

# 12 区块链应用

Let's now build on our understanding of bitcoin by looking at it as an *application platform*. Nowadays, many people use the term "blockchain" to refer to any application platform that shares the design principles of bitcoin. The term is often misused and applied to many things that fail to deliver the primary features that bitcoin's blockchain delivers.

现在，我们通过把比特币作为一个应用平台，以进一步加强对比特币的理解。

现在，很多人使用“区块链”这个词来表示任何使用了比特币设计原则的应用平台。

这个术语经常被滥用，并被应用于许多实际上比特币区块链不能提供的东西上。

In this chapter we will look at the features offered by the bitcoin blockchain, as an application platform. We will consider the application building *primitives*, which form the building blocks of any blockchain application. We will look at several important applications that use these primitives, such as colored coins, payment (state) channels, and routed payment channels (Lightning Network).

在本章中，我们将看看比特币区块链作为一个应用平台所提供的功能。

我们将考虑应用“构建原语”，它们形成了区块链应用的构建块。

然后，看看使用这些原语的几个重要应用：染色币、支付（状态）通道、路由支付通道（闪电网络）。

## 12.1 介绍

The bitcoin system was designed as a decentralized currency and payment system. However, most of its functionality is derived from much lower-level constructs that can be used for much broader applications. Bitcoin wasn't built with components such as accounts, users, balances, and payments. Instead, it uses a transactional scripting language with low-level cryptographic functions, as we saw in [\[transactions\]](#).

比特币系统被设计为：一个去中心化的货币和支付系统。

但是，它的大部分功能源于许多更低级的构件，这些构件可用于更广泛的应用。

比特币不是用这些组件构建的：帐户、用户、余额、支付。

而是使用一种交易脚本语言，它有低级加密功能。

Just as the higher-level concepts of accounts, balances, and payments can be derived from these basic primitives, so can many other complex applications. Thus, the bitcoin blockchain can become an application platform offering trust services to applications, such as smart contracts, far surpassing the original purpose of digital currency and payments.

就像账户、余额和支付这些高层概念可以从这些基本原语中导出一样，许多其它复杂的应用也可以。

因此，比特币区块链可以成为一个应用平台，向应用（例如智能合约）提供信任服务，这远远超出了作为数字货币和支付的最初目的。

## 12.2 构建块（原语）

When operating correctly and over the long term, the bitcoin system offers certain guarantees, which can be used as building blocks to create applications. These include: 当运行正常，且长期运行时，比特币系统提供了一定的保证，这种保证可作为构建块，用于创建应用。下面列出了18个构建块。

1	<b>No Double-Spend</b> 无双重支付	The most fundamental guarantee of bitcoin's decentralized consensus algorithm ensures that no UTXO can be spent twice. “比特币的去中心化共识算法”的最基本的保证是：保证utxo不会被花费两次。
2	<b>Immutability</b> 不可改变	Once a transaction is recorded in the blockchain and sufficient work has been added with subsequent blocks, the transaction's data becomes immutable. Immutability is underwritten by energy, as rewriting the blockchain requires the expenditure of energy to produce Proof-of-Work. The energy required and therefore the degree of immutability increases with the amount of work committed on top of the block containing a transaction. 一旦交易被记录在区块链中，并且用后面的区块添加了足够的工作量，那么，交易数据就变为不可篡改。 不可改变是由能源进行担保的，因为重写区块链需要消耗能源，才能产生工作量证明。所需的能源，以及由此带来的不可变的程度，随着在“包含交易的区块”之后提交的工作量而增加。
3	<b>Neutrality</b> 中立	The decentralized bitcoin network propagates valid transactions regardless of the origin or content of those transactions. This means that anyone can create a valid transaction with sufficient fees and trust they will be able to transmit that transaction and have it included in the blockchain at any time. 去中心化的比特币网络传播有效的交易，而不管这些交易的来源或内容如何。 这意味着，任何人在任何时候都可以用足够的交易费来创建有效的交易，并相信网络会传输这个交易，并且可以包含在区块链中。
4	<b>Secure timestamping</b> 安全时间戳	The consensus rules reject any block whose timestamp is too far in the past or future. This ensures that timestamps on blocks can be trusted. The timestamp on a block implies an unspent-before guarantee for the inputs of all included transactions. 共识规则拒绝任何时间戳距离现在太远（过去和将来）的区块。 这可以确保区块上的时间戳可以被信任。 区块上的时间戳意味着，为所有交易的输入提供了以前未被花费过的保证。
5	<b>Authorization</b> 授权	Digital signatures, validated in a decentralized network, offer authorization guarantees. Scripts that contain a requirement for a digital signature cannot be executed without authorization by the holder of the private key implied in the script. 数字签名被去中心化网络验证，它提供了授权保证。 没有私钥持有人的授权，脚本不会被执行。
6	<b>Auditability</b> 审计能力	All transactions are public and can be audited. All transactions and blocks can be linked back in an unbroken chain to the genesis block. 所有交易都是公开的，可以被审计。 所有交易和区块都可以用一个区块链链接起来，并最终追溯到创始区块。
7	<b>Accounting</b> 会计	In any transaction (except the coinbase transaction) the value of inputs is equal to the value of outputs plus fees. It is not possible to create or destroy bitcoin value in a transaction. The outputs cannot exceed the inputs. 在任何交易中（币基交易除外），输入金额等于输出金额加上交易费。 在交易中不可能创建或销毁比特币的价值。输出不能超过输入。

