨链交易来源的有效性，跨链交易的执行结果也通过该方式将执行结果返回来源中继链[11]。

第六章 链原生跨链数字身份

链原生跨链数字身份是 BitXHub 跨链平台首次提出的区块链原生支持的数字身份机制，通过链原生支持数字身份能够实现身份在多条链间的互通互认，可以更加方便地实现以身份为中心的数字资产在不同链间的可信流转。此外，BitXHub 还通过数字身份的其他功能（如可验证声明）优化原有的跨链流程，促使跨链流程更规范和可扩展。

BitXHub DID 整体设计符合 W3C DID（Decentralized Identifiers）[12]及相关规范，并且目前已经在 DIF Universal Resolver 下注册了 did:bitxhub 域名。BitXHub DID 整体设计目前分成两部分：链的数字身份（Chain DID）和账户的数字身份（Account DID）。

6.1 基本设计

6.1.1 标识符格式

BitXHub 将运用数字身份来标识应用链、节点、链上账户甚至合约等各种实体，其标识符格式略有差别：

其中，链的身份标识（Chain DID）的格式设计为：did:bitxhub:chain-name:.，第一字段为 did 固定标识，第二字段为 BitXHub 网络的固定标识，第三字段为每条链的链名，第四字段以.结尾。如 did:bitxhub:relaychain001:. 。

对于每条链上基于账户地址的账户数字身份标识（Account DID）其格式为：did:bitxhub:chain-name:address。

其中第四个字段为用户的账户地址。如：did:bitxhub:relaychain001:0x12345678。

6.1.2 文档格式

在 W3C DID 规范下每个标识符都对应有一个 Doc，用于存储该 DID 标识相关的身份信息，如公钥数组、认证方式等。由于 W3C DID 相关规范处于不断完善之中，为了维持系统的稳定，我们将 Doc 格式进行了改进保留了对数字身份而言最重要的一些字段，如公钥数组、验证方式等，并支持对未来的扩展。

6.1.3 系统基本结构

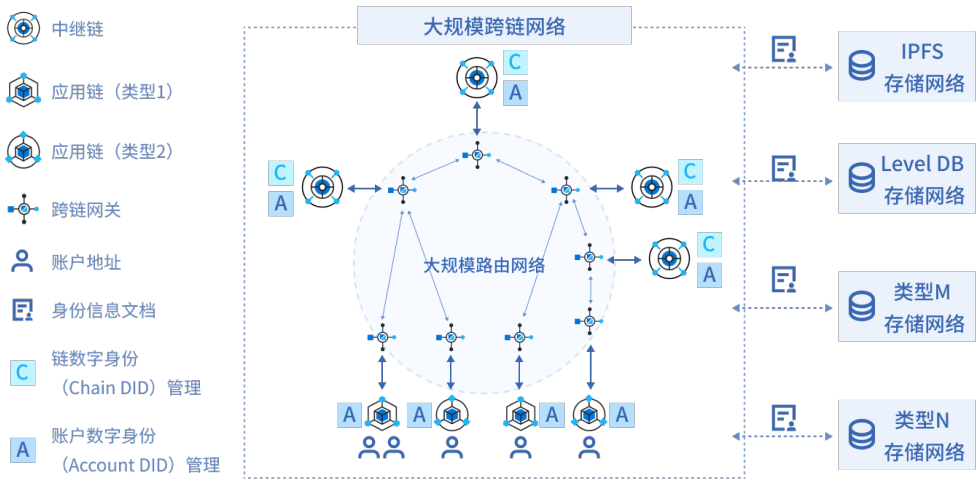

图 6-1 BitXHub 数字身份系统网络拓扑示意图

数字身份系统的网络拓扑示意图如上所示，整体为树状层次架构。其中，所有中继链都会实现链数字身份（Chain DID）管理功能及账户数字身份管理功能，而应用链将会实现账户数字身份（Account DID）管理功能。考虑到整个网络面临账户扩展的需求，BitXHub 提供了文档链下存储功能。

