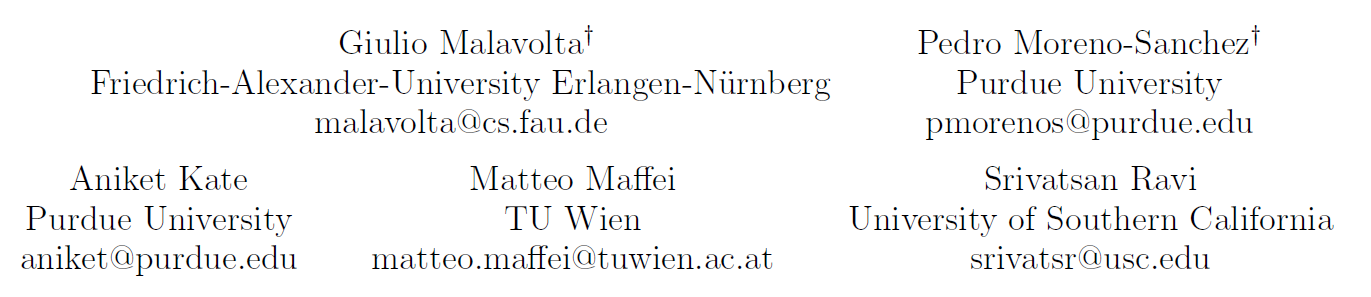
**关于支付通道网络的并行性和隐私安全性的研究**



# 摘要

无需许可的区块链协议，例如比特币，在交易量和交易延迟方面有着固有的限制。当前解决这个关键问题的努力大多集中于研究非死锁的支付通道。这类通道和支付通道网络（PCN）结合，从而能够使无数的、除了每个通道最初的跟最末尾部分的支付记录无需访问使用区块链。尽管这种方法为低延迟、大处理量的支付方式铺设出了一条大道，它在实际使用中却引起了人们对隐私问题的担忧，同时也面临着协议并行的支付本质技术到目前为止未被充分研究的问题。

在这篇论文中，我们为PCN的隐私与并行性问题的处理奠定基础，在普通可组合的框架和实践、理论上均被证明安全的方法这些方面正式给出了一些定义。特别地，我们研究出了Fulgor和Rayo。Fulgor是第一个PCN的支付协议，它提供了可被证明安全的PCN隐私保障，并且和比特币的脚本语言系统完全兼容。但是，Fulgor是一个封闭的协议因而在强调即时性的PCN中有可能会发生对并行交易的死锁现象。与之不同的Rayo是第一个能够拜托死锁的PCN协议。然而一个新的让人不愿接受的实验结果表明，摆脱死锁的技术必须要以减弱隐私性为代价。Fulgor和Rayo的核心是Multi-Hop HTLC，一个新型的智能合约。它与比特币的脚本语言系统兼容，提供虚拟的交易信息并且拥有比先前方法更好的响应速度和并行性。我们对Fulgor和Rayo的表现测验表明一条包含10名中间用户的交易信息只需花5秒就能处理完成，因而这也证明了该技术付诸实践的可行性。

**关键词**：区块链 比特币 Fulgor Rayo

目录

[摘要 1](#_Toc503117894)

[1 简介 3](#_Toc503117895)

[1.1 隐私方面的挑战 3](#_Toc503117896)

[1.2 并行性的挑战 4](#_Toc503117897)

[1.3 我们的贡献 4](#_Toc503117898)

[2 背景 5](#_Toc503117899)

[2.1 支付渠道 5](#_Toc503117900)

[2.2支付渠道网络](#_Toc503117901) 6

[2.3 最高水平的PCN技术 6](#_Toc503117902)

[2.3.1 PCN的路由 6](#_Toc503117903)

[2.3.2PCN上的交易 6](#_Toc503117904)

[3、问题探讨 8](#_Toc503117905)

[4、Fulgor：我们的成果 8](#_Toc503117906)

[4.1建立区块](#_Toc503117907) 8

[4.2多跳HTLC技术](#_Toc503117908) 9

[4.3构建细节](#_Toc503117909) 9

[4.4安全和隐私分析](#_Toc503117910) 8

[4.5对于系统的讨论 9](#_Toc503117911)

[5 无死锁的PCN交易 9](#_Toc503117912)

[5.1并行 VS 隐私 9](#_Toc503117913)

[5.2 理想中的功能 10](#_Toc503117914)

[5.3 Rayo：我们的成果 11](#_Toc503117915)

[5.4对系统的探讨 11](#_Toc503117916)

[5.5 Fulgor vs Rayo 11](#_Toc503117917)

[6 表现分析 12](#_Toc503117918)

[7 相关工作 13](#_Toc503117919)

[8 结论 14](#_Toc503117920)

[9 致谢 14](#_Toc503117921)

[参考文献 15](#_Toc503117922)

# 1 简介

比特币[57]是一种去中心化的网络数字虚拟货币，如今它被人广泛接受并成为了一种可供人们选择的货币支付系统。与把所有交易信息记录在某个值得信任的金融机构的线下总账这种方式不同，通过这种方式的交易信息会被记录在比特币区块链中。这里所说的比特币区块链是一个保存在世界各地很多并不互相信任的使用者中的数据库，这些使用者通过基于工作量证明的某种一致算法对数据库中的信息予以更新。然而，这种一致算法本身的无许可性将交易速率限制在了每秒钟10次，而其他的网络交易手段，例如Visa，支持每秒处理最多47000笔左右的交易[18]。

考虑到比特币用户的不断增长以及比特币交易对他们的重要程度,如今比特币的延展性被认为是构建比特币社会的一个重要担忧点[67,3]。许多研究课题和产业正致力于解决这个重要的难题[3,62,60,32,4,2]。

曾经为了研究非死锁的支付方式而产生的比特币正在蓬勃发展。此时，对于找到克服比特币延展性难题的承诺也被做了出来。在小的范围里，一些人开辟了一种支付渠道，把单一的交易信息添加到区块链中，这类区块链正是他们的比特币通过只能协议被安全储存的地方。然后，可以在本地就存款余额的重新新分配达成一致，执行几次线下支付。最后，共享支付通道的用户执行另一个比特币交易的时候，在区块链中添加最终余额，就能有效地关闭支付通道。

以这种方式，区块链被要求打开和关闭一个支付通道，但是这不是为用户之间的任何支付，因而可以减少区块链的负载并增加交易处理量。然而，这种简单的方法仅限于两个共享一个开放通道的用户之间的直接支付。有趣的是,创建一个从发起者到接收者的拥有充足空间的公开交易方式在原则上讲是行得通的，据此可以有效地创建一个网络支付渠道（PCN）[60]。

要使PCN满足越来越多的使用者以及越来越多的交易，许多难点还需要被克服。特别的，今天我们从类似的支付系统(如信用网络[36,37,15,17])了解到，一个成熟的PCN必须为若干问题提供解决方案，如流动性[29,55]，网络生成[30]，路由可伸缩性[61,71]，并行[49]，以及隐私[56,54,49]等。