8	<b>Nonexpiration</b> 永不过期	<p>A valid transaction does not expire. If it is valid today, it will be valid in the near future, as long as the inputs remain unspent and the consensus rules do not change.</p> <p>有效的交易不会过期。</p> <p>如果今天有效，将来仍然有效，只要输入仍未花费，且共识规则没有改变。</p>
9	<b>Integrity</b> 完整性	<p>A bitcoin transaction signed with SIGHASH_ALL or parts of a transaction signed by another SIGHASH type cannot be modified without invalidating the signature, thus invalidating the transaction itself.</p> <p>使用SIGHASH_ALL签名的一个比特币交易，或由另一个SIGHASH类型签名的交易的一部分，不能在签名还有效的情况下被修改，从而使交易本身无效。</p>
10	<b>Transaction Atomicity</b> 交易原子性	<p>Bitcoin transactions are atomic. They are either valid and confirmed (mined) or not. Partial transactions cannot be mined and there is no interim state for a transaction. At any point in time a transaction is either mined, or not.</p> <p>比特币交易是原子操作。（注：原子性是指交易全部执行或完全不执行，不存在中间状态）交易要么是有效的并且经过确认的（挖矿），要么不是。</p> <p>不存在挖矿出交易的一部分，交易也不存在中间状态。</p> <p>在任何时间，交易要么被挖出，要么没被挖出，不存在中间状态。</p>
11	<b>Discrete(Indivisible) Units of Value</b> 离散（不可分割）的价值单位	<p>Transaction outputs are discrete and indivisible units of value. They can either be spent or unspent, in full. They cannot be divided or partially spent.</p> <p>交易输出是离散和不可分割的价值单位。</p> <p>它们要么整体被花费，要么整体没被花费。</p> <p>它不能被分割或花费一部分。</p>
12	<b>Quorum of Control</b> 控制法定人数	<p>Multisignature constraints in scripts impose a quorum of authorization, predefined in the multisignature scheme. The M-of-N requirement is enforced by the consensus rules.</p> <p>脚本中的多签名规定了多签名方案中的预定义的法定权限。</p> <p>m-n要求由共识规则执行。</p>
13	<b>Timelock/Aging</b> 时间锁/老化	<p>Any script clause containing a relative or absolute timelock can only be executed after its age exceeds the time specified.</p> <p>包含相对或绝对时间锁的任何脚本语句，只能在超过指定时间后执行。</p>
14	<b>Replication</b> 复制	<p>The decentralized storage of the blockchain ensures that when a transaction is mined, after sufficient confirmations, it is replicated across the network and becomes durable and resilient to power loss, data loss, etc.</p> <p>区块链的去中心化存储确保了在交易在被挖矿之后，经过充分的确认，它被复制到整个网络上，可以耐受停电、数据丢失等的影响。</p>
15	<b>Forgery Protection</b> 防伪造	<p>A transaction can only spend existing, validated outputs. It is not possible to create or counterfeit value.</p> <p>每个交易只能花费经过验证的输出。</p> <p>不可能创建或伪造价值。</p>
16	<b>Consistency</b> 一致性	<p>In the absence of miner partitions, blocks that are recorded in the blockchain are subject to reorganization or disagreement with exponentially decreasing likelihood, based on the depth at which they are recorded. Once deeply recorded, the computation and energy required to change makes change practically infeasible.</p> <p>在没有矿工分化的情况下，根据记录的深度，这种可能性会呈指数级下降：记录在区块链中的区块被从新组织或不认可。</p> <p>一旦被记录在深层，改变所需的计算和能量将大到实际不可行的程度。</p>

17	<b>Recording External State</b> 记录外部状态	A transaction can commit a data value, via OP_RETURN, representing a state transition in an external state machine. 每个交易可以通过OP_RETURN提交一个值，表示一个外部状态机中的一个状态转换。
18	<b>Predictable Issuance</b> 可预测发行量	Less than 21 million bitcoin will be issued, at a predictable rate. 比特币总计不到2100万个，以可预测的速度发行。

The list of building blocks is not complete and more are added with each new feature introduced into bitcoin.

这个列表并不完整，当比特币中引入新功能时还会增加更多构建块。

## 12.3使用构建块的应用

The building blocks offered by bitcoin are elements of a trust platform that can be used to compose applications. Here are some examples of applications that exist today and the building blocks they use:

由比特币提供的构建块是一个可信平台的元素，可用于构建应用。

这里有一些当前的应用例子，及其这些应用使用的构建块。

### (1) Proof-of-Existence (Digital Notary) 证明存在（数字公证）

Immutability + Timestamp + Durability.

A digital fingerprint can be committed with a transaction to the blockchain, proving that a document existed (Timestamp) at the time it was recorded. The fingerprint cannot be modified ex-post-facto (Immutability) and the proof will be stored permanently (Durability).

Immutability 不可篡改性	Timestamp 时间戳	Durability 永久性
-----------------------	------------------	-------------------

- Timestamp : 可用一个交易把数字指纹提交给区块链，以证明文件在这个时间内是存在的。
- Immutability : 数字指纹不能在事后修改
- Durability : 证据将被永久存储

[ddk应用例子：保护论文知识产权](#)

### (2) Kickstarter (Lighthouse) 灯塔

Consistency + Atomicity + Integrity.

If you sign one input and the output (Integrity) of a fundraiser transaction, others can contribute to the fundraiser but it cannot be spent (Atomicity) until the goal (output value) is funded (Consistency).

Consistency 一致性	Atomicity 原子性	Integrity 完整性
--------------------	------------------	------------------

- Integrity : 如果你签署一个众筹交易的一个输入和输出，
- Consistency : 别人可以参与众筹，但在目标（输出值）达成之前，
- Atomicity : 这笔钱不能被花费出去。

[ddk应用例子：众筹交易](#)

### (3) Payment Channels 支付通道

Quorum of Control + Timelock + No Double Spend + Nonexpiration + Censorship Resistance + Authorization.

A multisig 2-of-2 (Quorum) with a timelock (Timelock) used as the "settlement" transaction of a payment channel can be held (Nonexpiration) and spent at any time (Censorship Resistance) by either party (Authorization). The two parties can then create commitment transactions that double-spend (No Double-Spend) the settlement on a shorter timelock (Timelock).

Quorum of Control 控制法定人数	Timelock 时间锁	No Double Spend 无双重支付
Nonexpiration 永不过期	Censorship Resistance 耐审查	Authorization 授权

一个有时间锁 (Timelock) 的2-2多签名 (Quorum) , 用作支付通道的结算交易, 可以被任何一方 (Authorization) , 在任何时间 (Censorship Resistance), 持有 (Nonexpiration) 和花费。

然后, 双方创建承诺交易, 它在一个更短的时间锁 (Timelock) 上双花 (No Double-Spend) 这个结算。

[ddk应用例子: 支付通道](#)

# 12.4染色币（Colored Coins）

The first blockchain application we will discuss is *colored coins*. Colored coins refers to a set of similar technologies that use bitcoin transactions to record the creation, ownership, and transfer of extrinsic assets other than bitcoin. By "extrinsic" we mean assets that are not stored directly on the bitcoin blockchain, as opposed to bitcoin itself, which is an asset intrinsic to the blockchain.