6.1.4 身份标识表

每条区块链的身份标识表（DID Table）用于记录身份标识相关的身份信息，其表项包含身份状态、文档存储地址、文档哈希等字段。存储链身份信息的身份标识表叫做“链身份标识表”(Chain DID Table)，存储账户身份信息的身份标识表叫做“账户身份标识表”(Account DID Table)。Chain DID Table 与 Account DID Table 的不同点在于，Chain DID Table 会在网络中所有中继链上进行同步，而 Account DID Table 不会，Account DID Table 只存储自己链上的账户身份信息。

6.2 功能流程

本章节以链数字身份为例来介绍下数字身份在实际应用场景下的使用流程。

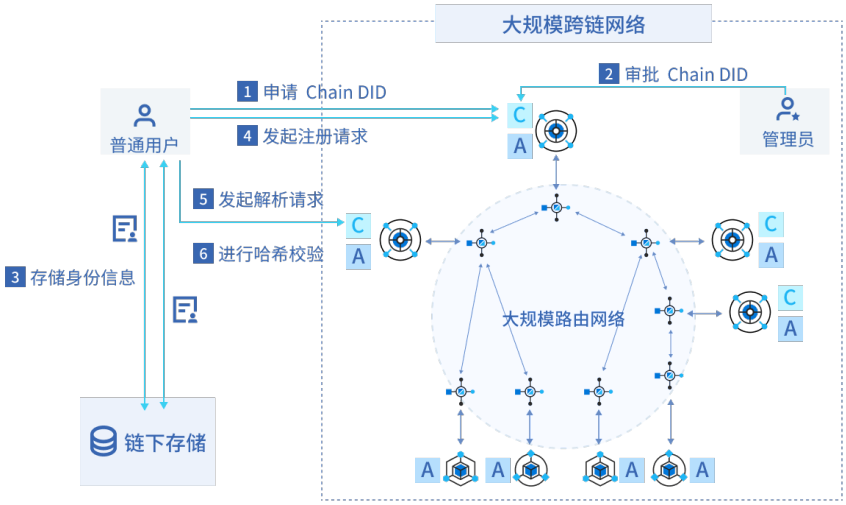

图 6-2 BitXHub Chain DID 的大体功能流程

如图 6-2 所示是 Chain DID 的大体功能流程，其中：

1) 申请 Chain DID：如图中［1］。向管理中继链发起请求，发送 Chain DID 名称（图中为 did:bitxhub: appchain001:.;

2) 审批 Chain DID：如图中［2］。管理员对用户的申请请求作出审批，结果为“审批通过”或者“审批驳回”;

3) 注册 Chain DID：如图中［3］、［4］。当审批通过后，可以进行注册。首先在流程［3］中，将链身份的相关信息组织成 Chain Doc 进行存储得到存储地址 docAddr 和内容哈希 docHash。然后在流程［4］中，以 docAddr 和内 docHash 等为参数向最高中继链发起注册请求;

4) 解析 Chain DID：如图中［5］、［6］。Chain DID 的信息在所有中继链上同步，所以可以向任意一条中继链发起 Chain DID 解析。首先在流程［5］中，客户端对任意中继链发起解析请求（图中为左边第一条），获得该身份的相关信息（包括存储地址和内容哈希）。然后在流程［6］中，通过存储地址获取实际的 Chain Doc，进行哈希校验，如果通过则证明 Doc 可信。

当某个 Chain DID 的注册过程在管理中继链上完成后，需要将该新建的 Chain DID Table 同步到网络中其他所有中继链，每次同步的单位是一个表项。当 Chain DID Table 表项发生

新建、更新、删除时，应该进行数据同步。Chain DID 数据同步需要借助中继链间跨链操作，这里中继链会向网络中其他所有中继链都发送一次跨中继链的跨链消息以进行数据同步，且整个同步过程是由最高中继链依次往下逐级同步的。

Account DID 的部分功能流程和图 6-2 类似，不过流程上少了申请和审批两个过程，同时 Account DID 不需要进行链间的数据同步。每条实现了 Account DID 的链管理以自己链上地址为基础的账户数字身份，不同链之间不互相影响。

除此之外，Account DID 的解析和 Chain DID 有一些区别。首先，由于某个 Account DID 的信息在全网里只存在一条链上，因此无法进行负载均衡。