比特币社区已经开始认识到这些挑战[47、41、40、48、43、22、3、67]。然而，目前的PCN技术仍然不成熟，这些挑战需要深入研究。在这篇论文中，我们为隐私和并行性的PCNs奠定了基础。有趣的是，我们证明了这两个性质是相互联系的，它们之间存在着一种内在的平衡。

## 1.1 隐私方面的挑战

现在看来，这种支付渠道似乎必然会改善比特币支付的隐私性，因为他们不再登录区块链。然而，这种普遍的想法已经开始受到社区的质疑，因为目前还不清楚PCN是否能够提供足够的隐私保证[68,22,43]。最近的一些研究[47,41,40]提出了hub支付网络的隐私保护协议，所有用户通过一个独特的中介进行线上支付。不幸的是，目前还不清楚如何将这些解决方案拓展到Multi-Hop PCN上。

目前，人们已经为定义一个成熟的PCN做出了一些努力[19,60,13,10]。在它们之中，闪电网络[60]已成为比特币社区中最突出的PCN[1]。然而，它目前的操作并没有提供对PCN的所有隐私保证。例如，通过支付路径进行路由的最大可能值的计算要求中级用户显示其支付通道的当前容量给发件人[62，第3.6条]，从而泄露敏感信息。此外,闪电网络中使用的比特币智能合约执行原子性的更新付款渠道包括在付款路径,需要揭示一个共同的在路径中的每个用户可以使用中间用户获得他们支付的哈希值 [60]。事实上，虽然有大量的学术论文研究了当前比特币支付对比特币的隐私保护[52,25,66,45,64,21,51]，但目前还没有对PCNs提供的或需要的隐私保证进行严格的分析。对他们的协议、威胁模型和隐私概念缺乏严格的定义，阻碍了对正在进行的尝试进行正式的安全和隐私分析，更不用说开发可保护的安全和隐私保护的解决方案了。

## 1.2 并行性的挑战

一致性算法，例如，在比特币上的工作量证明，简化了同时链支付的串行化。被允许在给定的时间内访问所有并行支付的矿工可以很容易地按照一组预定义的规则(例如，按支付费用排序)将它们序列化，然后将它们添加到区块链。然而，这已不再是PCN上的情况:大量的非限制支付没有添加到区块链，并且在协商一致时不能序列化。更重要的是，单个用户不能很容易地避免并行性问题，因为支付可能会涉及到除了支付人和收款人之外的其他几个用户。

在当前的PCNs(如闪电网络)中，一旦支付通道没有足够的容量(可能会同时为另一个线上支付)，就会中止支付。然而，这导致了死锁现象下，没有一个线上在处理的支付终止。综上所述，虽然当当前的PCNs扩展到大量的用户和非限制支付时，并行付款很可能会发生，但是固有的并行性问题还没有被彻底的研究过。

## 1.3 我们的贡献

这篇论文的贡献如下:

首先，我们首次正式确定PCN的安全性和隐私性概念，即平衡安全性、价值隐私和发送者/接收者匿名性，它们遵循通用可组合性(UC)的框架[27]。

其次，我们第一次研究了PCNs中的并行问题，并提出了两种协议——Fulgor和Rayo，通过不同的策略来解决这个问题。Fulgor是一个阻塞协议，它与一些类似的支付网络(如信用网络)提出的并行性解决方案(如信用网络)提出，这可能导致死锁，而并行付款没有通过。克服这一挑战，Rayo是第一个保证非阻塞进程的PCNs协议[42,20]。在这样做时，Rayo确保至少有一个并行付款终止。

第三，我们描述了在PCNs中隐私和并行之间的一个令人惊讶的权衡。特别地，我们证明了任何一个PCN都不可避免地减少了发送者和接收者的匿名性，从而削弱了隐私保障。

第四，我们正式描述了多跳哈希时间锁定协议(multi - hop HTLC)，这是一份位于Fulgor和Rayo核心的智能协议，与闪电网络形成对比的是，即使用户在付款路径上从payer到payee，也能保证隐私的属性。我们正式定义种HTLC协议和提供一个有效的基于最近提出的零起点证明系统的实例化ZK-Boo[38],[69]改善先前建议减少所需的数据从650 MB到17 MB,检定装置的运行时间从600毫秒到309毫秒和验证的运行时间从500毫秒到130毫秒。此外，多跳HTLC不需要对当前的比特币脚本系统进行修改，因此可以无缝地部署在当前的PCNs中，从而实现独立的兴趣。

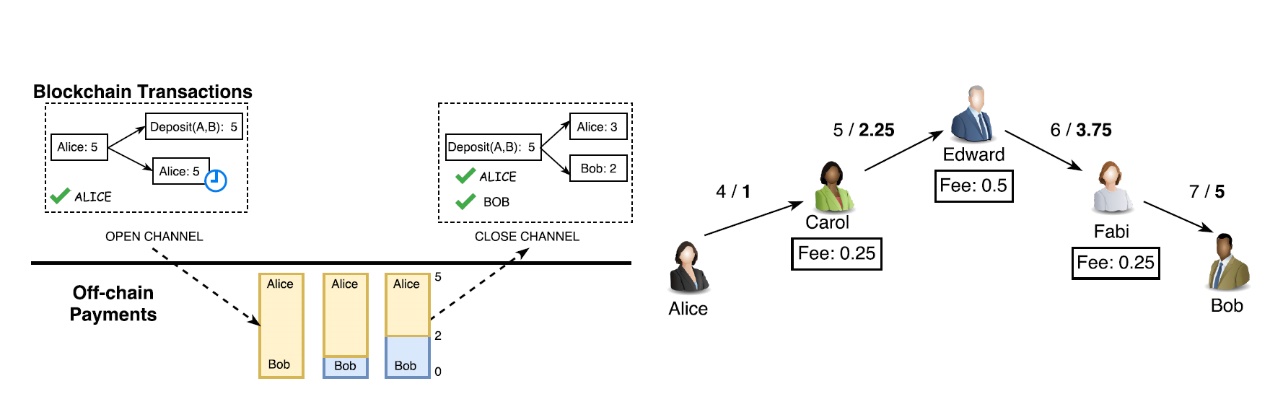
最后，我们在Python中实现了Fulgor和Rayo的原型，并对运行时间和通信成本进行了评估。我们的结果显示，在一个有10个中级用户的路径中，一个隐私保护的支付可以在5秒内完成，并且在17 MB的通信开销上产生。这表明我们对PCN的协议与其他保护隐私的支付系统是一致的[54,49]。此外，我们的评估显示，Fulgor和Rayo可以扩展以满足越来越多用户的需求，这些用户的开销相当小，可以通过优化的实现进一步减少组成部分。第二部分概述了所需的背景。第3节定义了我们在这个工作中处理的问题，并概述了我们关于PCNs的隐私保护解决方案Fulgor和Rayo。第4节详细介绍了Fulgor协议。第5节描述了我们对PCNs并行性的研究，并详细介绍了Rayo协议。第6节描述了我们的实验和评估结果。第7节讨论了相关的工作，第8节总结了本文。