我们讨论的第1个区块链应用是“染色币”。

“染色币”是一些类似的技术，它们使用比特币交易来记录“外部资产”的创建、所有权、转移。

所谓“外部资产”，指这些资产不直接存储在比特币区块链上。

比特币本身存储在区块链，比特币是内存在区块链中的资产。

Colored coins are used to track digital assets as well as physical assets held by third parties and traded through colored coins certificates of ownership. Digital asset colored coins can represent intangible assets such as a stock certificate, license, virtual property (game items), or most any form of licensed intellectual property (trademarks, copyrights, etc.). Tangible asset colored coins can represent certificates of ownership of commodities (gold, silver, oil), land title, automobiles, boats, aircraft, etc.

“染色币”用于跟踪第三方持有的“数字资产”和“实物资产”，并通过“染色币所有权证书”来进行交易。

染色币	代表	资产例子
数字资产染色币	无形资产	股票证书、许可证、虚拟财产（游戏装备），或大多数形式的许可知识产权（商标、版权等）
实物资产染色币	实物资产	商品（黄金/白银/石油）、土地所有权、汽车、船只、飞机等所有权证书

The term derives from the idea of "coloring" or marking a nominal amount of bitcoin, for example, a single satoshi, to represent something other than the bitcoin value itself. As an analogy, consider stamping a \$1 note with a message saying, "this is a stock certificate of ACME" or "this note can be redeemed for 1 oz of silver" and then trading the \$1 note as a certificate of ownership of this other asset.这个术语源于“着色”或“记入比特币的名义金额”的想法，

例如，1聪用来表示比特币价值本身以外的东西。

打个比方，我们给1美元钞票上标记信息：这是ACME的股票证书，或这张钞票可以兑换1盎司的银。

然后，可以这1美元钞票作为所有权证明交换其它资产。

The first implementation of colored coins, named *Enhanced Padded-Order-Based Coloring* or *EPOBC*, assigned extrinsic assets to a 1-satoshi output. In this way, it was a true "colored coin," as each asset was added as an attribute (color) of a single satoshi.

染色币的第一个实现是EPOBC，它将外部资产标记在1聪输出上。

这样，因为每个资产作为属性（颜色）被添加到了1聪上，它成了一个“染色币”。

More recent implementations of colored coins use the OP\_RETURN script opcode to store metadata in a transaction, in conjunction with external data stores that associate the metadata to specific assets.

染色币的最近实现使用OP\_RETURN脚本操作码在交易中存储元数据，它与外部数据存储结合（把元数据与特定资产关联）。

The two most prominent implementations of colored coins today are [Open Assets](#) and [Colored Coins by Colu](#). These two systems use different approaches to colored coins and are not compatible. Colored coins created in one system cannot be seen or used in the other system.

当前，染色币的两个最著名的实现是：Open Assets 和 Colu。

这两个系统使用不同的方法实现染色，并不兼容。  
在一个系统中创建的染色币在其它系统中无法看到或使用。

## 12.4.1 使用染色币

Colored coins are created, transferred, and generally viewed in special wallets that can interpret the colored coins protocol metadata attached to bitcoin transactions. Special care must be taken to avoid using a colored-coin-related key in a regular bitcoin wallet, as the regular wallet may destroy the metadata. Similarly, colored coins should not be sent to addresses managed by regular wallets, but only to addresses that are managed by wallets that are colored-coin-aware. Both Colu and Open Assets systems use special colored-coin addresses to mitigate this risk and to ensure that colored coins are not sent to unaware wallets.

染色币被创建、转移，并且通常用特殊的比特币交易的钱包来查看，这些钱包能理解含有染色币协议元数据。

必须特别注意，避免在常规的比特币钱包中使用染色币相关的密钥，因为常规钱包可能会破坏元数据。同样地，染色币也不应该被发送到由常规钱包管理的地址，而只能发送到由染色币能够识别的钱包管理的地址。

Colu和Open Assets这两个系统都使用特殊的染色币地址来降低这种风险，并确保染色币不会发送到不能识别的钱包。

Colored coins are also not visible to most general-purpose blockchain explorers. Instead, you must use a colored-coins explorer to interpret the metadata of a colored coins transaction.

染色币对多数通用区块链浏览器也是不可见的。

你必须使用染色币浏览器来解释染色币交易的元数据。

An Open Assets-compatible wallet application and blockchain explorer can be found at [coinprism](https://www.coinprism.info/).

Open Assets兼容的钱包应用程序和区块链浏览器可以在coinprism查找。

<https://www.coinprism.info/>

A Colu Colored Coins-compatible wallet application and blockchain explorer can be found at [Blockchain Explorer](http://coloredcoins.org/explorer/).

Colu染色币兼容的钱包应用程序和区块链探索器可以在Blockchain Explorer中找到。

<http://coloredcoins.org/explorer/>

A Copay wallet plug-in can be found at [Colored Coins Copay Addon](http://coloredcoins.org/colored-coins-copay-addon/).

Copay钱包插件可以在Colored Coins Copay Addon中找到。

<http://coloredcoins.org/colored-coins-copay-addon/>

## 12.4.2 发行染色币

Each of the colored coins implementations has a different way of creating colored coins, but they all provide similar functionality. The process of creating a colored coin asset is called *issuance*. An initial transaction, the *issuance transaction* registers the asset on the bitcoin blockchain and creates an *asset ID* that is used to reference the asset. Once issued, assets can be transferred between addresses using *transfer transactions*.

每个染色币的实现都通过不同的方法创造染色币，但它们都提供类似的功能。

创造染色币资产的过程称为“发行”。

作为初始交易，发行交易将资产登记在比特币区块链上，并创建用于引用资产的资产ID。

一旦发行，资产可以使用转账交易在地址之间传递。



Assets issued as colored coins can have multiple properties. They can be *divisible* or *indivisible*, meaning that the amount of asset in a transfer can be an integer (e.g., 5) or have decimal subdivision (e.g., 4.321). Assets can also have *fixed issuance*, meaning a certain amount are issued only once, or can be *reissued*, meaning that new units of the asset can be issued by the original issuer after the initial issuance.

作为染色币发行的资产可以有多种属性。

它们可以是可分割的或不可分割的，这意味着转账中的资产量可以是整数（比如5）或有小数（比如4.321）。

资产也可以固定发行，意思是一定数量的资产只可以发行一次；或者可以被再次发行，意味着原始发行人在初始发行后可以发行新资产。

Finally, some colored coins enable *dividends*, allowing the distribution of bitcoin payments to the owners of a colored coin asset in proportion to their ownership.