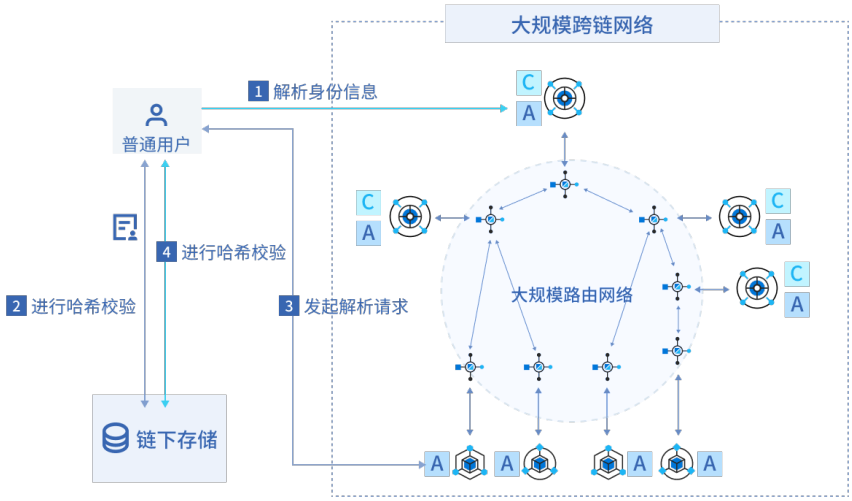

图 6-3 Chain DID 解析流程图

其次，在进行 Account DID 解析时需要先知道其信息存储在哪条链上，然后再去对应的区块链上拿取相应的 Account DID 信息。典型的 Account DID 解析流程如图 6-3 所示,其中：

1) 从 Account DID 中得到该身份所在的链 Chain DID，然后发送给任意一条中继链的 Chain DID 管理解析得到该链的 Chain docAddr 和 docHash；
2) 通过 Chain docAddr 和 docHash 向链下存储获取 Chain Doc，进行哈希校验，如果通过则证明 Chain Doc 可信；
3) 通过 Chain Doc 中该链的信息，向链发起 Account DID 解析请求，拿到 Account DID 的 docAddr 和 docHash；
4) 通过 Account docAddr 和 docHash 向链下存储获取实际的 Account Doc，进行哈希校验，如果通过则证明 Account Doc 可信，最终完成整个解析流程。

6.3 应用案例

如图 6-4 所示，假设图中的区块链（relaychain1、appchain001、appchain002）已经完成 Chain DID 的注册、跨链交易验证规则的注册，并且在各自的 Chain Doc 中写入了足够的链相关信息。

在使用 DID 之后，IBTP 中的 Proof 将会是类似可验证声明（Verifiable Credential，VC）的格式。

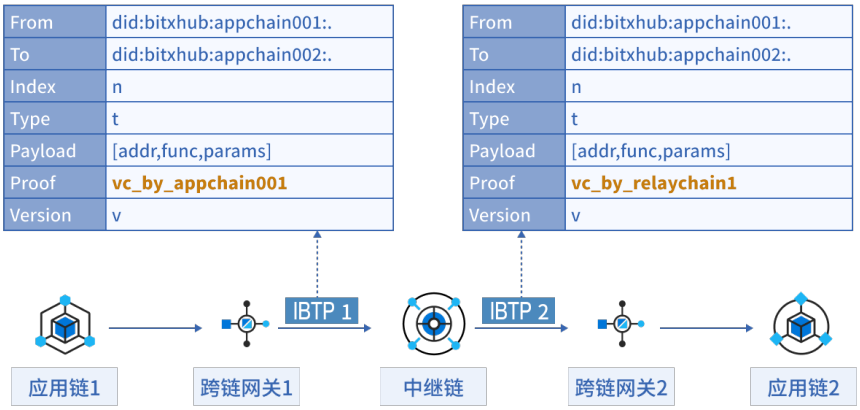

图 6-4 基于 DID 的跨链流程

如图 6-4 所示，假设跨链交易从 appchain001 发起；然后跨链网关 1 捕获 appchain001 抛出的跨链事件将其组装为 IBTP1；IBTP1 的 Proof 是一个由 appchain001 作为发行者发的 VC。然后网关 1 会将 IBTP1 发送到中继链，relaychain1 根据 appchain001 的链信息以及注册的验证规则调用正确的验证方式对 IBTP1 进行验证。如果验证失败则流程结束，如果验证成功，则网关 2 会将验证结果同步并组装 IBTP2，其 Proof 是一个由 relaychain1 发行的 VC，包含了验证结果。跨链网关 2 接下去会将 IBTP2 发送给目的链进行处理。appchain002 收到 IBTP2 后也会对其 Proof 进行验证，如果验证通过则执行 IBTP2 中的 Payload。VC 的验证过程主要是验证 VC 内容的正确性和 VC 签名的合法性两步。