# 2 背景

在这一节中，我们首先概述支付渠道的概念，然后描述网络支付渠道。

## 2.1 支付渠道

支付通道允许两个用户之间的几个比特币支付，而不需要对比特币区块链进行任何支付。支付渠道的基石是将比特币存入一个由两个用户控制的多签名地址，并保证所有比特币都在通道到期时，在双方约定的时间内返还。在下面的内容中，我们将概述支付渠道的基本内容，并将读者介绍给[60,32,50]以获取更多的细节。

在图1所示的演示示例中，Alice为Bob打开了一个支付通道，初始容量为5个比特币。这个打开的交易通道确保如果不使用支付通道，Alice在某个超时后取回钱。现在，Alice可以通过调整存款余额对Bob进行支付。每一次的连锁支付增加了Bob的余额并减少了Alice。当不需要更多的线下支付(或支付通道的容量耗尽)时，支付通道将关闭，并在区块链中包含结束交易。该交易将存入的比特币按支付渠道中最重要的余额向每个用户发送。

*图1:支付通道的说明示例。白实心盒表示比特币地址及其当前余额，虚线框代表比特币交易，时钟表示时间锁协议[7]，用户名旁边的符号表示她的签名来验证交易，彩色方框表示支付通道的状态。虚线箭头表示时间序列。爱丽丝第一次存入5个比特币，与Bob一起打开支付通道，然后用它在线下支付Bob。最后，支付通道以最近的余额数据关闭。*

*图2:一个PCN支付的例子。非粗体(粗体)数字表示从Alice到Bob的支付的通道的容量。爱丽丝想通过卡罗、爱德华和法比支付2比特币给鲍勃。因此，她用3个比特币(即:，支付金额+费用)打开了支付通道*

图1中描述的支付通道是单向通道的示例:它只能用于从Alice到Bob的支付。为了克服这一限制，我们定义了双向通道，因为在两个方向上都可能发生连锁支付。双向支付渠道本质上是单向的[[1]](#footnote-1)。主要的技术挑战在于改变航道的方向。在运行的示例中，假设当前支付通道余额*bal*是{Alice:4,Bob:1}，并进一步假设Bob将一个比特币支付给Alice。新的支付通道余额*bal’*是{Alice:5,Bob:0}。此时，Alice受益于*bal’*，而Bob则受益于*bal*。这种矛盾的解决方案是让Bob和Alice都承认之前的任何余额都是无效的。不同的“失效”技术已经被提出，我们将之后告诉给读者[60,32,65]。

比特币协议最近已经更新，完全支持支付渠道。特别是，在最近采用隔离证人的比特币协议中，交易可延展性[8]，以及一系列有趣的新功能被加入到比特币协议中[16]。这一事件为今天[70]的主要比特币区块链的PCNs的实现和测试铺平了道路。

## 2.3 最高水平的PCN技术

支付渠道[41,47,32]和PCNs[50]的概念已经引起了研究界的关注。在实践中，PCN在比特币中有几个正在进行的实现方案[11,12,10,19]。其中，闪电网络已经成为比特币社区中最突出的例子，alpha实现方案最近已经发布[1]。PCN的想法是为了提高可伸缩性问题，这不仅是在比特币方面，还在其他基于区块的支付系统上，如Ethereum[13]。

### 2.3.1 PCN的路由

在PCNs中，一个重要的任务是找到在发送方和接收方之间有足够容量的路径。在我们的设置中，每个用户都知道网络拓扑。这是由于每个支付通道的打开都记录在公共可用的区块链中。此外，用户之间的八卦协议可以用来传播任何支付渠道的存在[62]。此外，每个用户收取的费用也可以通过类似的方式公开。在这些条件下，发送方可以在本地计算发送方和接收方之间的路径。在本文的其余部分，我们假设发送者根据自己的标准选择路径。然而，我们认为路径选择是一个有趣的但正交的问题。

### 2.3.2PCN上的交易

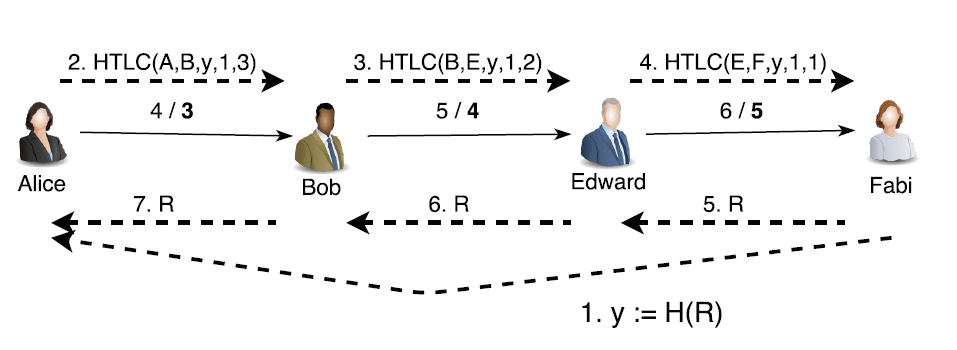
交易通道的付款方式是通过根据支付金额和相关费用更新路径中的每个支付通道的容量(参见2.2节)。这样的操作增加了原子性的重要挑战:要么更新路径中的所有通道的容量，要么没有更改通道。允许更改路径中的一些通道会导致用户丢失比特币(例如，用户可以向路径中的下一个用户支付一定的比特币，但不会从之前的邻居那里收到相应的比特币)。

闪电网络的当前提议包括一个名为哈希时间锁(HTLC)的智能协议[60]。这个协议锁住了x比特币，只有在协议履行时才能释放。根据一个哈希值y:= H(R)，R被随机选择，比特币x的数量和一个超时t，如下所示:

HTLC(Alice,Bob,y,x,t):

1、如果Bob在t日之前生成R \*，从而算出H(R \*)= y, Alice支付Bob x比特币。

2、如果t天过去了，Bob未算出R \*，爱丽丝取回x比特币。



*图3:用HTLC协议让Alice支付给Fabi一个比特币的例子。*

首先，条件是从Fabi发送到Alice。然后，该条件被转发给在每个支付通道中持有1比特币的路径的用户。最后，接收方显示R，在每个支付通道中释放被持有的比特币。为了简单起见，我们假设本例中没有支付费用。

第一，支付金额(即:(1比特币)从发件人到接收者，然后从接收者释放到发件人。在更详细的情况下，在接收方(Fabi)将条件发送给发送方(Alice)之后，Alice设置了一个HTLC与她的邻居，有效地设置了保存的支付值。这样的HTLC在接收方路径的每个支付通道上设置。此时，接收方知道每个支付通道上的支付值都是固定的，因此她会显示出R的值，这样她就可以完成协议，并在路径的每个支付通道中处理新的容量。