最后，一些染色币启用分红，即允许按照所有权按比例分配比特币，支付给染色币资产的所有者。

## 12.4.3 染色币交易

The metadata that gives meaning to a colored coin transaction is usually stored in one of the outputs using the OP\_RETURN opcode. Different colored coins protocols use different encodings for the content of the OP\_RETURN data. The output containing the OP\_RETURN is called the *marker output*.

给染色币交易提供意义的元数据通常使用OP\_RETURN操作码存储在一个输出中。

不同颜色的染色币协议对OP\_RETURN数据的内容使用不同的编码。

包含OP\_RETURN的输出称为标记输出。

The order of the outputs and position of the marker output may have special meaning in the colored coins protocol. In Open Assets, for example, any outputs before the marker output represent asset issuance. Any outputs after the marker represent asset transfer. The marker output assigns specific values and colors to the other outputs by referencing their order in the transaction.

输出的顺序和标记输出的位置在染色币协议中可能具有特殊含义。

例如，在Open Assets中，标记输出之前的任何输出都代表资产发行。标记输出后的任何输出表示资产转账。通过参考各个输出在转账中的顺序标记输出将特定值和颜色分配给其他输出。

In Colored Coins (Colu), by comparison, the marker output encodes an opcode that determines how the metadata is interpreted. Opcodes 0x01 through 0x0F indicate an issuance transaction. An issuance opcode is usually followed by an asset ID or other identifier that can be used to retrieve the asset information from an external source (e.g., bittorrent). Opcodes 0x10 through 0x1F represent a transfer transaction. Transfer transaction metadata contain simple scripts that transfer specific amounts of assets from inputs to outputs, by reference to their index. Ordering of inputs and outputs is therefore important in the interpretation of the script.

在Colu中，通过比较，标记输出编码一个定义元数据该如何被理解的操作码。操作码0x01至0x0F表示发行交易。发行操作码通常后面是资产ID或可用于从外部来源（例如bittorrent）取得资产信息的其他标识符。操作码0x10到0x1F表示转账交易。转账交易元数据包含简单的脚本，通过参考输入输出的索引（顺序），将特定数量的资产从输入转账到输出。因此，输入和输出的排序对脚本的解释很重要。

If the metadata is too long to fit in OP\_RETURN, the colored coins protocol may use other "tricks" to store metadata in a transaction. Examples include putting metadata in a redeem script, followed by OP\_DROP opcodes to ensure the script ignores the metadata. Another mechanism used is a 1-of-N multisig script where only the first public key is a real public key that can spend the output and subsequent "keys" are replaced by encoded metadata. 如果元数据太长而不能放入OP\_RETURN，则染色币协议可能会使用其他“技巧”在交易中存储元数据。示例包括将元数据放在兑换脚本中，紧接着OP\_DROP操作码，以确保脚本忽略元数据。另一种被使用的机制是1-N 多重签名脚本，其中只有第一个公钥是可以花费输出的真实公钥，随后的“密钥”则用被编码的元数据替代。

In order to correctly interpret the metadata in a colored coins transaction you must use a compatible wallet or block explorer. Otherwise, the transaction looks like a "normal" bitcoin transaction with an OP\_RETURN output.

为了正确解释染色币交易中的元数据，你必须使用兼容的钱包或区块资源浏览器。

否则，该交易会看起来像一个具有OP\_RETURN输出的“正常”比特币交易。

As an example, I created and issued a MasterBTC asset using colored coins. The MasterBTC asset represents a voucher for a free copy of this book. These vouchers can be transferred, traded, and redeemed using a colored coins-compatible wallet.

例如，我使用染色币创建并发行了MasterBTC资产。

“MasterBTC”代表了可以获取本书免费拷贝的兑换券。

这些兑换券可以使用染色币兼容的钱包进行转让，交易和兑换。

For this particular example, I used the wallet and explorer at <https://coinprism.info>, which uses the Open Assets colored coins protocol.

对于这个例子，我使用了<https://coinprism.info>的钱包和浏览器，它使用了Open Assets染色币协议。

[The issuance transaction as viewed on coinprism.info](#) shows the issuance transaction using the Coinprism block explorer:

下图在coinprism.info上查看的发行交易 显示使用Coinprism块浏览器的发行交易：<https://www.coinprism.info/tx/10d7c4e022f35288779be6713471151ede967caaa39eecd35296aa36d9c109ec>

#### Transaction

Hash	10d7c4e022f35288779be6713471151ede967c...	Transaction confirmed
Date	Sunday, August 17, 2014 5:42:41 PM	Confirmations 137057 confirmations
Fee paid	0.0001 BTC	Time Sunday, August 17, 2014 5:...
Assets transacted	1	Block 00000000000000000150ab5...
		Height 316117




Bitcoin
  akTnsDt5uzpioRST76VFRQM8q8sBF... -0.0001 Fees 0.0001
Free copy of "Mastering Bitcoin" AcuRVsoa81hoLHmVTNXrRD8KpTqUXe...
 + Issued assets -20 akTnsDt5uzpioRST76VFRQM8q8sBFnQ... 20

Figure 1. The issuance transaction as viewed on coinprism.info

As you can see, coinprism shows the issuance of 20 units of "Free copy of Mastering Bitcoin," the MasterBTC asset, to a special colored coin address:

正如你看到的，coinprism显示了发行的20个“精通比特币的免费拷贝”，简称为MasterBTC的资产，发给了一个特殊的染色币地址：

akTnsDt5uzpioRST76VFRQM8q8sBFnQiwcx

warning: Any funds or colored assets sent to this address will be lost forever. Do not send value to this example address!

**警告：**发送到该地址的任何资金或染色币将永远丢失。不要发送到这个示例地址！

The transaction ID of the issuance transaction is a "normal" bitcoin transaction ID. [The issuance transaction on a block explorer that doesn't decode colored coins](#) shows that same transaction in a block explorer that doesn't decode colored coins. We'll use *blockchain.info*:

发行交易的交易ID是正常比特币交易ID。

下图12-2不对染色币进行解码的区块链浏览器中看到的发行交易显示同一笔交易（和12.1同一笔）在不会对区块链解码的区块浏览器中的样子。

我们将使用blockchain.info：

<https://blockchain.info/tx/10d7c4e022f35288779be6713471151ede967caaa39eecd35296aa36d9c109ec>

## Transaction View information about a bitcoin transaction

10d7c4e022f35288779be6713471151ede967caaa39eecd35296aa36d9c109ec		
1HpyyIGaXLq7ZCHk3s9EKVEFoG15oLn2Us (0.01 BTC - Output)	→ 1HpyyIGaXLq7ZCHk3s9EKVEFoG15oLn2Us - (Unspent) 0.000006 BTC Unable to decode output address - (Unspent) 0 BTC 1HpyyIGaXLq7ZCHk3s9EKVEFoG15oLn2Us - (Spent) 0.009894 BTC	0.0099 BTC

Figure 2. The issuance transaction on a block explorer that doesn't decode colored coins

As you can see, *blockchain.info* doesn't recognize this as a colored coins transaction. In fact, it marks the second output with "Unable to decode output address" in red letters.