第七章 跨链数字资产交换

在数字资产交换、数据同步和业务互补三大典型跨链场景中，数字资产交换是安全要求最高、价值最大的跨链场景。跨链资产交换是指不同应用链的资产可以通过跨链的方式实现链上价值自由交换，极大提升资产的流动性。针对不同区块链的数字资产模型，中继链会采取不同的资产跨链方案，每种方案的侧重点不同，旨在为用户提供完备、安全、稳定、高效的跨链数字资产交换体验。中继跨链平台将提供三种跨链数字资产交换的方式：中继节点多签方案、基于安全多方计算和门限签名方案和去中心化用户自主控制托管方案。

7.1 中继节点多签方案

中继节点多签方案主要是针对资产模型是 UTXO 的应用链的资产跨链解决方案。由于采用 UTXO 模型的区块链大部分无法提供图灵完备的智能合约，所以中继链采用“同步区块头+多签地址托管”的方式进行数字资产跨链，以一种多中心化的方式来映射应用链的数字资产，主要流程如下：

- 启动阶段：

1) 中继链多个节点生成应用链的多签地址；
2) 跨链网关监听该多签地址的数字资产余额变动；
3) 如果必要的话，中继链节点需要同步应用链区块头，保持最长链的更新

- 数字资产跨链锚定流程，如图 7-1 所示：

1) 用户发送交易将自己的数字资产锁定到应用链多签地址，该交易的备注中包含用户的中继链地址、资产数量等；
2) 跨链网关监听到应用链的锁定交易，将应用链交易原文 Tx 和交易存在性证明 Proof 提交给中继链；
3) 中继链验证通过后，映射对应的应用链数字资产到用户的中继链地址上，完成跨链锚定操作。

