Chain Interoperability

跨链论文中文版翻译，来自V神

1. Chain Interoperability

在区块链领域刚出现早年间，“单链控制所有”是可以被原谅的，然而在这几年，这种思路已经越来越不被接受。在公链领域，不同的项目在安全性、隐私性、效率、灵活性、平台复杂性、开发人员操作难度甚至是是否有政治价值等方面的权衡选择往往有很大差别。在私链和联盟链领域，显然在不同领域存在不同的区块链，甚至在同一领域也存在多条不同的区块链。在中情况下，一个问题呼之欲出：区块链之间如何交互呢？使用将密码身份验证自然包含到每个操作中的平台的优点之一是，与以前现有的系统相比，我们实际上可以在平台之间提供更紧密，更安全的耦合。我们可以超越集中式系统中最常见的方法，即通过API完成区块链之间的交互，在某些情况下，甚至可以达到在一个链上具有智能合约代码来直接验证其他链上事件的共识确定性的程度，达到完全的去中心化信任。

可以互操作的链开拓了一个新的领域。在这个领域中，资产从一个平台转移到另一个平台，或是付款转账，或是获得另一条链内部数据的访问权（例如“身份链”和支付系统可能会成为合理的存在）会变得容易，甚至可以直接由第三方实现无需基础区块链协议的运营商做出任何额外的工作。目前为止，链间交互的概念已经有了很多理论但是缺少实践，这个也太事故突变为跨链操作要求不止一个，而是两个已经存在的，稳定高效且功能强大的区块链平台来构建该体系结构，但是这种情况已经慢慢开始改变。

到目前为止，跨链研究主要是在公链的环境下进行的，并且本文档在某种程度上是这方面迄今为止已完成工作的“文献综述”。尽管其中的许多技术同样在私链和联盟链中的配置中适用，但此类设置也将面临其自身的独特挑战，本文不对这些挑战进行论述。例如，至今尚未有一个单独的许可链被大量使用，因此尚不明确公链实验中的共识算法和其他功能能在多大程度上转化为可以在私链和联盟链中适用的模块。另外，很大一部分的“交互操作”要求与传统的系统和协议（(SWIFT, SEPA, FIX, 等）进行交互；不幸的是，这项研究不适用于公链，因为在可以预见的将来，公链不太可能直接与这些系统相连。

2. Types of Interoperability

从技术上讲，跨链操作的策略可以分为三大类：

1. 中心化或者多重公证人机制，当A链上发生某事件时，公证人或公证人团体同意对B链采取动作。
2. 侧链/中继，一个区块链内部的系统，可以验证和读取其他区块链中的事件和状态。
3. 哈希锁，在链A和链B各自设置一个操作，这两个操作有相同的触发条件，通常这个条件是揭示某个哈希值的原像（即由怎样的原始数据可以哈希得到此哈希值）

   跨链交易还可以实现许多潜在的应用场景：

   1. **便携式资产**，例如，能够从其所有权最终具有权威性的“家庭账本”中获取一个Fedcoin（由政府发行的假想数字资产）单位，安全地转移到另一条链上进行交易，将其用作抵押品或其他方式 充分利用这一链条上的优势，并有信心在需要时始终可以选择将Fedcoin转移回本地账户（例如，将信任最小化的一对一支持）
   2. **付款与付款或付款与交付**：从技术术语上来讲，这个概念通常被称为“原子交换”，“原子”一词在古希腊语中意为“不可分割的”，可以被理解为双方的交易要么都发生，要么都没有发生。本质上，这样做的目标是目标是允许用户X将数字资产A转移到用户Y，以换取Y将数字资产B转移到用户X（其中A和B在不同的链上，而X和Y在每个链上都有帐户），以保证原子性和安全性的方式。
   3. **跨链预言机**：例如，人们可能会想象一条链上的智能合约仅在收到证明另一条链上的身份预言指定该

地址是特定的唯一身份的证据时才对某个地址执行某些操作。 请注意，在这种互操作事件过程中，正在读取的链不会改变。

4. **资产负担**：锁定链X上的资产A，锁定条件取决于链Y上的活动。应用场景包括留置权，金融衍生产品的抵押品，破产回扣，法院命令以及涉及保证金的各种用例。
5. **普通跨链合约**：例如，将X链上的股息支付给所有权在Y链上注册的资产的所有者。

在以上应用场景中，最受关注的两个是跨链数字资产和跨链交换（即付款对付款和付款对交付）；但是，后面的部分将以通用术语描述适用于所有应用程序的跨链问题的解决方案，然后从零开始着眼于安全性以及与跨链便携式数字资产和跨链资产交换有关的其他问题。然而，值得注意的是，跨链资产的可移植性只是实现其预期目标的一种方法。 在像BTC和ETH这样的无抵押的仅加密资产的世界中，这确实是实现预期目标的唯一途径，但是如果我们要处理发行人支持的资产，则可以采用其他方法（例如，在多个链上分别发行资产， 隐含可兑换性在**法律层**而不是在区块链层）也是可能的。

尽管经常使用“侧链”一词来指代跨链便携式数字资产，但是这个词的许多用法在很多方面都令人误解和混淆。第一，Block-stream组织对“侧链”的一般定义为“侧链是验证来自其他区块链的数据的区块链”。但这种说法很珍贵，在这样的定义下，在此定义下，由于有了BTCRelay（在后面的部分中进行了介绍），以太坊已经成为比特币的侧链。在正常情况下，术语“侧链”通常用于

指Blockstream所谓的“锚定侧链”，从其他区块链读取数据的区块链功能，用于促进跨链资产的可移植性。这是一个进步，这要求产生相互作用的两方互为侧链或者要求存在一个值得信赖的联盟（请参阅后面关于公证方案的部分）和跨链通信机制之上分层的方案一样，该机制实际上实现了链资产可移植性逻辑。

第二，“链A是链B的侧链”这句话不能合理地说明两条链之间存在从属关系。就像上面提到的例子，以太坊可以被称作比特币的侧链，但是从任何角度都不能说以太坊从属于比特币系统。

术语“侧链”和“锚定侧链”最初是在区块链是单个主导资产的所在地的情况下出现的，因此很自然地将包含锚定代币的链视为“锚定侧链”。然而，实际上“锚定侧链”是区块链之上的单个资产的一种属性，而不是区块链本身的一种属性。