正如你所看到的，*blockchain.info*不认为这是一个染色币交易。事实上，它以红色字母表示第二个输出“无法解码输出地址”。

If you select "Show scripts & coinbase" on that screen, you can see more detail about the transaction ([The scripts in the issuance transaction](#)).

如果你在该屏幕上选择“显示脚本和币基”，可以看到有关交易的更多详细信息(下图3发行交易的脚本)。

### Output Scripts

OP_DUP OP_HASH160 b895201a7cfd91a9bfb3b42cd114d42e3a634d2 OP_EQUALVERIFY OP_CHECKSIG	OK
OP_RETURN 4f41010001141b753d68747470733a2f2f6370722e736d2f466f796b777248365559 (decoded) "OA_____u=https://cpr.sm/FoykwrH6UY"	Strange
OP_DUP OP_HASH160 b895201a7cfd91a9bfb3b42cd114d42e3a634d2 OP_EQUALVERIFY OP_CHECKSIG	OK

Figure 3. The scripts in the issuance transaction

Once again, *blockchain.info* doesn't understand the second output. It marks it with "Strange" in red letters. However, we can see that some of the metadata in the marker output is human-readable:

*blockchain.info*并不能理解第二个输出。它以红色字母表示“Strange”。但是，我们可以看到，标记输出中的一些元数据是可读的：

```
OP_RETURN 4f41010001141b753d68747470733a2f2f6370722e736d2f466f796b777248365559
(decoded) "OA_____u=https://cpr.sm/FoykwrH6UY"
```

Let's retrieve the transaction using *bitcoin-cli*:

我们使用*bitcoin-cli*检索交易：

```
$ bitcoin-cli decoderawtransaction `bitcoin-cli getrawtransaction
10d7c4e022f35288779be6713471151ede967caaa39eecd35296aa36d9c109ec`
```

Stripping out the rest of the transaction, the second output looks like this:

去掉其余的交易，第二个输出如下所示：

```
{
  "value": 0.00000000,
  "n": 1,
  "scriptPubKey": "OP_RETURN
4f41010001141b753d68747470733a2f2f6370722e736d2f466f796b777248365559"
}
```

The prefix 4F41 represents the letters "OA", which stands for "Open Assets" and helps us identify that what follows is metadata defined by the Open Assets protocol. The ASCII-encoded string that follows is a link to an asset definition:

前缀4F41表示字母“OA”，代表“Open Assets”，并帮助我们确定以下元数据是由Open Assets协议定义的紧接着的ASCII编码的字符串是指向资产定义的链接：

```
u=https://cpr.sm/FoykwrH6UY
```

If we retrieve this URL, we get a JSON-encoded asset definition, as shown here:  
如果我们检索此URL，我们将获得JSON编码的资产定义，如下所示：

```
{
  "asset_ids": [
    "AcuRVsoa81hoLHmVTNXrRD8KpTqUXeqwgH"
  ],
  "contract_url" : null,
  "name_short" : "MasterBTC",
  "name" : "Free copy of \"Mastering Bitcoin\"",
  "issuer" : "Andreas M. Antonopoulos",
  "description" : "This token is redeemable for a free copy of the book \"Mastering Bitcoin\"",
  "description_mime" : "text/x-markdown; charset=UTF-8",
  "type" : "Other",
  "divisibility" : 0,
  "link_to_website" : false,
  "icon_url" : null,
  "image_url" : null,
  "version" : "1.0"
}
```

**add说明：（未必正确）**

- 这两个说的应该是一回事：
  - 《精通比特币》12.4 染色币
  - 《区块链 技术驱动金融》 9.1.8 Overlay Currencies
- 必须开发更复杂的逻辑来验证新货币中的交易，并且这个逻辑必须驻留在每个终端用户那里，他们参与发送和接受这种货币。

## 12.5 合约币 (Counterparty)

Counterparty is a protocol layer built on top of bitcoin. The Counterparty protocol, similar to colored coins, offers the ability to create and trade virtual assets and tokens. In addition, Counterparty offers a decentralized exchange for assets. Counterparty is also implementing smart contracts, based on the Ethereum Virtual Machine (EVM).

合约币是在比特币之上建立的协议层。

与“染色币”类似，“合约币协议”提供了创建和交易虚拟资产和代币的能力。

此外，合约币提供了去中心化的资产交易。

基于Ethereum虚拟机（EVM），合约币还在实现了“智能合约”。

Like the colored coins protocols, Counterparty embeds metadata in bitcoin transactions, using the OP\_RETURN opcode or 1-of-N multisignature addresses that encode metadata in the place of public keys. Using these mechanisms, Counterparty implements a protocol layer encoded in bitcoin transactions. The additional protocol layer can be interpreted by applications that are Counterparty-aware, such as wallets and blockchain explorers, or any application built using the Counterparty libraries.

像染色币协议一样，合约币使用“OP\_RETURN操作码”或“1-N多签名地址”将元数据嵌入到比特币交易中，它在公钥的位置编码元数据。

使用这些机制，合约币实现了在比特币交易中编码的协议层。

这个协议层可以由能理解合约币的应用程序来解读，如钱包和区块链浏览器，或使用合约币库（library）构建的任何应用程序。

Counterparty can be used as a platform for other applications and services, in turn. For example, Tokenly is a platform built on top of Counterparty that allows content creators, artists, and companies to issue tokens that express digital ownership and can be used to rent, access, trade, or shop for content, products, and services. Other applications leveraging Counterparty include games (Spells of Genesis) and grid computing projects (Folding Coin).