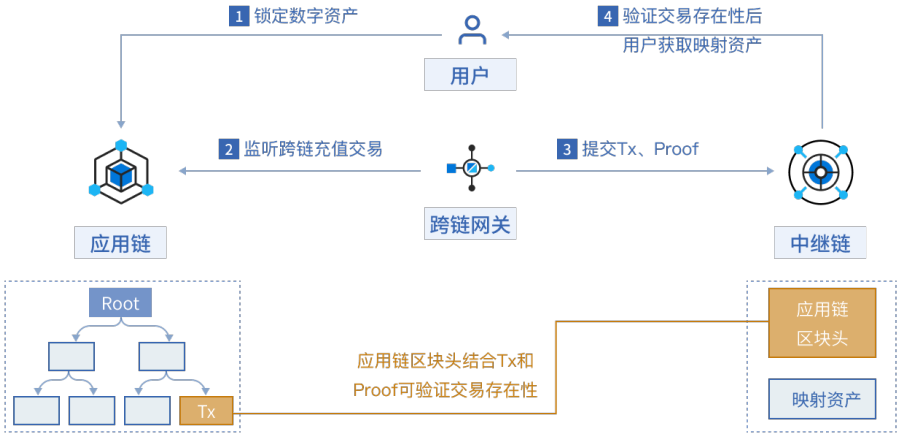

图 7-1 中继多签数字资产跨链锚定流程图

- 数字资产跨链解锚流程，如图 7-2 所示：

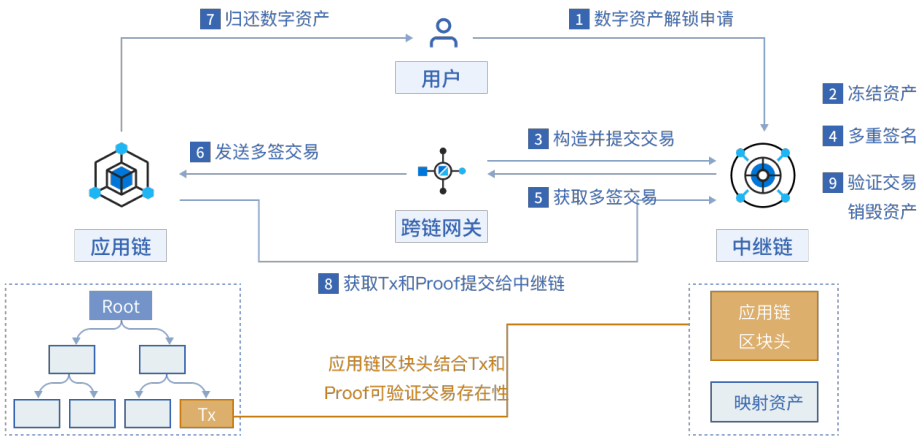

图 7-2 中继多签数字资产跨链解锚流程图

1) 用户向中继链发起数字资产解锁申请，申请中包含用户在应用链的地址、数字资产数量等；
2) 中继链将用户的映射资产冻结，抛出数字资产跨链解锚事件；
3) 跨链网关监听到中继链的跨链解锚事件后，构造出应用链的未签名交易并提交给中继链；
4) 中继链多个节点对收到的未签名交易验证通过后，进行多重签名；
5) 跨链网关监听到中继链已完成对该交易的多重签名，将多签交易广播给对应应用链；
6) 用户从应用链收到等量的数字资产；
7) 跨链网关从应用链获取交易原文 Tx 和交易存在性证明 Proof 提交给中继链验证；
8) 中继链验证通过后，销毁用户的映射资产，完成跨链解锚操作。

7.2 基于安全多方计算和门限签名方案

基于安全多方计算和门限签名方案是针对支持门限签名算法的应用链资产跨链解决方案。该方案中，N 个中继链节点通过安全多方计算协商，使每个节点各得到一个私钥片段和公钥片段；公钥片段公开，大于等于 M（M<N）个公钥片段可以恢复出完整公钥，同时，只需有大于等于 M 个节点使用私钥片段给交易进行签名，便可得到完整的交易签名，该签名可以使用公钥来验证。该方案中，多个节点维护各自的私钥片段，不会暴露完整私钥，是一个更加安全的去中心化的数字资产跨链方案。本方案的主要流程如下：

- 启动阶段，如图 7-3 所示：

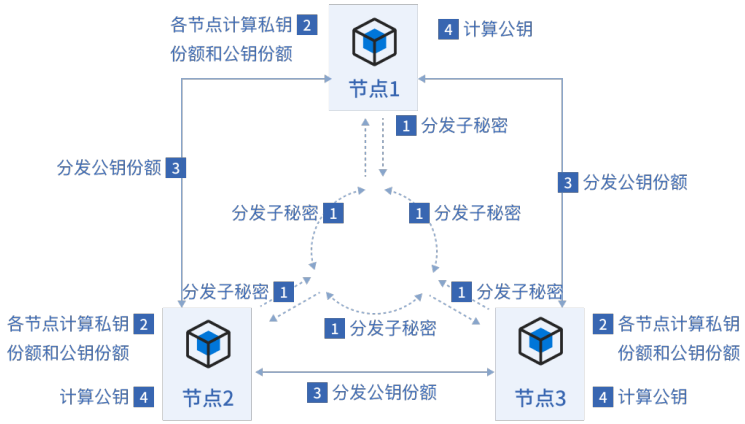

图 7-3 安全多方计算密钥分发图

1) N 个中继链节点各自选取随机数，通过秘密分享的方式将由随机数构成的子秘密通过安全信道发送给其他 N-1 个中继链节点；
2) 各个中继链节点根据收到的所有其他节点发送的子秘密计算自己的私钥片段和公钥片段；
3) 各个中继链节点将公钥片段分发给所有其他中继链节点；
4) 所有中继链节点可以根据不少于 M 个公钥片段计算得到完整的公钥，并可根据公钥得到对应的账户地址，该账户称为托管账户；
5) 如果必要的话，中继链还需要同步应用链区块头，保持最长链更新。