例如，以太坊系统包含以太币，一种以太坊自带的代币资产，然而如果其他区块链根据其本身所需进行了适当的协议更改（或者如果出现了可信赖的联盟），则以太坊也许最终会包含叫“e-BTC”的资产，由BTC在比特币区块链上以1:1支持。e-DOGE由DOGE在Dogecoin区块链上以1比1支持，等等; 因此，以太坊将同时成为所有这些区块链的“侧链”。但是，显然因为这些应用存在，以太坊区块链本身没有锚定任何有意义的东西。因此，最好使用诸如“链B可以读取链A”，“链A的中继存在于链B上”或“D是账本A上的跨链便携式数字资产”之类的术语会更

好。而不是简单地说一个给定的链是否是一个“侧链”。

Notary Schemes（公证人机制）

在技术上认证大部分跨链动作最简单的机制是**公证人机制**。在公证人机制中，使用一个受信任的实体或一组受信任的实体来声明X链上发生了Y链上的给定事件，或者关于Y链的特定声明为真，从而声明了X链。这样的实体（公证人）可以是活动的，正在侦听并根据某个链中的事件自动采取行动，也可以是被动的，仅在被询问时才发布签名消息。目前这个跨链方向做出最先进地努力的是Ripple开发的Interledger project项目。
Interledger 至少在其描述为“原子模式”的情况下，使用拜占庭容错共识算法，以便在一组公证人之间就是否发生给定事件进行共识操作，然后发布用于基于这种共识算法最终确定有条件的付款的签名。

Interledger还设想了可以构成这种公证机制的支付链概念。如果X方和Y方希望交换数字资产捆绑，但这些捆绑存在于链A和F中，其中A和F没有直接链接，则可以在中间分类账B，C，D和E上找到中介。 如果每对相邻的中间分类帐确实有直接链接（即，A和B，B和C等之间存在公证人和交换机会），则可以确定一堆满足A和F偏好的交换，而同时为中介机构赚取少量套利收入，然后对整个交易所使用单一的共识流程，以确保所有转移都发

生或不发生。

需要注意的是这不是实现这种原子化的唯一方法，哈希锁也具有这样的功能并且在闪电网络中也半夜跨链交易器的角色。但是，它在技术上相对简单，并且只在其信任模型下可以实现其预期的目标（即，拜占庭所选择的公证人所占比例不超过一小部分）。这些公证人可以被一些特定的交易单独的确认。另外，可以想象的是公证人可以与以下所描述的方案结合使用：创建链间交换协议，这些协议的安全性尽可能达到其能达到的最高级别，但是如果基础链还不支持更不信任的中继机制，则退回到公证方案。

另一个相关的方案是联邦锚定侧链的概念。这个概念意图拥有一种可跨链移动的资产，其中的单向或者双向的锚定是通过multisig机制实现的。这种方式已经被Liquid实现了，Liquid是由BlockStream创建的支持的BTC侧链。在Liquid链中用户可以将BTC转移到由参与者议会控制的multisig地址中，同时接收到名为“L-BTC”的Token，一旦Liquid Chain共识机制认为BTC交易已经发生，便立即进入Liquid链。L-BTC可以在Liquid链中被自由交易，同时用户也可以销毁L-BTC，这时控制多重签名的联盟将从多重签名向销毁L-BTC的一方发送等量的BTC（即1:1的兑换比例）。

请注意，用户可以在两个公链之间或者在一条公链和一条联盟链之间进行锚定。在后一种情况下，使组成财团的实体也成为控制多重签名的实体可能是最简单的。在前一种情况中，可以说该方案只有在“侧链”也是独立于该锚定资产的应用程序所在地时才具有价值。否则，使用公链共识只会增加费用，而不会对系统的实际安全性产生相应的改进，而这种安全行本质上依赖于参与者议会的诚实度作为信任模型的一部分。

Relays（中继模型）

中继机制是促进跨链操作更直接的一种方式，在这种机制中，中继不是依靠受信任的中介机构来将有关一个链信息提供给另一条链，而是多链链本身自己高效地完成了这个过程。一般方法如下问所述。假设在B链上执行的智能合约想要了解链上发生的特定事件或者A链状态下的某个特定对象在某个特定时间包含某个值。另外假设A链的设计类似比特币或者以太坊，因为其具有”区块“、”区块头“的概念，其中”区块头“是一种紧凑的信息，以一种经过密码验证的方式”代表“该块（可能是以状态数据的方式）（最可能使用Merkle树）。

我们可以在B链上创建一个智能合约，该合约将采用A链上这些区块头其中的一个。采用一套标准的合法性校验过程来对A链的共识算法进行验证进而验证区块头的合法性。在PoW中，这个合法性过程将对PoW过程进行足量的校验，以至于不会被冲突的区块所影响。而在联盟链流行的传统拜占庭容错共识算法中，将包括言则会那

个2/3的校验者对这个区块头进行的签名。一旦Relay确认区块头已完成校验，Relay就可以通过针对区块头验证Merkle树的单个分支来分别验证任何所需的交易或帐户/状态条目。

这种所谓的“轻客户端验证”技术的使用非常适合中继，因为资源从根本上限制了区块链。实际上，使用链A的内在机制来完全校验B链或者是链B的内在机制来完全校验A链都是不可能的，对于这种机制局限性出现的原因是两个沙箱不可能同时包含彼此：A链需要重新执行B链上重新执行的A链的部分，包括A链上曾经重新执行过的B链的部分（无限迭代），以此类推。然而，借助轻客户端，链A包含链B的小块而链B包含链A的小块（按需拉动）的协议是完全可行的。想要验证链A上特定交易、事件或状态信息的链B上的中继上的智能合约，就像传统的轻客户端一样，将验证链A的加密哈希树的分支，然后验证该分支的根在内部的区块头，并且如果两项检查均通过，则将接受该事务，表明事件或状态信息是正确的。（请注意，请注意，由于区块链是完全独立的环境，并且无法自然地访问外部环境，因此用户需要将链A的相关部分馈送到链B中； 但是，由于数据在密码学意义上是“自我验证”的，因此无需信任该用户提供此信息。）

Relays机制是非常强大的，它们基本上可以不受限制地用于资产可移植性，原子交换或任何其他更复杂的用例（有关中继之上的资产可移植性系统的视图，请参见上图）。

请注意，通常中继后的复杂密码验证可以轻松地抽象出来并对开发人员不可见。事件验证本身可以做成一个智能合约，其他合约可以将其称为**事件验证预言机**。事件读取可以抽象为一种异步操作：可以想象一种跨链智能合约编程语言，其中包含原始的createEvent（destinationChain，params），该语言注册事件并为其分配唯一的ID，并具有onReceiveEvent（senderChain，params）函数 } {...}，仅当传入事件的加密证明时才可以调用，并且当提供这种证明时，它将记录保存在存储器中，以防止再次用同一事件调用该函数。