反过来，合约币可以用作一个平台，提供给其它应用程序和服务。

例如，Tokenly是一个建立在合约币之上的平台，允许内容创作者，艺术家和公司发行表达数字所有权的代币，并可用于租赁、访问、交易或购买内容、产品和服务。

利用交易合约币的“其它应用”包括游戏（Spells of Genesis）和网格计算项目（Folding Coin）。

More details about Counterparty can be found at <https://counterparty.io>. The open source project can be found at <https://github.com/CounterpartyXCP>.

更多关于合约币的内容参见：<https://counterparty.io>

开源项目参见：<https://github.com/CounterpartyXCP>

**dad说明：**参见《区块链 技术驱动金融》9.1.8 Overlay Currencies

## 12.6 支付通道和状态通道

### Payment Channels and State Channels

*Payment channels* are a trustless mechanism for exchanging bitcoin transactions between two parties, outside of the bitcoin blockchain. These transactions, which would be valid if settled on the bitcoin blockchain, are held off-chain instead, acting as *promissory notes* for eventual batch settlement. Because the transactions are not settled, they can be exchanged without the usual settlement latency, allowing extremely high transaction throughput, low (submillisecond) latency, and fine (satoshi-level) granularity.

“支付通道”是一种无信任机制，用于在比特币区块链之外，双方之间交换比特币交易。

这些交易如果在比特币区块链上结算则是有效的，但它们却在区块链外，以期票形式等待最终批量结算。由于交易尚未结算，因此它们可以在没有通常的结算延迟的情况下进行交换，从而可以满足极高的交易吞吐量、低（亚毫秒）延迟、细（satoshi级）粒度。

Actually, the term *channel* is a metaphor. State channels are virtual constructs represented by the exchange of state between two parties, outside of the blockchain. There are no "channels" per se and the underlying data transport mechanism is not the channel. We use the term channel to represent the relationship and shared state between two parties, outside of the blockchain.

实际上，“通道”是一个比喻。

“状态通道”是虚拟结构，在区块链外，由双方之间的状态交换来表示。

实际上没有“通道”，底层数据传输机制不是通道。

我们使用“通道”表示区块链之外双方之间的关系和共享状态。

To further explain this concept, think of a TCP stream. From the perspective of higher-level protocols it is a "socket" connecting two applications across the internet. But if you look at the network traffic, a TCP stream is just a virtual channel over IP packets. Each endpoint of the TCP stream sequences and assembles IP packets to create the illusion of a stream of bytes. Underneath, it's all disconnected packets. Similarly, a payment channel is just a series of transactions. If properly sequenced and connected, they create redeemable obligations that you can trust even though you don't trust the other side of the channel.

为了进一步解释这个概念，想一想TCP流。

从高层协议的角度来看，它是一个横跨互联网连接两个应用程序的“socket”。

但是，如果你查看网络流量，TCP流只是IP数据包之上的虚拟通道。TCP流的每个端点通过排序并组装IP数据包以产生字节流错觉。实际上在背后，所有数据包都是断开分散的。

同理，支付通道只是一系列交易。如果妥善排序和连接，即使你不信任通道的另一方，（经过排序连接后的交易）也可以创建可以信任的可兑换的证券。

In this section we will look at various forms of payment channels. First, we will examine the mechanisms used to construct a one-way (unidirectional) payment channel for a metered micropayment service, such as streaming video. Then, we will expand on this mechanism and introduce bidirectional payment channels. Finally, we will look at how bidirectional channels can be connected end-to-end to form multihop channels in a routed network, first proposed under the name *Lightning Network*.

在本节中，我们将介绍各种形式的“支付通道”。

- 首先，将检视用于构建计量小额支付服务（如流媒体视频）的单向支付通道的机制。
- 然后，将扩大这一机制，引入双向付费通道。
- 最后，看看双向通道如何被连接为端到端，从而在一个路由中形成多个通道，这是闪电网络首次提出的。

Payment channels are part of the broader concept of a *state channel*, which represents an off-chain alteration of state, secured by eventual settlement in a blockchain. A payment channel is a state channel where the state being altered is the balance of a virtual currency.

“状态通道”是比“支付通道”更快的概念。



“状态通道”代表了链外状态的变化，通过区块链上的最终结算得到保障。  
支付通道是一种状态通道，其中被改变的状态是虚拟货币的余额。

## 12.6.1 状态通道：基本概念和术语

A state channel is established between two parties, through a transaction that locks a shared state on the blockchain. This is called the *funding transaction* or *anchor transaction*. This single transaction must be transmitted to the network and mined to establish the channel. In the example of a payment channel, the locked state is the initial balance (in currency) of the channel.

一个状态通道是在两方之间通过一个交易建立的，这个交易锁定了区块链上的一个共享状态。

这被称为“资金交易”或“锚点交易”。

这笔交易必须传送到网络并被挖矿，才能建立这个通道。

在支付通道的示例中，“被锁定的状态”就是这个通道的初始余额（以货币计）。

The two parties then exchange signed transactions, called *commitment transactions*, that alter the initial state. These transactions are valid transactions in that they *could* be submitted for settlement by either party, but instead are held off-chain by each party pending the channel closure. State updates can be created as fast as each party can create, sign, and transmit a transaction to the other party. In practice this means that thousands of transactions per second can be exchanged.

随后，双方交换签名的交易，这被称为“承诺交易”。承诺交易会改变初始状态。

这些交易都是有效的，因为任何一方都可以提交结算的请求，而不是各方脱链持有，等待通道关闭。

状态更新可以快速创建，每一方可以创建、签名和发送一个交易给另一方。

实践中，这意味着每秒可进行数千笔交易。

When exchanging commitment transactions the two parties also invalidate the previous states, so that the most up-to-date commitment transaction is always the only one that can be redeemed. This prevents either party from cheating by unilaterally closing the channel with an expired prior state that is more favorable to them than the current state. We will examine the various mechanisms that can be used to invalidate prior state in the rest of this chapter.

当交换“承诺交易”时，双方同时废止之前的状态，因此，最新的承诺交易总是唯一可赎回的承诺交易。

这样可以防止任何一方欺骗：用一个已过期的以前状态来单方面关闭通道，这个状态比当前状态更有利。

我们将在本章的其余部分中检视可用于废止先前状态的各种机制。

Finally, the channel can be closed either cooperatively, by submitting a final *settlement transaction* to the blockchain, or unilaterally, by either party submitting the last commitment transaction to the blockchain. A unilateral close option is needed in case one of the parties unexpectedly disconnects. The settlement transaction represents the final state of the channel and is settled on the blockchain.