- 数字资产跨链锚定流程和中继节点多签方案的锚定流程相同，如图 7-1 所示：

1) 用户发送交易将自己的数字资产锁定到应用链的托管账户地址，该交易的备注中包含用户的中继链地址、数字资产数量等；

2) 跨链网关监听到应用链的锁定交易，将应用链交易原文 Tx、交易存在性证明 Proof 提交给中继链；

3) 中继链验证通过后，映射对应的应用链数字资产到用户的中继链地址上，完成跨链锚定操作。

- 数字资产跨链解锚流程，如图 7-4 所示：

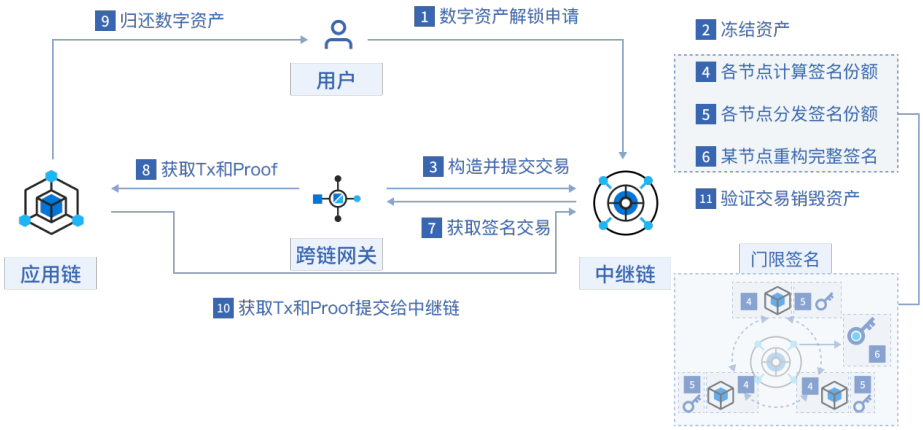

图 7-4 门限签名数字资产跨链解锚流程图

1) 用户向中继链发起数字资产解锁申请，申请中包含用户在应用链的地址、数字资产数量等；

2) 中继链将用户的映射资产冻结，抛出数字资产跨链解锚事件；

3) 跨链网关监听到中继链的跨链解锚事件后，构造出应用链的未签名交易并提交给中继链；

4) 中继链多个节点对收到的未签名交易验证通过后，使用自己维护的私钥片段计算签名份额，并将产生的签名份额分发给其他中继链节点；

5) 当某个中继链节点收到不少于 M 个签名份额时，其便可重构出完整的交易签名；

6) 跨链网关监听到中继链已完成对该交易的签名，将该签名交易广播给对应应用链；

7) 用户从应用链收到等量的数字资产；

8) 跨链网关从应用链获取交易原文 Tx 和交易存在性证明 Proof 提交给中继链验证；

9) 中继链验证通过后，销毁用户的映射资产，完成跨链解锚操作。

7.3 去中心化用户自主控制托管方案

去中心化用户自主控制托管方案是针对用户需要对托管数字资产全方位保护的解决方案，用户具有托管跨链数字资产的一票否决权。这种方案可以在上文提到的两种方案的基础上加工实现。

- 基于上文中继链节点多签方案，通过把用户的公钥加入到多签脚本中，避免用户或中继节点任一方单独花费跨链数字资产的风险，实现真正的去中心化托管。
- 基于上文 MPC 和门限签名方案，通过用户独自保管具有一票否决权的门限签名私钥片段，实现真正的去中心化托管。

其主要流程如下：

- 启动阶段：

1) 用户将其在应用链的公钥、应用链信息和用户的中继链地址发往中继链；
2) 中继链生成由用户密钥片段（公钥）绑定的特殊地址；
3) 跨链网关监听该多签地址的数字资产余额变动；
4) 如果必要的话，中继链节点需要同步应用链区块头，保持最长链的更新。

- 数字资产跨链锚定流程和中继节点多签方案的锚定流程相同，如图 7-1 所示：

1) 用户发送交易将自己的数字资产锁定到应用链特殊地址，该交易的备注中包含用户的中继链地址、数字资产数量等；
2) 跨链网关监听到应用链的锁定交易，将应用链交易原文 Tx 和交易存在性证明 Proof 提交给中继链；
3) 中继链验证通过后，映射对应的应用链数字资产到用户的中继链地址上，完成跨链锚定操作。