可能有人注意到，这种异步事件读取模型可以看作是以太坊1.0风格的有状态区块链方案和类似于比特币的“未用交易输出”模型的混合体，其中UTXO的等同物是未消耗的事件记录。上面的异步事件体系结构假定事件被快速消耗，但是人们也可以想象一种编程语言，其中直到需要时才消耗事件，而未消耗的事件成为长期状态中非常重要的一部分。

跨链可移植“ Fedcoin”代码的草图可能如下所示：

```
function sendCrossChain(destChain, to, value) {
        if (balances[msg.sender] < value) throw;
        createEvent(destChain, {name: SEND, to: to, value: value});
        balances[msg.sender] -= value;
        crossChainBalances[destChain] += value;
}
function onReceiveEvent(senderChain, params) {
        if (params.name == SEND) {
                if (crossChainBalances[senderChain] < params.value) throw;
                balances[params.to] += params.value;
                crossChainBalances[senderChain] -= params.value;
        }
        ...
}
```

该合约将在Fedcoin主链和次要链上均初始化。代码的逻辑很简单。sendCrossChain函数首先检查源链是否有足够的Fedcoin进行发送； 如果没有，则退出并显示错误。 如果源链确实有足够的资金，它将创建一个事件，规定应在目标链上创建代币，减去源链的余额，并增加目标链的余额。代码中用于管理目标链余额的部分是一项安全功能：即使目标链的共识以某种方式被破坏，目标链也只能将与主链发送的Fedcoin一样多的Fedcoin发送回主链。从而防止潜在的不受信任的子链中的错误使攻击者无法在仍完好无损的链上创建无限量的资产。

onReceiveEvent函数确保源链可以发送给定数量的FedCoin，如果是，它将增加收件人的余额并减少源链的余额。 请注意，在每个次级链上，crossChainBalances [mainChain]（其中mainChain是

主Fedcoin链的标识符）将被初始化为实质上无穷大；这表示二级链接受主链作为Fedcoin的权威发行人。

比特币和以太坊之间的中继合约已成功以BTCRelay的形式实现，这是以太坊上可以读取比特币链的智能合约。但需要注意的是，互操作性是单向的：比特币无法读取以太坊链，因为其脚本语言不够复杂，无法做到这一点。BTCRelay在一些场景下已经获得了使用。在撰写本文时，有一个名为EthereumLottery.io的应用程序，其中彩票智能合约逻辑本身位于以太坊公共链上，但它使用比特币区块头作为随机性的来源，因为每个比特币区块头都有较大的回报，因此操作起来更加昂贵。这是一个原始的，尽管在技术上非常出色的示例，可以被称为“跨链预言”应用程序，并且中继的使用在提高应用程序的安全性方面起着重要作用。

Relays for Cross-Chain Atomic Swaps（跨链原子交换中继）

除了资产可移植性和跨链预言机之外，中继的下一个使用案例是跨链原子交换，即将**链A上的资产M交换为链B上的资产N**。对于BTCRelay和以太坊，MakerDAO团队已经在努力实现这一目标。 然而，不幸的是，由于比特币协议本身的基本局限性，任何这样的系统都必然有其自身的弱点。一个特别值得关注的问题是竞争条件攻击。例如，假设甲方希望以1 BTC的价格出售50 ETH，

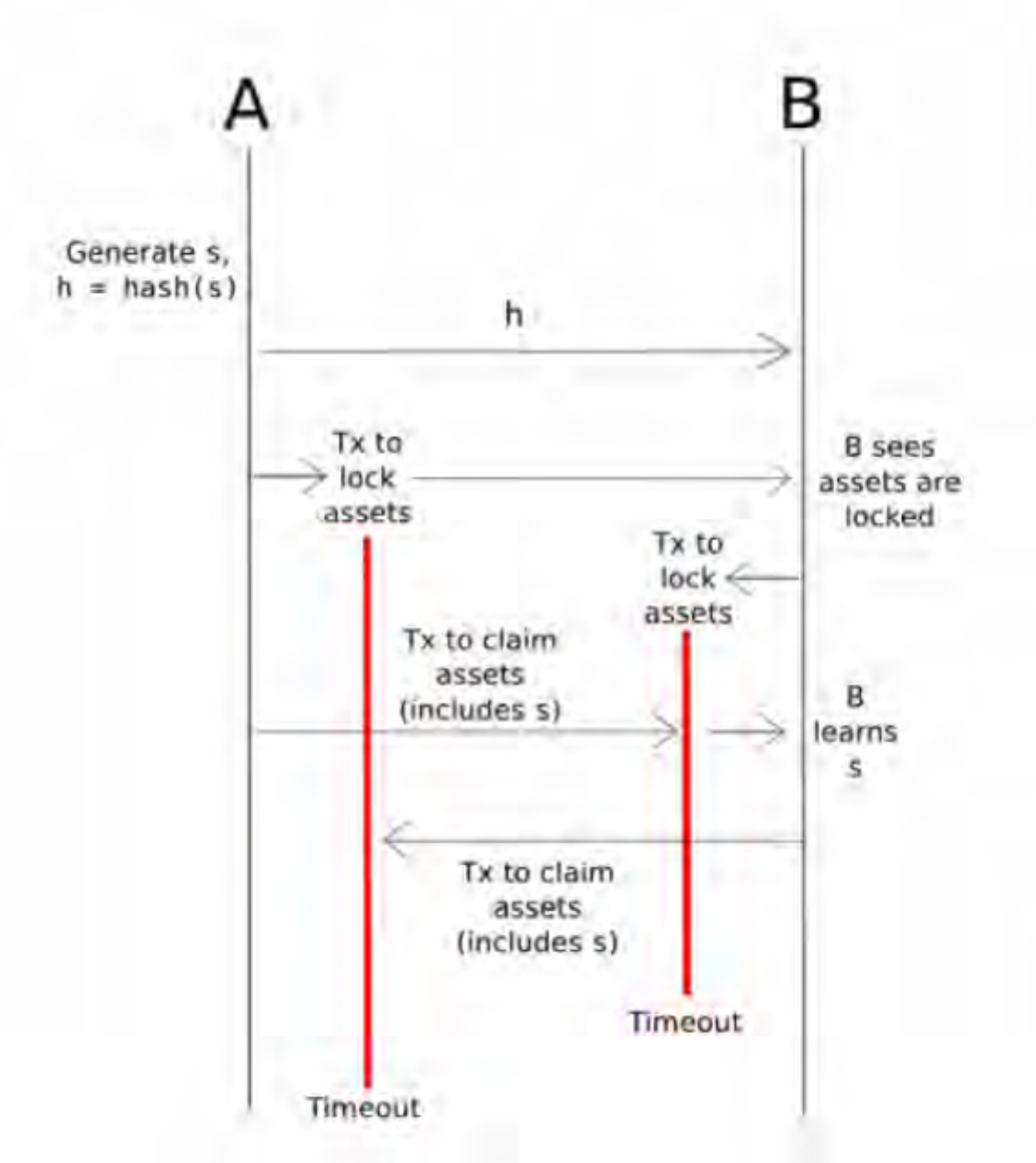
并将50 ETH存入一份合同，该合同实质上说“谁提供证明他们发送1 BTC到X的地址就得到50 ETH”。乙方随即发送了1个BTC。甲方现在可以自己在自己的合同中发送1个BTC，并尝试在乙方可以之前向自己索取ETH； 他们很有可能会成功，给甲方留下2个BTC和50 ETH，而乙方则一无所有。