需要注意的是，路径中的每个用户都将HTLC设置为输出的支付通道，其超时时间小于传入支付通道的HTLC。以这种方式，用户可以确保在她的比特币被从她的继任者拉下之后，她就能从她的前任身上提取比特币。离线用户可以外包与她的支付通道相关的开放HTLC协议的实现的监控[33]。

虽然HTLC与比特币完全兼容，但它在实践中的使用导致了重要的隐私泄露:很容易看到H(R)的值，也能靠这个独一无二的值分辨出参与此交易的用户。这一事实有两大影响。首先，路径上的任何两个串连用户都可以从三个方面推导出他们参与相同支付的事实，这可以用来重构发送者和接收者的身份。第二，如果HTLC语句被上传到区块链(例如，由于支付路径中没有协作的中间用户)，观察者可以很容易地跟踪用于支付支付的完整路径，即使她不是支付的一部分。在这一工作中，我们提出了一个新的多跳HTLC智能协议，避免了这个隐私问题，同时确保没有中间用户丢失她的比特币。

在当前的PCNs中，一个重要的问题是处理并行支付，这需要在他们的路径中共享支付通道。如果仅在路径上的共享支付通道的余额不足，当前的建议就会中止支付。然而，正如我们在第3.3节中展示的那样，这种方法可能导致死锁情况，在这种情况下，没有同时支付终止。我们提出了一种支付协议，确保非阻塞进程，即至少一个并行支付终止。此外，我们还展示了任何完全分布式支付网络的并行性和隐私性之间的内在权衡。

# 3、问题探讨

# 4、Fulgor：我们的成果

在本节中，我们将介绍我们所构建的必需的加密模块(4.1节)，我们描述了多跳HTLC协议的细节(第4.2节)，我们详细介绍了PCN操作的结构(第4.3节)，分析了它的安全性和隐私(第4.4节)，并给出了一些备注(第4.5节)。

**表示方法**

我们认为系统的安全参数可以用A级表示，我们用标准定义来表示可忽略的函数。我们通过在支付渠道中决定可能发生的事件来表示。决策转发信号在支付通道中锁定与支付值对应的余额。由于支付的流产，中止在支付通道中释放锁定资金的信号的决定也会流产。相应的，当用户接受交易，那么信号也会被决定接受。

为了便于标记，我们假设用户标识符(Ui,Ui + 1)可以从通道标识符C < ui,ui+1 >中提取。

**系统的假设**

我们假设PCN中的每个用户都知道完整的网络拓扑结构，即所有用户的集合和每对用户之间存在一个支付通道。我们进一步假设，付款人根据她自己的标准选择付款路径到接收方。每个支付通道上的当前值并没有被发布，而是由用户共享一个支付通道，否则隐私就会被破坏。我们进一步假设每个用户都知道PCN中每个用户收取的支付费用。

这可以在实践中完成。两个用户之间的支付通道的打开需要在区块链中添加一个包含用户标识符的交易交易。因此，PCN的拓扑结构是可以泄漏的。此外，用于打开支付通道的交易可以包含用户定义的数据[5]，这样每个用户都可以嵌入自己的支付费用。通过这种方式，每个用户都可以主动地收集关于网络拓扑和来自区块链的费用的更新信息，或者通过一个八卦协议来传播(62,48)。

我们进一步假设，对共享支付通道的用户对通过安全且经过身份验证的通道进行通信(如TLS)，这很容易实现，因为每个用户都是由公钥唯一标识的。我们还假设发送方和接收方(可能是间接的)可以通过安全的直接通道进行通信。最后，我们假定支付的发送方可以与每个中间用户创建一个匿名支付通道。要到达每个用户的IP地址可以在通道创建交易中进行编码，因此可以在区块链中登录。我们注意到，我们的协议完全是关于路由的参数，因此任何类似于洋葱的路由技术都会在这方面起作用。

我们考虑了有界的同步通信设置[23]。在这样的通信模型中，时间被划分为固定的通信回合，并且假定用户在轮中发送的所有消息都可以在执行过程中有限的步骤中向预期的接收方提供。因此，没有消息表明在轮中没有来自用户的通信。在实践中，可以通过PCN[28]中的用户之间的松散同步时钟实现这一点。

最后，我们假设用户之间有一个总的顺序(例如，通过他们的公共验证键对字母进行排序)。

## 4.5对于系统的讨论

**与比特币的兼容性**

我们注意到，所有非标准的加密操作(比如NIZK证明)都发生在链外，而在验证区块链时需要执行的唯一算法是哈希函数H，它可以用SHA-256来计算。因此，我们的多跳HTLC方案和Fulgor作为一个整体与当前比特币脚本完全兼容。此外，如第2.1节所述，在比特币中加入了SegWit或类似的解决方案，使比特币系统的支付通道得以完全实现[70]。

**普遍性**

Fulgor通常是PCNs(不仅与比特币绑定)。Fulgor要求:(i)开放通道允许嵌入自定义数据(例如，费用);(二)支付渠道余额的有条件更新。由于任意数据可以包含在加密货币交易中[5]并且大多数PCNs支持HTLC协议，Fulgor可以用于许多其他的PCNs，比如Raiden，一个用于Ethereum的PCN[13]。

**支持双向通道**

Fulgor可以很容易地扩展，以支持双向支付通道，并且只需要两个较小的更改。首先，支付信息必须包括每个支付渠道所要求的方向。第二，通道C < uL，uR >的容量是一个(L,R,T)的多元函数，其中L为uL的当前余额，R是uR的当前余额，T是信道的总容量。如果L > v和R + v < T，则可以从左向右支付v的值。在这种情况下，元组被更新为(L - v,R + v,T)。从右到左的付款也相应处理。

# 5 无死锁的PCN交易

在本节中，我们将讨论如何以无死锁的方式处理并行支付。换句话说，如何保证至少有一组并行付款终止。

下面我们从一个不可能的结果开始，它决定了我们的无死锁支付协议Rayo的设计。然后，我们描述了理想世界功能和Fulgor实现它们所需的修改。最后，我们讨论了这些修改对隐私属性的影响。

## 5.1并行 VS 隐私

我们表明，实现无死锁的成果需要一块与每个支付相关联的全球性的基石。具体地说，我们不能提供对PCNs的分离访问并行性和非阻塞进程。形式上，PCN实现是分离访问并行的，对于任何两个支付通道ei、ej、通道状态(ei)和通道状态(ej)的交集为空集。

引理2。对于提供无死锁进程的支付通道问题，不存在严格可序列化的并行实现方法。

我们遵从附录D的证明草图。在确定了并行和隐私的固有成本之后，我们通过附加到每个支付的Txid字段对全局状态进行建模。但是，我们注意到，这个Txid允许对手减少支付的可能的发送者和接收者，因此不可避免地减少隐私保证，正如我们在5.2节中讨论的那样。