- 数字资产跨链解锚流程，如图 7-5 所示：

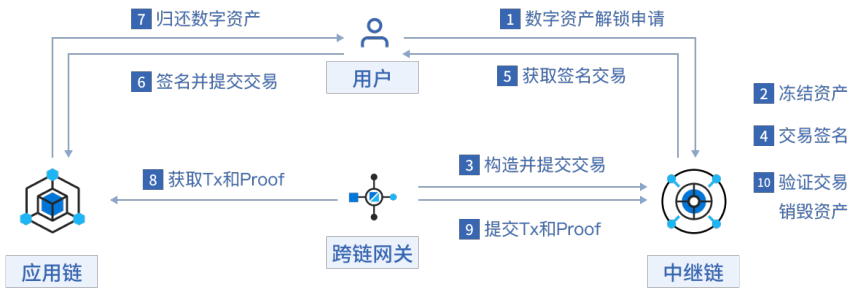

图 7-5 用户自主控制数字资产跨链解锚流程图

1) 用户向中继链发起数字资产解锁申请，申请中包含用户在应用链的地址、数字资产数量等；
2) 中继链将用户的映射资产冻结，抛出数字资产跨链解锚事件；
3) 跨链网关监听到中继链的跨链解锚事件后，构造出应用链的未签名交易并提交给中继链；
4) 中继链多个节点对收到的未签名交易验证通过后，进行签名；
5) 用户观察到中继链的签名交易已经完成，获取签名交易；
6) 用户使用自己的私钥对交易进行签名，然后提交交易到对应的应用链；
7) 用户从应用链收到等量的数字资产；
8) 跨链网关从应用链获取交易原文 Tx 和交易存在性证明 Proof 提交给中继链验证；
9) 中继链验证通过后，销毁用户的映射资产，完成跨链解锚操作。

总结与展望

BitXHub 为区块链提供了一套自主创新、透明可信的跨链技术方案，基于通用跨链消息传输协议 IBTP 打造了一个异构区块链跨链示范平台，秉承着去中心、可扩展、高可用、易接入的设计理念，为链上的资产、数据、服务开拓价值互通的渠道，助力区块链技术从“链孤岛”到形成“链网络”的发展。BitXHub 将不断探索并为用户提供多方共治理、安全高可用、通用易扩展、留痕可追溯的跨链服务，真正实现万链互联、价值互通的区块链互联网。

参考文献

[1] Buterin,Vitalik. "Chain interoperability." R3 Research Paper (2016).

[2] Castro,Miguel, and Barbara Liskov. "Practical Byzantine fault tolerance." OSDI. Vol. 99. No. 1999. 1999.

[3] Cachin, Christian. "Architecture of the hyperledger blockchain fabric." Workshop on distributed cryptocurrencies and consensus ledgers. Vol. 310. 2016.

[4] Wood, Gavin. "Ethereum: A secure decentralised generalised transaction ledger." Ethereum project yellow paper 151.2014 (2014): 1-32.

[5] Szydlo, Michael. "Merkle tree traversal in log space and time." International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Berlin, Heidelberg, 2004.

[6] Nakamoto, S. . "Bitcoin: A Peer-to-Peer Electronic Cash System." (2009).

[7] Fabric docs: https://hyperledger-fabric.readthedocs.io/en/release-1.4/txflow.html

[8] Kan, Luo, et al. "A multiple blockchains architecture on inter-blockchain communication." 2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C). IEEE, 2018.

[9] Haas, Andreas, et al. "Bringing the web up to speed with WebAssembly." ACM SIGPLAN Notices. Vol. 52. No. 6. ACM, 2017.

[10] Chen, Zhi-dong, et al. "Inter-blockchain communication." DEStech Transactions on Computer Science and Engineering cst (2017).

[11] Gervais, Arthur, et al. "On the security and performance of proof of work blockchains." Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. ACM, 2016.

[12] W3C Decentralized Identifiers (DIDs) v1.0 Core architecture, data model, and representations.

趣链科技

小助手微信

杭州（总部）地址：杭州市滨江区丹枫路399号2号楼A楼2001室
官网：https://www.hyperchain.cn 市场合作：marketing@hyperchain.cn