最后，通道可以合作关闭（即向区块链提交最后的结算交易），或单方面关闭（由任何一方提交最后承诺交易到区块链）。

单方面关闭的选项是必要的，以防万一交易中的一方意外断开连接。

结算交易代表通道的最终状态，并在链上进行结算。

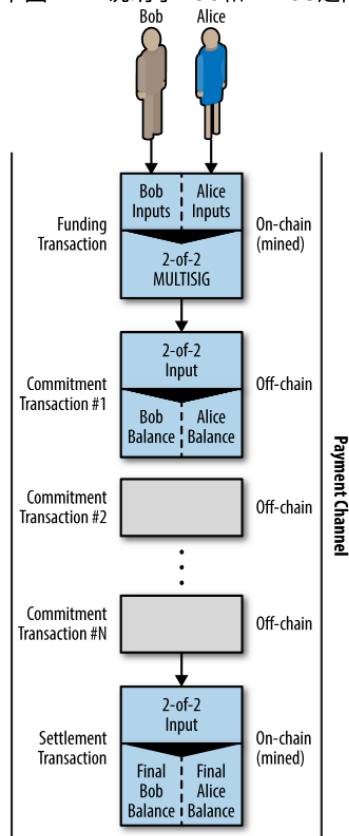
In the entire lifetime of the channel, only two transactions need to be submitted for mining on the blockchain: the funding and settlement transactions. In between these two states, the two parties can exchange any number of commitment transactions that are never seen by anyone else, nor submitted to the blockchain.

在这个通道的整个生命周期中，只有两个交易需要提交给区块链进行挖矿：资金交易、结算交易。

在这两个状态之间，双方可以交换任何数量的承诺交易，任何其他人永远不会看到，也不会提交到链上。

[A payment channel between Bob and Alice, showing the funding, commitment, and settlement transactions](#) illustrates a payment channel between Bob and Alice, showing the funding, commitment, and settlement transactions.

下图12-4说明了Bob和Alice之间的支付通道，显示了资金交易、承诺交易和结算交易。



## 12.6.2简单支付通道示例

To explain state channels, we start with a very simple example. We demonstrate a one-way channel, meaning that value is flowing in one direction only. We will also start with the naive assumption that no one is trying to cheat, to keep things simple. Once we have the basic channel idea explained, we will then look at what it takes to make it trustless so that neither party *can* cheat, even if they are trying to.

为了解释“状态通道”，我们从一个非常简单的例子开始。

我们展示一个单向通道，意味着价值只向一个方向流动。

为了便于解释，我们还以一个天真的假设开始，假设没有人要试图欺骗他人。

一旦我们解释了基本的通道概念，我们将会接着看看是什么使得支付通道可以无信任化，从而没有一方可以欺骗，即使他们想欺骗。

For this example we will assume two participants: Emma and Fabian. Fabian offers a video streaming service that is billed by the second using a micropayment channel. Fabian charges 0.01 millibit (0.00001 BTC) per second of video, equivalent to 36 millibits (0.036 BTC) per hour of video. Emma is a user who purchases this streaming video service from Fabian. [Emma purchases streaming video from Fabian with a payment channel, paying for each second of video](#) shows Emma buying the video streaming service from Fabian using a payment channel.

对于这个例子，我们假设两个参与者是：Emma和Fabian。

Fabian提供了一个视频流服务，它是按秒来计费，使用了一个微支付通道。

每秒收费0.01毫比（0.00001 BTC），相当于每小时36毫比（0.036 BTC）。

Emma从Fabian那里购买视频流服务。

下图12-5显示了Emma使用一个支付通道从Fabian购买视频流服务。



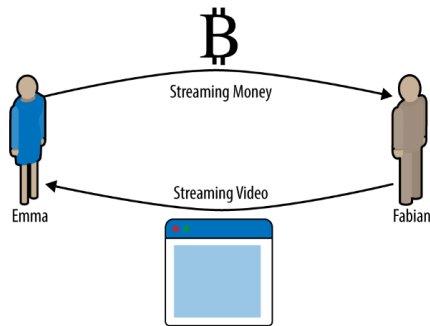


Figure 5. Emma purchases streaming video from Fabian with a payment channel, paying for each second of video

In this example, Fabian and Emma are using special software that handles both the payment channel and the video streaming. Emma is running the software in her browser, Fabian is running it on a server. The software includes basic bitcoin wallet functionality and can create and sign bitcoin transactions. Both the concept and the term "payment channel" are completely hidden from the users. What they see is video that is paid for by the second.

在这个例子中，Fabian和Emma使用特殊的软件，它处理这个支付通道和视频流。

Emma在她的浏览器中运行该软件，Fabian在他的服务器上运行该软件。

该软件包括基本的比特币钱包功能，可以创建和签署比特币交易。

“支付通道”的概念和术语对于用户都是完全不可见的，他们看到的是按秒支付的视频。

To set up the payment channel, Emma and Fabian establish a 2-of-2 multisignature address, with each of them holding one of the keys. From Emma's perspective, the software in her browser presents a QR code with a P2SH address (starting with "3"), and asks her to submit a "deposit" for up to 1 hour of video. The address is then funded by Emma. Emma's transaction, paying to the multisignature address, is the funding or anchor transaction for the payment channel.

为了建立这个支付通道，Emma和Fabian建立了一个2-2多签名地址，双方各持一个密钥。

从Emma的角度来看，她的浏览器中的软件提供了一个带有P2SH地址的二维码（以“3”开头），要求她提交最多1小时视频的“押金”。因而该地址得到了Emma的注资。Emma的交易（支付给这个多签名地址）是这个支付通道的资金交易或锚点交易。

For this example, let's say that Emma funds the channel with 36 millibits (0.036 BTC). This will allow Emma to consume *up to* 1 hour of streaming video. The funding transaction in this case sets the maximum amount that can be transmitted in this channel, setting the *channel capacity*.

就这个例子而言，我们说Emma支付了36个毫比（0.036 BTC）到这个通道中。

这将允许Emma消费最多1小时的视频流。

这个资金交易设定了可以在这个通道上发送的最大数量，即设置了“通道容量”。

The funding transaction consumes one or more inputs from Emma's wallet, sourcing the funds. It creates one output with a value of 36 millibits paid to the multisignature 2-of-2 address controlled jointly between Emma and Fabian. It may have additional outputs for change back to Emma's wallet.