## 5.2 理想中的功能

在这里，我们将展示如何修改第3.2节中描述的理想功能F，以解释在任何PCN中实现非阻塞进程的更改。首先,单一标识符Txid(相对于独立采样hi)用于所有的付款通道路径(c < u0,u1 >,…,c <un,un+1>)。第二，当支付通道中没有剩余的容量时，F不再中止支付。相反，如果它的Txid高于当前的支付，或者中止它的Txid更低，则F将对其进行排队。我们在附录C中详细介绍了修改后的理想功能。

**相关讨论**

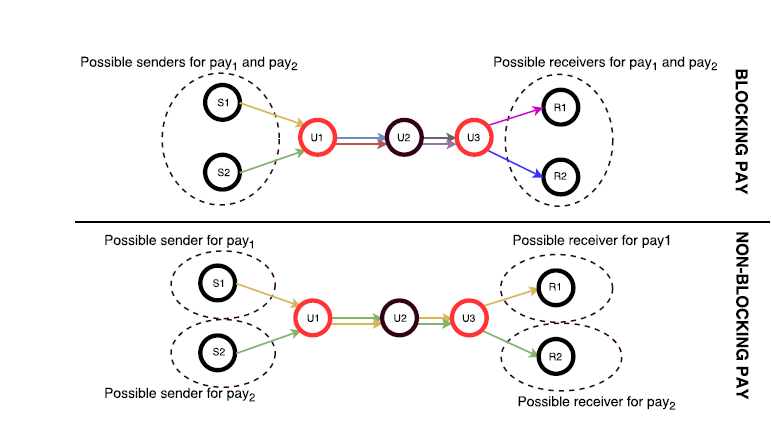
在这里，我们将讨论修改后的理想世界定义如何捕获第3.1节所述的安全和隐私概念。特别地，我们很容易看到，平衡安全和价值隐私的概念是沿着同样的路线执行的。然而，在支付路径中所有中间用户之间的相同支付标识符的泄漏，减少了可能的发送者和接收者对实际发送方和接收方的支付，从而破坏了关系的匿名性。因此，在如何处理并行支付(死锁或非死锁)和匿名保证之间存在一个固有的折衷。

图9显示了这个折衷的一个例子。它显示了两个同时处理交易pay1((c<S1,U1 > ,c<U1,U2 > c<U2,U3>,c<U3,R1>),v)

和pay2((c<S2,U1 > ,c<U1,U2 > c<U2,U3>,c<U3,R2>),v)的方式的隐私性区别在于它是否可能死锁。我们假设这两笔付款都可以在当前的PCN上成功完成，并且这两笔付款都转移了相同的支付金额v，因为其他关系的匿名性是微不足道的。

对于阻塞支付的情况，每个中间用户Uj会观察每个支付payi独立选择的标识符Txidij。因此，攻击者不能将(Txid11,Txid21)和 (Txid13,Txid23)联系起来。它遵循由任何节点发出的支付操作，例如S1，敌方观察到的可能接收者的集合是{R1,R2}。

然而，当并行支付以非死锁方式处理时，对手会观察到在pay1中Txid11 = Txid13。因此，对手可以很容易地推导出，由S1发起的支付的唯一可能的接收者是R1。



*图9:在并行和隐私之间折衷的例子。每个节点代表一个用户:黑色的节点是诚实的，红色是复杂的。在这两种情况下，我们假设两个并行支付:S1和S2通过路径U1、U2和U3分别支付给R1和R2。箭头的颜色表示支付标识符。虚线椭圆表示每种情况的匿名性。*

## 5.3 Rayo：我们的成果

**建立区块**

我们选择了与4.1节和4.2节中描述的相同的构件块。唯一的区别是，两个用户之间的通道状态现在被定义为通道状态:=(curK，«j)n,Q(ui,uj)n,cap(ui,uj .))，在这里，cur表示当前使用(部分)支付通道可用容量的支付的数组;Q表示在支付通道等待足够容量的支付的数组，cap表示支付通道的当前容量值。

**操作方法**

打开通道和关闭通道操作仍然如第4.3节所述。然而，为了确保非死锁的支付，必须增加支付操作。我们已经描述了图7和图8中浅蓝伪代码的附加操作。

在下面的内容中，我们非正式地描述了支付操作所需的这些传统操作。简单地说，当支付到达路径中的中间用户时，可以触发几个事件。最简单的情况是当相应的支付通道还未饱和时(即剩下足够的通道保证交易的成功)。用户接受支付，并简单地将其信息存储在cur中作为一种实时支付。

更有趣的情况发生在支付通道饱和时。这意味着(可能几次)付款已经通过支付渠道。在这种情况下，最简单的解决方案是中止新的支付，但是这会导致死锁情况。相反，我们确保死锁不会通过使用支付标识符的总顺序来发生:如果新的支付标识符(Txid)高于支付通道中当前活动的任何支付标识符(即:，包括在cur[]中)，由Txid标识的支付会以这种方式存储在Q中，如果当前的活动支付被中止，则可以从Q中恢复队列支付(Txid \*)，并向接收方重新发送。另一方面，如果Txid低于当前活动支付的所有标识符，则Txid所标识的支付将直接中止，因为在PCN中有更高标识符的并行支付时，Txid无法完成支付。

## 5.4对系统的探讨

**系统的讨论**

Rayo与比特币兼容，普遍地适用于PCN，支持与Fulgor类似的双向支付通道。此外，Rayo协议提供了非阻塞的进展。具体地说，Rayo确保了某些支付在每次执行中都成功终止。从直观上看，这是因为任何两个相互冲突的支付都必须由它们各自的惟一标识符来排序:最高支付标识符是确定的，并在较低优先级支付中止时成功地终止。

## 5.5 Fulgor vs Rayo

在这篇论文中，我们描述了在此工作中所呈现的两个协议之间的权衡。如表1所示，这两种协议都保证了关键的安全性和正确性属性，如平衡安全性和可串行性。通过设计，Rayo是唯一确保非死锁的协议。最后，关于隐私，我们的目标是尽可能地达到最严密的隐私。然而，尽管这两种协议都保证了隐私的价值，但我们已经表明，不可能同时实现非阻塞的进程和匿名性。因此，Fulgor实现了强大的匿名性，而Rayo以削弱匿名保证的成本实现非阻塞

的进展。我们注意到，只有当中间节点没有被破坏时，Rayo才会提供关系匿名性。直观上，Rayo提供了这个(较弱的)隐私保证，因为它仍然使用多跳HTLC作为Fulgor。

|  |  |  |
| --- | --- | --- |
|  | Fulgor | Rayo |
| 资产安全 | ● | ● |
| 可串行性 | ● | ● |
| 能否防止死锁 | 🌕 | ● |
| 价值隐私 | ● | ● |
| 匿名性 | ● | ◐ |