如果比特币区块链具有类似以太坊的智能合约功能，这将不是问题：这样的话，比特币上的目标地址可以简单地是一项合约，该合约可以自动退款除第一个以外的所有传入转账。因为没有这种能力，所以我们最好的方式就是在以太坊一侧使用信号量：允许买方保留交易的专有权一段时间，然后在窗口内安全地提出索赔；这确实是MarkerDAO使用的解决方案。但是，信号量本身可能会成为拒绝服务的载体，因为有人可能会试图反复保留交易权而从不主张权力。使预订操作的代价高昂，可以解决此问题，但这种方式会导致用户体验问题，即通过该方案必须持有以太币才能购买更多的以太币（有多种方法可以使用不受信任的第三方来解决此问题，但是它们很复杂，而且在比特币方面需要复杂的2比2托管和锁定时间技术）。但是，不管解释的细节如何，更广泛的观点是，竞争条件是基于中继的应用程序的合法安全问题，开发人员应像在编写单链时应注意重新进入一样注意它们。 编写Web应用程序时，以太坊应用程序或SQL注入攻击。

中继的一个相关弱点是它们本身的异步性。 特别是如果一条链或两条链使用的共识算法缓慢完成，那么一条链要花费很长时间才能验证另一条链对某些操作具有共识，这限制了跨链操作的速度。对于跨链原子资产交换，这是一个特别不幸的弱点，因为市场汇率在原子交换操作过程中可能会发生变化。 因此，中继可以在具有快速终结性的链上很好地工作，但是在它们的速度太慢的情况下，该方法可能会出现问题。 （尽管在第三条链上使用出块时间非常快的存款可能是解决任何问题的一种方法）

Hash-locking

还有另一种实现跨链原子操作的众所周知的技术，该技术不要求链之间深入地了解对方：哈希锁。哈希锁技术是由Bitcoin论坛上名为TierNolan的用户所提出的，最近，Interledger协议正在积极探索将哈希锁作为消除公证人信任要求的一种手段。对于跨链数字资产交换，此机制的最简单描述如下：



1．A产生一个随机密钥s，并且计算该密钥的Hash h，A把h发给B

2．A和B都按照以下规则将其资产锁定到智能合约中（A先锁定，B在看到A的资产成功锁定后才锁定）。 在A方面，如果在2X秒内提供了s，则资产将转移到B，否则将其发送回A。在B方面，如果提供了正确的s（即哈希值为h的值） X秒钟，然后将资产转移到A，否则将资产发送回B。

3．为了在B的合同中要求资产，A会在X秒内发送s。 但是，这也可以确保B学习到允许B从A的合同中索取资产的密钥。

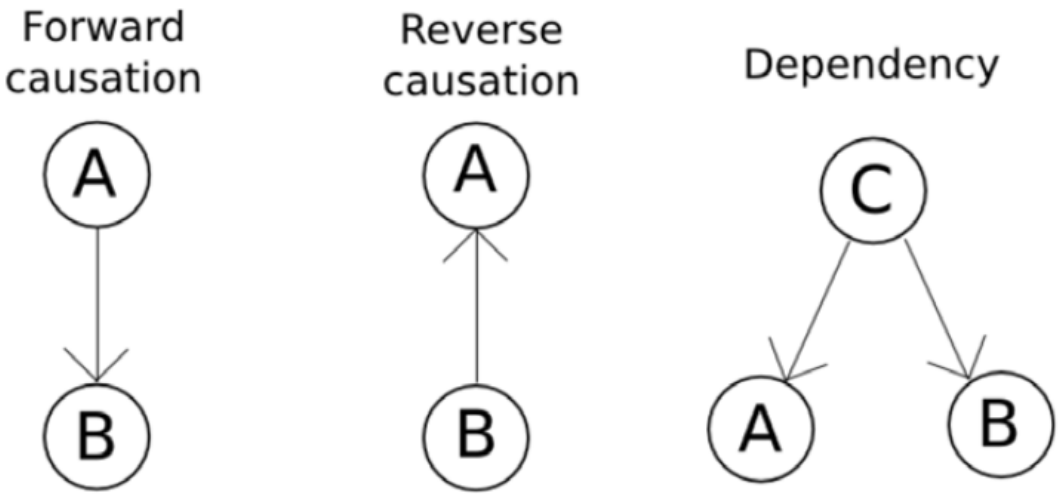
请注意这被证明是原子化的。如果A在X秒内发送了s，则这提供了至少X秒的窗口，B可以在该窗口内索取其资产。A在这个过程中可能会犯错并太晚地发送s，阻止他们收回自己的资产，但这是他们自己的错，很容易避免。如果A在X到2X时间内发布了s，则A将不会得到它们的资产，但是B可以得到，然而这是A自己的问题。如果A在2秒钟后显示s（或从不显示），则双方都将收回自己的资产。如果A不锁定资产，那么B也将不锁定资产。如果B没有锁定他们的资产（或错误地锁定了它，例如在s的最后期限错误），则A绝对不会泄露s，从而收回他们的资产。

请注意，在汇率波动的世界中，财务安全保证是不完善的，因为恶意软件A可以等待X / 2秒，并且只有在此之后的汇率朝着有利的方向移动时才会发布s； 通过参与此哈希锁，B有效地给了A免费的选择权。有人可能会争辩说，在均衡状态下，我们可能期望B仅仅期望A以这种方式行事，并且希望获得一定的溢价以换取这种选择（公链去中心化交换将是检验此假设的绝佳机会），尽管即使这种情况也不完美，因为这等于是说，如果某人希望交换自己的资产，那么他要么需要反复重试好几次，或者在价格上给出足够的优惠，才能激励它的对手方一次完成交换，这就减弱了市场参与者有效地对冲风险的能力。