表1

# 6 表现分析

在本节中，我们首先对Fulgor的性能进行评估。最后，我们描述了Rayo所需的开销。

我们已经开发了Python的概念验证，以评估Fulgor的性能。我们与lnd[1]的API进行交互，最近发布的闪电网络实现，我们使用listchannels来提取一个开放支付通道的当前容量，listusers从网络中的其他用户提取公钥列表，getinfo提取用户自己的公钥。我们用SHA-256实例化了散列函数。我们使用基于python的ZK-Boo[63]实现了多跳HTLC来创建零起点证明。我们将ZK-Boo设置为使用SHA-256 136次，达到了2-80的验证错误率，并有一个32字节的记录[69]。

**实现级优化**

在协议描述期间，我们假定发送方为每个中间用户创建了一个不同的匿名通信通道。然而，在我们的实现中，我们使用Sphinx[31]在发送方和接收方之间创建一个单一的匿名通信通道，中间节点是路径中的中间用户。Sphinx允许向每个中间用户发送所需的支付信息，同时为路径中的其他用户和路径的实际长度进行模糊处理，并填充所转发的数据。这种优化已经在比特币社区中进行了讨论，并在lnd[9]的当前版本中实现。

**临床实验**

我们已经模拟了5个用户，并创建了一个支付渠道的线性结构:用户i和用户i-1、用户i+1仅开通了一个交易的支付通道，用户0是发件人，用户4是支付操作的接收者。我们在一个独立的，配备了Intel Core i7 3.1 GHz处理器和2GB 内存的虚拟机中运行每个用户。这些机器在本地网络中连接，平均延迟时间为111.5毫秒。对于我们的实验，我们假设每个用户已经打开了相应的支付渠道，并在PCN中得到了每个用户的公共验证密钥。由于这是一次安装操作，我们在实验中没有说明。

**具体表现**

我们首先执行了Ind软件中可用的支付操作，它使用基于html的支付作为支付通道中的条件更新的契约。我们观察到，在一个有5个用户的路径上进行(非私有)支付操作需要609 ms，因此需要Fulgor。此外，Sdr必须运行SetupHTLC(n + 1)协议，从而增加其计算时间。此外，Sdr必须将与多跳HTLC协议对应的附加信息发送给每个中间用户，从而增加了通信的复杂性。

发送方需要309 ms来计算每个中间用户的证明πi。每个验证码大小为1.65 MB，最后，每个中间用户需要130 ms来验证πi。我们把重点放在零起点证明上，因为它们是最昂贵的操作。

因此，总计算开销为1.32秒(Ind支付和多跳HTLC)，总通信开销小于5mb(3个零起点证明加上每个中间用户的小尺寸值的元组(xi、yi、yi+1)。我们注意到之前的提案[69]需要大约10秒来计算一个零起点证明。相比之下，Fulgor的交易操作只需不到2秒的计算，在整个支付操作的过程中，在用户之间通信不到5 MB，这说明了Fulgor的实用性。

**延展性**

为了测试Fulgor中支付操作的延展性，我们研究了每个角色在支付中所需要的运行时间和通信开销。、发送方、接收方和中间用户)。在这里，我们考虑到Sphinx需要将转发的消息填充到最大路径长度。由于在实践中没有广泛的PCN，我们在测试中设置了最大路径长度为10，如Ripple credit network等类似支付网络所建议的[49]。

对于计算时间，发送者需要3.09秒为每个中间用户创建πi。但是，如果利用当前多核系统并行计算，计算出不同的πi，可以提高计算时间。每个中间用户需要130 ms，只需要检查与路径中的继任者和前任用户的支付渠道协议。最后，接收方只需要检查给定的值是否是给定的散列值的正确预映像，就会产生很少的时间。

对于通信开销，发送方必须创建一个包含10个知识证明的消息，以及与每个中间支付通道相关的其他几个字节。所以总的来说，发送者必须大约转发17MB。由于Sphinx需要在每个节点上填充消息，以确保匿名，每个中间用户必须转发相同大小的消息。

总之，这些结果显示，即使是没有优化的实现，10个中间用户的支付也只需要不到5秒，并且需要在每个中间用户上花费大约17MB的通信开销。因此,Fulgor诱发一个相对较小的开销而使在几秒钟内PCN中的任何两个用户之间的支付和有潜力与成长基地部署为PCN的用户执行支付,结果也符合其他隐私保护支付系统[54，49]。

**非死锁支付（Rayo）**

考虑到Fulgor和Rayo的相似定义，我们用对Fulgor的性能评价来对Rayo进行评估。此外，非死锁支付的要求中间用户获得当前的实时支付的列表以及等待可用容量的支付队列(Q)。这些数据结构的管理只要相当小的计算开销。此外，根据Rayo的规范，存储在这些数据结构中的消息的数量在路径的长度上是明显的线性的。具体地说，涉及到长度k的路径会引起大约(c - k)的复杂性，其中c是受并行冲突支付总数的限制。

# 7 相关工作

支付渠道首先由比特币社区引入[2]，自那以后，已经提出了一些前的紧张关系。Decker和Wat - tenhofer[32]描述了双向支付渠道[32]。Lind等[47]利用可信赖的平台模块使用支付通道，而不妨碍比特币的兼容性。然而，这些作品专注于一个单一的支付渠道，其支持PCNs的扩展仍然是一个公开的挑战。

TumbleBit[41]和Bolt[40]提出了连锁的基于路径的支付，同时在TumbleBit中实现了发送者/ re- ceiver的匿名性，并在Bolt中使用匿名支付。然而，这些方法仅限于单跳付款，而且还不清楚如何将这些方法扩展到通用的多跳PCNs，并提供由Fulgor和Rayo实现的隐私感兴趣的概念。

闪电网络[60]已成为比特币中最重要的一项提议。其他的PCNs如雷[19]和Eclair[10]为Ethereum的比特币和Raiden[13]正在被提议作为对闪电网络的轻微修改。然而，他们使用HTLC泄露了每个支付的常见标识符，从而减少了我们在这项工作中描述的匿名保证。此外，目前的提议缺乏对并行付款的非阻塞解决方案。相反，Fulgor和Rayo依靠多跳HTLC克服了HTLC的可链接性问题。它们提供了非阻塞进程和匿名之间的折衷。

最近的论文[54,49]提出了信用网络的隐私定义，一个支持类似于PCNs的多跳支付的支付系统。此外，对集中式[54]和分散式信用网络的隐私保护协议进行了描述[49]。然而，信用网络与PCNs的不同之处在于，它们不需要确保对底层区块链的可靠性。此需求减少了用于设计PCN的加密操作集。尽管如此，gor和Rayo还是提供了类似的隐私保证，即使是在这些限制条件下。