跨链哈希锁定也可以与**状态通道技术**结合使用，以通过利用状态通道比基本区块链层更快的速度来创建更快的交换，从而缓解此问题； 这仍然是研究和开发的活跃领域。请注意，哈希锁仅对原子操作有用，对于资产可移植性或跨链Oracle用例而言没有用。它对跨链oracle用例无用的原因很简单：访问跨链oracle本质上是相对于正在读取的链的被动操作，而哈希锁是双方固有的主动操作 。之所以对资产可移植性没有用，是有些微妙的原因：哈希锁原子交换协议保留了不变性，即每条链上每项资产的总供应量保持不变，因此它实际上无法将资产从一条链转移到另一条链上。 但是哈希锁可以和中继技术结合使用，以提高效率：使用中继将资产从一个链移动到另一个链的可能性确保了一种资产对不同链上同一资产的汇率保持非常接近1：1 就像任何其他汇率都会创造套利机会一样，但是普通用户只是使用哈希锁机制，通过与想要向相反方向移动的人进行交易，将资产从一个链移动到另一个链。

Theory and Implementation

如前所述，在许多不同的情况下，需要多链交互。从高级应用程序的角度来看，用例范围从财务，身份验证到可能被认为具有区块链价值的任何应用程序。但是，从计算机科学理论的叙述中，我们可以创建互操作性类型的简单得多的分类。我们可以将它们定义为因果图，并将其称为正向因果关系，反向因果关系和依存关系；如下图所示：



具有基本统计学背景的人员可能会将此图表识别为代表了三种可能的关联原因类别，但不包括巧合，这是有充分理由的：多链交互在某种程度上是关于（可靠）关联的。正因果关系很简单：链A可以在链B上引起事件（这也可以表示为“链B可以读取链A”）。反向因果关系是链B可能导致链A上发生事件的情况-类似的关系，但值得将其分开考虑（直觉上，将这种因果关系视为与您希望因果关系可能发生的方向相反的方向）。依赖关系是两条链上的动作何时可以依赖于某些其他外部事件而发生。中继可以提供正向和反向因果关系，具体取决于链的功能。公证人可以提供任何因果关系，但是，哈希锁只能将散列值的显示用作“共同原因”，从而提供交叉依赖性。这是理解为什么哈希锁从根本上比中继技术更弱的另一种方式。

如果我们有某种因果关系形式，而没有其他因果关系形式，我们可以问一个问题：也许有某种方法可以将这三种因果关系中的任何一种转化为与其他因果关系相同的东西吗？事实证明，有一种方案依靠一个或多个受链上安全保证金激励的团体（或者可能在联盟链环境中，

具有约束力的合法合同的强度）来激励，但是其能力有限并且具有很重要的限制条件/警告。这种论点是，如果A链可以读取B链，那么人们就可以在链A上创建一个智能合约，其中包含一笔存款，只有在每次在链A上上创建了一个事件，在链B创建了相应的触发器时，才可以偿还该笔押金。这种方式的一种可能的用例是："贫民窟双向锚定"，其中存在一种多重签名，将BTC存储在比特币区块链上，参与者也持有自己在以太坊上的ETH抵押品，参与者必须发送来自多重签名的交易，为了允许从以太坊上的"e-BTC"IOU代币转换回比特币链上的BTC（从BTC转换为e-BTC更加容易：只需使用BTCRelay），将根据需要进行多次签名。请注意，这样的方案确实具有很高的ETH抵押要求，以保证"加密经济上的安全"，因为ETH/BTC速率下降太多，可能没有足够的抵押来抑制多重信号窃取BTC.

哈希锁也可以在某种程度上转换为直接因果关系，尽管可能性要小得多而且难以实现。一个简单的方案包括在A链和B链上同时放置一系列哈希，其中在A链上，当触发跨链消息时，哈希的原像需要押金，而在B链上原像触发跨链消息的接收方。当跨链消息在A链上被触发时，有存款的一方别无选择，只能发送原像，从而也触发B链上的事件。但是请注意，这需要预先安排特定的事件数据。具有更好的属性的方法涉及到从哈希锁转换为签名锁定：链B上的机制将来自某些Oracle的消息是为跨链消息，但随后这些消息可以由第三方重新导入到A链的机制中，这种机制会重新检查正确的消息，并且只签署正确的消息，如果出现不当行为，那么会破坏Oracle的保证

金。但即使如此，上述讨论中所涉及到的复杂性也仅表明这些方法充其量是次要的替代品，并且可以通过使所有链直接存在所有形式的因果关系来实现最佳的多链互通。

如果我们想使用这些原语来满足尽可能多的用例集，我们可能想提出一种表达这些问题的高级跨链编程语言。可以通过事件创建和事件监听器原语来表达因果关系。交叉依赖关系也可以通过事件的使用来表达，尽管这些事件将由**被揭示的哈希承诺**而不是链上的事件触发。对于程序员而言，理想的情况是编写带有A链和B链上的组件的应用程序，这两个组件都包含一行代码，例如，onPreimageReveal（0x172db5b4 ...），然年后程序员可以确信如果原像被揭露，则将运行由事件触发的代码；此代码的编译版本将不仅输出智能合约代码，还有守护线程（或者是守护线程的插件），它将检查多链互通的区块链并交叉发布事件（如果事件在一条链上发布）。这个守护线程可以被任何人调用并且不需要被信任；一个副本可以由合同的受益方运行，也可以由用户、开发人员或者给定的链作为公共服务来组织。

Formal Modeling and Security

可以使用一个模型来对这些跨链脚本进行形式化验证，该模型假设链A上的某些时间戳概念在某个误差余量δ1内是正确的，B链上的时间戳在误差余量δ2内是正确的，并且链之间的消息传递延迟小于δ3（这些增量将包

括封锁时间、发生近距离51%攻击的风险近距离审查等）。因此，例如一个onPreimageReveal事件将在链A和链B上发生且最大时间差异为δ1+δ2+δ3，或者由链A上的合同创建的事件将由链B读取，并且存在其他最大时间差异。 这样一来，考虑到这些假设，那么跨链的付款与交付合同就可以拥有原子性的正式证明。

请注意，在跨链环境中，需要将此类假设纳入正式模型中。在单链环境中，对应用程序运行协议进行建模的必要性大大降低，但可能存在防审查假设的例外情况（“如果A确实希望将交易X纳入链中，他们将能够在10分钟内完成操作”）。这是因为，从分布式计算机内部的角度来看，分布式计算机是绝对安全的：每一项操作都可以按计划保证100％正常工作。但是，在多链环境中，链A和链B的安全模型是分开的，需要对跨链信息传输进行建模，对链B内部的链A验证也必须进行建模，等等。因此，出现了复杂性，一旦我们在下一节中讨论故障模式，这些复杂性就会变得更加有趣。

此外，尤其是在公链的情况下，不可避免地要讨论经济学。例如，有51％的攻击，交易垃圾邮件攻击和其他类似措施。 此类攻击始终是可能的，但也要付出一定的代价，因此，更丰富的形式化模型甚至可能会提出以下形式的声明：“具有这些参数的算法最多可以安全地保护$ 200,000，但不能超过$ 200,000，因为在这样的规模下，攻击在经济在经济上可行并且需要增加时间窗口。

然而，但建模并不总是那么容易，一个特殊的原因是可能同时影响多个应用程序的攻击。例如，通过交易垃圾邮件进行的审查通常无法有效地从高价值应用程序中窃取资金，因为合法发件人只需要进行一次交易就可以保证安全，而攻击者则需要在公链环境中填充每个区块。发件人可以向他们的交易中增加一笔很高的费用，这只会使他们略有退缩，但需要攻击者每小时烧掉数十万美元才能阻止它们。但是这样的攻击会同时损害链上的所有应用程序，因此如果链在正常情况下具有将近全部使用的状态，并且如果这些操作是时间敏感的（例如通道状态结算），则交易垃圾邮件攻击可能会导致进行足够多的动作，是通过同时失败并且支付高额代价。51%算力攻击具有类似的性质，即使每个应用程序的设计方式都使遭受51%的攻击的损失不会超过50000美元。但如果同时存在1000个此类应用程序，那么将花费500万美元的51%攻击将并性地攻击所有这些应用程序，这将会是有利可图的。因此，在公链环境下，类似于金融抵押品和风险敞口管理的风险建模形式可能是合适的。

Governance and Failure Modes（关于失败时的讨论）

在多链互联架构中涉及到的两条链都正常运行时，以上讨论才有意义。但是当其中一条或者双方执行失败或者经历不规则的治理事件时，会发生什么？

快速列出链中可能发生的正面和负面违规情况如下：

1. 51％算力攻击导致交易回滚；
2. 51％的攻击导致成功（全部或部分）审查；
3. 51%算力攻击会导致非法链的创建，这种链不会被其之上的“全节点”接受，但是会被轻节点和中继接受。这种非法链也许会包含非法的状态转换（例如，您实际上不需要10万亿美元的Fedcoins即可欺骗中继器以使其认为您具有进行51％攻击的能力）；
4. “软分叉”的功能更改（如果对这种说法不满意，也可以理解为(2)）。
5. "硬分叉"，即链中的所有或几乎所有用户群均有效地统一使用规则变更或不规则状态转换的新链，并且足够协调以至于旧链上的名称、品牌和社区基本完全转移到了新链上。
6. 网络分裂导致一条链“分裂”为两部分，分裂持续了一段时间。（脑裂）
7. 使用强一致性共识算法的区块链系统丢失了太多节点以至于新区块无法被做出并且会使链系统“停止失败”（可以看作等同于(2)但几乎在所有情况下都将导致(4)）

当跨链应用程序“存在”的一条或两条链上发生的此类事件时，它将如何处理？ 如果不存在故障模式处理，则后果因情况而异。 例如：

1. 交易回滚事件可能会导致原子跨链操作的原子性丧失，因为交易的一部分可能会被撤消（A链上的），而另一部分则不能撤回（B链上的）。

2. 重播欺骗事件可能导致原子跨链操作的原子性损失，因为交易的一部分可能实际上没有发生。
3. 审查事件有时可能会导致原子跨链操作失去原子性，因为它可能在很长时间内阻止提款交易，以至于对手方退出协议（超时）。 但是，执行此操作的能力取决于每个应用程序，并不总是存在这种问题的情况。
4. 一个区块链系统的硬分叉有可能导致“中继”对这个系统的追踪断裂，尽管这种情况也并非一定会发生，并且实际上在很大程度上取决于分叉的细节和中继机制的实现。
5. 脑裂是一种很复杂的情况，因为取决于共识算法和中继的细节，它可能类似于还原事件或故障停止事件。PoW机制中的脑裂会导致返工，传统拜占庭容错共识算法上的脑裂会导致故障停止；以太坊的Casper权益证明模型是混合的，允许中继系统设计选择等待的“确认程度”，从而根据应用在两种故障模式之间进行选择。

总的来说，可以采用强一致性共识算法来使跨链操作取得“更安全”效果。要了解原因，请考虑上面的“Fedcoin”示例，该示例基于某些home ledger，并且在某些外部链A上具有M个现成的硬币。假设链A遭受反复的反向或欺骗攻击。然后，链A可以通过发送一条SEND消息，等待该消息在本机分类帐上的中继中被确认，返回到发送消息之前的一个块并重复执行，将任意数量的SEND消息发送到本机链。本地链将接受来自链A的价值M个硬币的此类消息，并将M个硬币发送给攻击者，但随后它（链A）将停止。 尽管不可避免地会丢失M个硬

币，但本地链对子链余额的核算可以使其免受不必要的损失。请注意，逆向攻击也适用：如果主链可能遭受反向攻击，则攻击者可以用任意数量的硬币填充所有子链，从而使它们毫无价值。（通胀？？？）

现在考虑，假设链A遭受审查检查攻击或者是“fail-stop”(如果链A更强调一致性而不是可用性，则这种故障模式更有可能发生)。现在可以阻止链A上的参与者将其Fedcoins转移到home链上，但是攻击者无法窃取任何东西。因此，失败模式的危害较小，并且在进行攻击时不会给攻击者带来任何好处。

另一个重要的考虑因素是所讨论的账本环境的类型。如果本地账本是**联盟链**，那么要采取紧急的措施（例如补救硬分叉，冻结攻击者的资产）来减轻损失会更容易，因此总损失可能会少得多。如果使用的是公链的话，则损失可能取决于其他因素，包括资产本身的“无许可”程度。但是，没有任何一种技术可以阻止攻击者完全获利，因为攻击者可能能够以比其他管理员检测到并阻止盗窃所能采取的更快的速度将资金兑换成其他司法管辖区的资产，甚至是无许可的加密货币。