Miller等[53]提出了一种支付渠道的建设，以减少资金被锁定在中间支付渠道的时间(即:这是一个有趣的问题，但与我们的工作是正交的。此外，他们将多跳支付的构建形式化，成为一种理想的功能。然而，他们关注的是抵押品成本，不讨论隐私保证，并行支付仅以阻塞方式处理，而且他们的构建依赖于以太网络上可用的智能契约，这与当前的比特币脚本系统是不兼容的。

Towns提出[69]基于zk - UNK ks的HTLC协议的变更，以避免其在支付通道中的链接能力问题。然而，比特币社区由于其低效率而没有采用这种方法。在这个工作中，我们使用一个具有可验证安全性的正式协议来重新讨论这个解决方案，并基于zk - boo[38]提供一个有效的实例化。

# 8 结论

在全球协商一致协议中管理的无规则的区块链，在其他方面，在满足不断增长的用户和支付基础上的可伸缩性问题。一种新兴的克服这一挑战的方法是由PCNs和最近的努力在第一个alpha实现中派生出来的，比如[60]在比特币或者是在Ethereum的Raiden[13]中。然而，我们只是在表面上对流动性、网络形成、路由可伸缩性、并行性或隐私等诸多挑战进行彻底的研究。

在这篇论文中，我们为隐私和并行性的PCNs奠定了基础。特别是，我们在通用的可组合性框架中定义了两种操作模式，用于处理并行支付如何处理(阻塞与非阻塞)。我们为每个人提供了正式证明的实例化(Fulgor和Rayo)，提供了非阻塞进程和匿名之间的折衷。我们的评估结果表明，在实践中部署Fulgor和Rayo是可行的，并且可以扩展以满足越来越多的用户。

# 9 致谢

我们感谢匿名评论者的有益评论，以及Ivan Pryvalov提供了基于python的zk - boo的实现。

这篇论文得到了英特尔/ CE- RIAS研究助理的部分支持，并得到了国家科学基金会的资助。

这项研究是基于德国研究基金会(DFG)通过合作研究中心1223和巴伐利亚州纽伦堡技术学院(NCT)所支持的工作。NCT是由friedrich - alexanderen - university erlangen - Niirnberg(FAU)和Technische Hochschule Niirnberg Georg Simon Ohm(THN)开展的一项研究合作。

# 参考文献

1. Alpha release of the lightning network daemon. Blog entry. **http://lightning.community/** release/software/lnd/lightning/2017/ 01/10/lightning-network-daemon-alpha- release/.
2. Bitcoin wiki: Bitcoin contract. https://en. bitcoin.it/wiki/Contract.
3. Bitcoin wiki: Bitcoin scalability faq. https: //en.bitcoin.it/wiki/Scalability\_FAQ.
4. Bitcoin wiki: Block size limit contro­

versy. **https://en.bitcoin.it/wiki/Block\_** size\_limit\_controversy.

1. Bitcoin wiki: Op\_return. https://en.bitcoin.it/wiki/OP\_RETURN.
2. Bitcoin wiki: Payment channels. **https://en.**bitcoin.it/wiki/Payment\_channels.
3. Bitcoin wiki: Timelock. https://en. bitcoin.it/wiki/Timelock.
4. Bitcoin wiki:Transaction malleabil­ity. https://en.bitcoin.it/wiki/

Transaction\_Malleability.

1. Bolt#4:Onion routing protocols. **https://** github.com/lightningnetwork/lightning-rfc/blob/master/04-onion-routing.md.
2. Eclair implementation of the lightning net­work. https://github.com/ACINQ/eclair.
3. Lightning network daemon. Github im­plementation. https://github.com/LightningNetwork/lnd.
4. Lightning protocol reference implementation. Github implementation. https://github. com/ElementsProject/lightning.
5. Raiden network. Project’s website. http://raiden.network/.
6. Reliable transaction submission. Rip­ple protocol’s documentation. https://

ripple.com/build/reliable-transaction-submission/.