请注意，在正常的假设下，从单链到具有跨链可移植资产的模型，除了增加了异步性之外，对流程几乎没有什么改变，但是在故障模式下，它以一种微妙的方式削弱了安全模型：**还原攻击成为盗窃漏洞**。这类似于以状态通道技术更改安全模型的方式，尽管在这种情况下采取的方法略有不同：检查攻击变成了盗窃漏洞。这在公链环境中具有特别的意义：为什么矿工不串通51％攻击

自己的链条的一个普遍论点是，他们无法从破坏支持他们的生态系统中受益。但是，在跨链资产流动面前，则完全没有这个约束，完全有可能提出借口，说明为什么攻击一个特定的“侧链”实际上对主链有所帮助。因此，侧链会发现它们本身与矿工之间存在很不安全的政治关系，这些矿工属于这些侧链的资产所存在的home链上。

Interoperability and Lifecycle Events（跨链与生命周期事件）

生命周期事件是一类在联盟链环境中特别令人感兴趣分析的事件。 这包括：

1. 经济边界的变化（例如英国退欧）
2. 实施和取消制裁（例如针对俄罗斯和伊朗的制裁）
3. 货币不再存在（例如，1999年引入欧元）
4. “分叉”货币（例如伊拉克第纳尔）

这可能导致几种事件：

1. 由于锚定的货币（一种特定的货币），区块链逐渐消失了。
2. 区块链系统开始容纳新资产
3. 与ETH/ETC类似的分叉的链（以这种方式从字面上将资产分成两部分，仅在分拆的分叉中具有先例主流融资，例如在Ebay和Paypal的情况下，但它很可能最终被用于分裂方案中的货币）

4. 由于制裁，两个链之间的某些种类的跨链操作被阻止了。还有一种“事实上的软制裁”的可能性，在这种情况下，由于法律的变更，特定种类的跨链操作突然承担了更高的合规负担。

在类似于1999年欧元引入方案的情况下应格外小心：最好避免的方案是创建一种新资产，该资产不具有单个home ledger，而是具有多个在部分余额中具有权威性的链。可以说，可以避免与此类场景相关的复杂性是区块链技术整体优势的很大一部分。一种可能的技术引入方式是保留旧货币的Home ledger,但从属于新的家庭分类账（即，在我们之前的Fedcoin示例中，其状态与“链A”相同）,并将余额简单地乘以当地汇率。

在某些方面，防止出于监管原因的跨链操作更加困难； 原因是任何人都可以为另一个链中的一个链创建验证程序，并且哈希锁甚至更容易。 甚至有可能在看起来像数学上复杂的金融衍生工具的东西内部隐藏一个哈希锁。 但是，仍然可以采取明显的行动：例如，如果国家A的中央银行希望冻结国家B对其货币的访问权限，那么它可以简单地冻结与该国任何链相关的余额。 也可以采用较软的措施，例如将**跨链索赔限制为每人每年$ X**。通常，可以实施对资产从一个链到另一个链的几乎任意控制； 但是，防止同一链上的资产之间的交易更具挑战性，可能只能通过类似于当今用于处理欺诈和结构化的启发式模式检测技术来解决。

在所有这些情况下，一个非常重要的目标是，无论发生什么意外事件，链上应用程序（包括链上合同）的转换都应尽可能地平稳。 这可以通过以下两种方式来实现：预先通知（使用最大程度的“清洁”技术来实施更改）以及在链中任何财务或其他合同中使用适当的缓解和恢复技术。 从技术上讲，混乱的分裂，冻结和其他黑天鹅只应在战争等罕见的紧急情况下发生（或可能进行低级政治报复，一个可能的例子是政府在迅速分裂的情况下破坏了对该地区的金融准入），在这种情况下，应避免损害和合同一级的恢复技术可能是缓解风险的唯一形式。

缓解和从故障模式中恢复

从经验中我们知道，永远不会遭受攻击或治理事件的，持久，不变和不变的协议的概念是不切实际的幻想。 需要改变，优先级改变，意外发生，并且实现本身常常包含错误。 如果由比特币社区中的许多人认为，比特币核心C ++代码确实是比特币协议的规范，那么，通过整数溢出攻击凭空创造930亿个BTC的攻击者绝对不应该如此。 通过协调的软分叉否认了他应得的利益，该分叉最终恢复了12小时的历史记录并消除了该漏洞。 鉴于这些不可避免的现实，我们如何管理风险？

一种可能的对策是将交互限制在**具有相似治理策略和共识机制的链之间**，或者至少是相互信任的治理策略和共识机制之间。 出于这个原因，有些人更喜欢留在私链和联盟链的领域，将“无政府状态”的公链看作是传统金融

的水源。 但是，不同的联盟链治理机制可能最终会彼此意见分歧，甚至试图主动地互相攻击。 故意防止多链交互操作的尝试以财务制裁的形式开创先例，例如。 那些反对俄罗斯和伊朗的人。 一个国家/地区的政府可能会企图在其大部分节点位于该国的情况下指挥一个区块链系统，并对试图将其作为网络战形式与该区块链系统对接的敌对国家/地区的服务进行攻击。 因此，如果我们想拥有一个最大程度地适用于全球的模型，则该模型中的每个链仅信任自己，并假定来自其他链的任意行为的可能性，即。 每个链条都是“无政府状态”的地方似乎是一个合理的起点； 毕竟，将地缘政治本身描述为无政府状态并不是完全错误的。 因此，我们想应对链条失效的可能性。

此时，您可能会问，如果链可能会失败，那么分析链的互操作性和分析集中式系统之间的互操作性有什么区别？ 区块链技术带入了什么讨论，这些讨论产生了根本上新的考虑因素，例如SWIFT与国家银行系统之间不存在的考虑因素？ 一个论点是，我们所谓的“区块链技术”或“ crypto 2.0”实际上是一系列相关技术和理念的集合，包括数字签名，哈希，Merkle树，对等网络和通用概念。 “不信任第一”和“推挤而不是拉扯”，而这些哲学选择加在一起的结果是，即使在各种形式的51％攻击的情况下，区块链环境也呈现出我们可以按顺序利用的独特属性。 设计更安全的系统。