1. Ripple protocol. Project’s website. https:// ripple.com/.
2. Segregated witness adoption. Blog en­try. https://bitcoincore.org/en/segwit\_adoption/.
3. Stellar protocol. Project’s website. https://www.stellar.org/.
4. Stress test prepares visanet for the most wonderful time of the year. Blog entry. http://www.visa.com/blogarchives/ us/2013/10/10/stress-test-prepares- visanet-for-the-most-wonderful-time-of-the-year/index.html.
5. Thunder network. Project’s website. https://github.com/blockchain/thunder.
6. Alpern, B., and Schneider, F. B. Defining liveness. Inf. Process. Lett. 21, 4 (Oct. 1985), 181-185.
7. Androulaki, E.Karame,G.0.,Roeschlin, M., Scherer, T., and Cap- kun, S. Evaluating user privacy in bitcoin. Financial Cryptography and Data Security 2013.
8. Atlas, K. The inevitability of pri­vacy in lightning networks. Blog entry. https://www.kristovatlas.com/the-inevitability-of-privacy-in-lightning-networks/.
9. Attiya, H., and Welch, J. Distributed Computing. Fundamentals, Simulations, and Advanced Topics. John Wiley & Sons, 2004.
10. Backes, M., Kate, A., Manoharan, P., Meiser, S., and Mohammadi, E. Anoa: A framework for analyzing anonymous communi­cation protocols. In IEEE 26th Computer Se­curity Foundations Symposium (2013).
11. Barber, S., Boyen, X., Shi, E., and Uzun,
12. Bitter to better. how to make Bitcoin a better currency. Financial Cryptography and Data Security 2012.
13. Camenisch, J., and Lysyanskaya, A. A formal treatment of onion routing. In Advances in Cryptology—CRYPTO (2005).
14. Canetti, R. Universally composable security: A new paradigm for cryptographic protocols. In ”FOCS,01”.
15. Cristian, F., Aghili, H., and Strong, H. R. Approximate clock synchronization de­spite omission and performance faults and pro- cessorjoins. In Proceedings of the 16th Interna­tional Symposium on Fault-Tolerant Comput­ing (July 1986).
16. Dandekar, P., Goel, A., Govindan, R., and Post, I. Liquidity in credit networks: a little trust goes a long way. In ACM Conference on Electronic Commerce (2011).
17. Dandekar, P., Goel, A., Wellman, M. P., and Wiedenbeck, B. Strategic for­mation of credit networks. In WWW (2012).
18. Danezis, G., and Goldberg, I. Sphinx: A compact and provably secure mix format. In 30th IEEE Symposium on Security and Privacy (S&P 2009).
19. Decker, C., and Wattenhofer, R. A fast and scalable payment network with bit- coin duplex micropayment channels. In Sta­bilization, Safety, and Security of Distributed Systems (2015).
20. Dryja, T.Unlinkable outsourced channel monitoring. (Talk transcript) https://diyhpl.us/wiki/transcripts/ scalingbitcoin/milan/unlinkable- outsourced-channel-monitoring/.
21. Fischer, M. J., Lynch, N. A., and Pater­son, M. S. Impossibility of distributed con­sensus with one faulty process. J. ACM 32, 2 (Apr. 1985), 374-382.
22. Friedenbach, M., BtcDrak, Dorier, N., and kinoshitajona. Bip 68: Relative lock­time using consensus-enforced sequence num­bers. https://github.com/bitcoin/bips/ blob/master/bip-0068.mediawiki.
23. Fugger, R. Money as ious in social trust networks & a proposal for a decen­tralized currency network protocol. Techni­cal Report, 2004. http://archive.ripple- project.org/decentralizedcurrency.pdf.
24. Ghosh, A., Mahdian, M., Reeves, D. M., Pennock, D. M., and Fugger, R. Mecha­nism design on trust networks. In WINE^07.
25. Giacomelli, I., Madsen, J., and Orlandi, C. Zkboo: Faster zero-knowledge for boolean circuits. In USENIX Security (2016).
26. g〇1111111 (pseudonym). Idea to im­prove lightning network. Forum post. https://bitcointalk.org/index.php?topic=1134319.0.
27. Green, M., and Miers, I. Bolt: Anonymous payment channels for decentralized currencies. In CCS (2017).
28. Heilman, E., Alshenibr, L., Baldimtsi, Scafuro, A., and Goldberg, S. TumbleBit: An untrusted bitcoin-compatible anonymous payment hub. In NDSS (2017).
29. Herlihy, M., and Shavit, N. On the nature of progress. In OPODIS (2011), pp. 313-328.
30. Herrera-Joancomarti, J., and Perez- SolA, C. Privacy in bitcoin transactions: New challenges from blockchain scalability so­lutions. In Modeling Decisions for Artificial Intelligence (2016).
31. Kosba, A., Miller, A., Shi, E., Wen, Z., and Papamanthou, C. Hawk: The blockchain model of cryptography and privacy­preserving smart contracts. In IEEE S&P (2016).
32. Koshy, P., Koshy, D., and McDaniel, P. An analysis of anonymity in bitcoin using p2p network traffic. In Financial Cryptography and Data Security (2014).
33. Lamport, L., and Fischer, M. Byzan­tine generals and transaction commit proto­cols. Tech. Rep. 62, SRI International, Apr. 1982.
34. Lind, J., Eyal, I., Pietzuch, P. R., and Sirer, E. G. Teechan: Payment channels using trusted execution environments. http://arxiv.org/abs/1612.07766.
35. Lopp, J. Lightning’s balancing act: Chal­lenges face bitcoin7s scalability savior. Blog en­try. http://www.coindesk.com/lightning-technical-challenges-bitcoin- scalability/.
36. Malavolta, G., Moreno-Sanchez, P., Kate, A., and Maffei, M. SilentWhispers: Enforcing security and privacy in credit net­works. In NDSS (2017).
37. McCorry, P., Moser, M., Shahandashti, S. F., and Hao, F. Towards bitcoin payment networks. In Australasian Conference Informa­tion Security and Privacy (2016).
38. Meiklejohn, S., and Orlandi, C. Privacy­enhancing overlays in bitcoin. In BITCOIN (2015).
39. Meiklejohn, S., Pomarole, M., Jordan, G., Levchenko, K., McCoy, D., Voelker,
40. M., and Savage, S. A fistful of bitcoins： Characterizing payments among men with no names. In IMC (2013).
41. Miller, A., Bentov, I., Kumaresan, R., and McCorry, P. Sprites: Payment chan­nels that go faster than lightning. CoRR abs/1702.05812 (2017).
42. Moreno-Sanchez, P., Kate, A., Maffei, M., and Pecina, K. Privacy preserving pay­ments in credit networks. In NDSS (2015).
43. Moreno-Sanchez, P., Modi, N., Songhela, R., Kate, A., and Fahmy, S. Mind your credit: Assessing the health of the ripple credit network. CoRR abs/1706.02358 (2017).
44. Moreno-Sanchez, P., Zafar, M. B., and Kate, A. Listening to whispers of ripple:

Linking wallets and deanonymizing transac­tions in the ripple network. In PETS (2016).

1. Nakamoto, S. Bitcoin: A peer-to-peer elec­tronic cash system. https://bitcoin.org/ bitcoin.pdf, 2008.
2. Papadimitriou, C. H. The serializability of concurrent database updates. J. ACM 26 (1979), 631-653.
3. Pfitzmann, A., and Hansen, M. Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management - a consolidated proposal for terminology, 2008.
4. Poon, J., and Dryja, T. The bit-coin lightning network:Scalable off-chain instant payments. Technical Report.

https://lightning.network/lightning-network-paper.pdf.

1. Post, A., Shah, V., and Mislove, A. Bazaar: Strengthening user reputations in on­line marketplaces. In NSDI (2011).
2. Prihodko, P., Zhigulin, S., Sahno,

M., and Ostrovskiy, A. Flare: An

approach to routing in lightning network. http://bitfury.com/content/5-white- papers-research/whitepaper\_flare\_ an\_approach\_to\_routing\_in\_lightning\_network\_7\_7\_2016.pdf.

1. Pryvalov, I. pyZKBoo++ implementation. Project’s website. https://sites.google. com/view/pyzkboopp/home.
2. Reid, F., and Harrigan, M. An analysis of anonymity in the bitcoin system. In Security and Privacy in Social Networks (2013).
3. Russell, R. Reaching the ground with light­ning. Technical Report. http://ozlabs.org/ ~rusty/ln-deploy-draft-01.pdf.
4. Spagnuolo, M., Maggi, F., and Zanero, S. BitIodine: Extracting intelligence from the bitcoin network. In Financial Cryptography and Data Security (2014).
5. Torpey, K. Brock pierce: Bitcoin’s scalabil­ity issues are a sign of its success. Blog entry.

https://bitcoinmagazine.com/articles/brock-pierce-bitcoin-s-scalability-issues-are-a-sign-of-its-success-1459867433/.

1. Torpey, K. Does the lightning network threaten bitcoin?s censorship resistance? Blog entry. **https://bitcoinmagazine.** com/articles/does-the-lightning- network-threaten-bitcoin-s-censorship- resistance-1461953131/.
2. Towns, A.Better privacy with

SNARKs. Mailing List.https://lists.linuxfoundation.org/pipermail/ lightning-dev/2015-November/000309. html.

1. van Wirdum, A. Segwit or not, bitfury is getting ready for lightning with suc­cessful bitcoin main net test. Blog entry. https://bitcoinmagazine.com/articles/ segwit-or-not-bitfury-ready-lightning- successful-bitcoin-main-net-test/.
2. Viswanath, B., Mondal, M., Gummadi, K. P., Mislove, A., and Post, A. Canal: Scaling social network-based sybil tolerance schemes. In EuroSys (2012).
3. Wikstrom, D. Auniversallycomposable mix- net. In Theory of Cryptography Conference (2004), M. Naor, Ed.

1. [↑](#footnote-ref-1)