从技术上讲，我们可以从一个特定的属性开始讨论：区块链在其处理的每笔交易和其进行的每个状态更改中都留下一个基于散列和基于签名的**可加密验证轨迹**（Merkle树根）。 即使发生了51％的攻击，旧的链仍然存在，并且在密码学上-不仅在经济上或社会上是加密的，而且在受到硬数学保护的意义上在密码学上-是不可变的。 攻击确实发生的事实可以被所有人看到，而且如果查看区块链的给定参与者（或接力者）拥有自己的内存和自己的区块，它甚至可以看到哪个分支先出现。 如果一段时间内未创建任何新块，则也可以看到这一点。

这种情况的某些后果相当平凡，其中包括我们已经讨论的内容：仅由于在区块链中大量使用哈希和签名进行密码认证，我们才可以构建中继样式的互操作性机制以及诸如哈希锁。 在传统的集中式API领域中，这两种技术都遥不可及。 但是，检测故障模式的能力更加有趣。例如，假设我们想要一个事件onChainReverted-（chainID，k）。 如果中继器看到某个时刻，M块是链的头，那么该事件可以在工作证明上下文中自动触发，然后头切换到N块，其中N和M仅共享一个共同的祖先 彼此落后k代-k块还原。

另一个示例是onFailStop（chainID，s）事件，该事件在区块链**未能在s秒内**创建新块时触发。 要实现的更复杂的检查系统将是检查机制，任何人都可以提交交易，并且如果该交易看起来值得包含在一个区块中但没有被包含在例如30个区块之内，则该oracle会回答“yes”。协议硬分叉事件可以通过以下方式触发：自愿包

括对表示分叉的“标志合约”进行状态更改，从而允许自动触发onFork事件。 当然，针对上述所有问题的更集中解决方案是多重签名。

当应用程序收到这些消息之一时该怎么办？ 对于这种情况下系统正常执行会有一个限度。从根本上讲，不可能阻止攻击者通过反复恢复侧链来耗尽侧链，因此，无论采用哪种编程语言的计算机科学手段，打破上一节中假设的“链A”的攻击者都能够窃取M个虚拟货币Fedcoins。 尽管可以将智能合约与还原预案一起使用，以创建一种低信任度的“区块链故障保险” 。 但是，在其他情况下（例如fail-stop），在没有人需要承担任何风险的情况下，可以获得更好的性能。

一种策略很简单：让每个合同都任命一个“紧急策展人”，在这种情况下，策展人仅在fork oracle声称所讨论的合同正在与之相互作用的链之一经历不规范的情况下才获得对该合同的权力。 然后，该策展人可以根据其原始意图，单独尝试以其最大的能力执行合同。策展人可以是银行，监管机构，非政府组织，甚至本身可以是这三者的任何组合组成的联盟。 尽管策展人不会经常依靠，但这是一个需要信任的角色，因此，强烈建议选择一个值得信赖的机构来担任该角色。 该策略最容易编写代码，人类也很容易理解（包括理解为什么在正常情况下不引入其他集中化方法），因此很可能在短期内实现。

另一种方法是**尝试自动处理某些情况。** 例如，如果给定链遇到故障停止事件，则可以延长超时直到恢复。 有序的硬分叉可以在onFork事件参数中携带标志，这些标志指定侵入性的级别和类型，并且可以根据所讨论的类型采取措施。 在某些情况下，回退可能会起作用：如果链B上的合同从链A寻找身份信息，而链A不可用，则它可能会尝试从其他来源获取相同的信息，或者指定“回退KYC策展人”。在某些情况下（例如，跨链利息优惠券付款），您可以简单地将付款注册为失败，然后在一段时间后重试。

总而言之，我们可以看到多链交互为应用程序增加了一层全新的复杂性。这种复杂性带来了巨大的希望，尽管在增加了复杂性的情况下，跨链智能合约仍可能比当今管理系统间转移，合约和交换的流程在复杂性，脆弱性上要低得多。 但是，它也带来了自己独特的挑战，独特的风险，并且需要了解这些风险以及解决这些风险的缓解策略。 从长远来看，我们希望能够围绕这种跨链可编程语言的概念来设计互操作性研究，从而确保安全地实现互操作性的基本构建块所面临的挑战，以及利用此类功能并将其实现为互操作性所面临的挑战。 实际应用程序可以单独处理； 但是，我们仍处于起步阶段，还有很长的路要走。

多链交互的实践

	公证人	中继/侧链	哈希锁
互操作性类型	All	全部（如果两个链上都存在中继，否则仅因果关系）	仅交叉依赖
信任模型	多数公证人诚实	链不会失败或不会受到“ 51%的攻击”	链不会失败或不会受到“ 51%的攻击”
可用于跨链交易	是	是	是
可用于跨链资产的可移植性吗?	是（但需要普遍的长期公证信任）	是	否
可用于跨链预言机吗?	是	是	不能直接使用
可用于跨链资产负担吗?	是（但需要普遍的长期公证信任）	是	在很多情况下可以，但是有困难

	公证人	中继/侧链	哈希锁
实现的困难程度	中等	困难	简单

涉及链互操作性的用例是可能花费最长时间实现的少数几个案例，仅是因为依赖项集如此之大。 这种用例需要（i）足够成熟的链A，（ii）足够成熟的链B，以及（iii）某些应用或需求无法在单个区块链上通过实现来满足。（iii）只有在（i）和（ii）本身都具有相当成熟的需要互操作的应用程序时，才能满足要求。

因此，短期互操作性用例可能与公链加密货币交换以及可能的公链加密货币衍生产品有关。 诸如MakerDAO之类的项目可能对加密货币衍生品用例最感兴趣，因为他们有意打算基于许多区块链在多种资产中分散其持有的资产。 联盟链的用例将花费更长的时间，因为Liquid可能是唯一看到任何重要用途的联盟链，甚至其主要用例都完全依赖于公链加密货币。 在主流金融中使用联盟链还很遥远。 因此，在短期内，**公链加密货币金融系统**将很可能成为跨链互操作性机制的主要测试平台，一旦该技术可以应用于其他用例，则为该技术的成熟提供了充足的时间。

从长远来看，构建可互操作的应用程序应在响应实际需求的基础上完成； 虽然肯定可以抽象地开发基础技术，但是尝试构建“将链条连接在一起”的应用程序而没有明确的基于用例驱动的理由说明为什么这样做有用的做法可能会浪费资源。 同时，很明显，公证人，中继和哈

希锁定方案是开始在其之上构建的优秀构建块，下一步可能是继续开发这些工具，同时在一些简单的示例测试用例中证明它们的适用性。 包括付款与付款，交付与付款，资产可移植性和跨链Oracle。 如果该技术被构建为具有足够的通用性，那么随着时间的推移，它在最终成为最佳应用程序中的适用性将自然而然地显现出来。