这个资金交易从Emma的钱包中消耗一个或多个输入，以筹集资金。

它创建一个价值为36毫比的输出，支付给Emma和Fabian共同控制的多签名2-2地址。

它也可能有一个额外输出，用于找零钱给Emma的钱包。

Once the funding transaction is confirmed, Emma can start streaming video. Emma's software creates and signs a commitment transaction that changes the channel balance to credit 0.01 millibit to Fabian's address and refund 35.99 millibits back to Emma. The transaction signed by Emma consumes the 36 millibits output created by the funding transaction and creates two outputs: one for her refund, the other for Fabian's payment. The transaction is only partially signed—it requires two signatures (2-of-2), but only has Emma's signature. When Fabian's server receives this transaction, it adds the second

signature (for the 2-of-2 input) and returns it to Emma together with 1 second worth of video. Now both parties have a fully signed commitment transaction that either can redeem, representing the correct up-to-date balance of the channel. Neither party broadcasts this transaction to the network.

一旦资金交易得到确认，Emma可以开始观看视频。

Emma的软件创建并签名一笔承诺交易，改变通道余额，将0.01毫比归入Fabian的地址，并退回给Emma的35.99毫比。Emma签署的交易消耗了由资金交易创造的36毫比输出，并创建了两个输出：一个用于找钱，另一个用于付款给Fabian。

交易只是部分被签了，它需要两个签名（2 - 2），但只有Emma的签名。

当Fabian的服务器接收到此交易时，它会添加第二个签名（用于2-2输入），并将其返回给Emma并附带时长1秒的视频。

现在双方都有谁都可以兑换的完全签署的承诺交易，这个承诺交易代表着通道中的最新正确余额。

双方还不会将此交易广播到网络中。

In the next round, Emma's software creates and signs another commitment transaction (commitment #2) that consumes the *same* 2-of-2 output from the funding transaction. The second commitment transaction allocates one output of 0.02 millibits to Fabian's address and one output of 35.98 millibits back to Emma's address. This new transaction is payment for two cumulative seconds of video. Fabian's software signs and returns the second commitment transaction, together with another second of video.

在下一轮，Emma的软件创建并签署第二个承诺交易（承诺#2），该交易从资金交易中消耗相同的2-2输出。#2承诺交易分配一个0.2毫比的输出给Fabian的地址，还有一个输出为35.98毫比，作为找零返回给Emma的地址。

这个新交易支付的是连续两秒的视频内容。Fabian的软件签署并返回第二个承诺交易，再加上视频的下一秒内容。

In this way, Emma's software continues to send commitment transactions to Fabian's server in exchange for streaming video. The balance of the channel gradually accumulates in favor of Fabian, as Emma consumes more seconds of video. Let's say Emma watches 600 seconds (10 minutes) of video, creating and signing 600 commitment transactions. The last commitment transaction (#600) will have two outputs, splitting the balance of the channel, 6 millibits to Fabian and 30 millibits to Emma.

利用上述的方法，Emma的软件继续向Fabian的服务器发送承诺交易，以换取视频流。

因为Emma观看了更多秒数的视频，通道中属于Fabian的钱逐渐增加。

假设Emma观看600秒的视频，创建和签署600笔承诺交易。

最后的承诺交易（#600）有两个输出，将通道的余额分成两笔，6毫比属于Fabian，30毫比属于Emma。

Finally, Emma clicks "Stop" to stop streaming video. Either Fabian or Emma can now transmit the final state transaction for settlement. This last transaction is the *settlement transaction* and pays Fabian for all the video Emma consumed, refunding the remainder of the funding transaction to Emma.

最后，Emma点击“Stop”停止视频流。

Fabian或Emma现在可以发送最终的状态交易，以进行结算。

最后一笔交易即为“结算交易”，向Fabian支付视频服务费，并将余额退还给Emma。

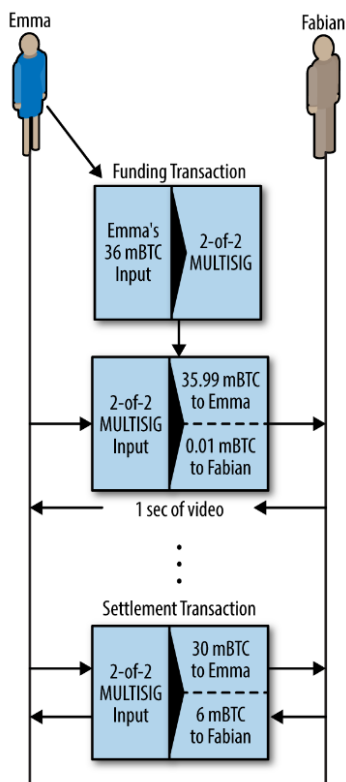
[Emma's payment channel with Fabian, showing the commitment transactions that update the balance of the channel](#) shows the channel between Emma and Fabian and the commitment transactions that update the balance of the channel.

图12-6显示了Emma和Fabian之间的通道，以及更新通道余额的承诺交易。

In the end, only two transactions are recorded on the blockchain: the funding transaction that established the channel and a settlement transaction that allocated the final balance correctly between the two participants.

最后，只有两个交易记录在区块链上：

- 建立通道的资金交易
- 在两个参与者之间正确分配最终余额的结算交易



### 12.6.3 尝试无需信任的通道

The channel we just described works, but only if both parties cooperate, without any failures or attempts to cheat. Let's look at some of the scenarios that break this channel and see what is needed to fix those:

我们刚刚描述的通道可以工作，但只有在双方合作的情况下才行，没有任何失败或企图欺骗。

我们来看看破坏这个通道的一些场景，并且看看需要什么来修补这些场景：

- Once the funding transaction happens, Emma needs Fabian's signature to get any money back. If Fabian disappears, Emma's funds are locked in a 2-of-2 and effectively lost. This channel, as constructed, leads to a loss of funds if one of the parties disconnects before there is at least one commitment transaction signed by both parties.  
一旦资金交易发生了，Emma需要Fabian的签名才能收回任何钱。  
如果Fabian消失，Emma的资金将被锁定在2-2中，并实际丧失了。  
这个通道一旦建立，在双方共同签署至少一个承诺交易之前，如果任何一方断开，就会导致资金的流失。
- While the channel is running, Emma can take any of the commitment transactions Fabian has countersigned and transmit one to the blockchain. Why pay for 600 seconds of video, if she can transmit commitment transaction #1 and only pay for 1 second of video? The channel fails because Emma can cheat by broadcasting a prior commitment that is in her favor.  
当这个通道在运行时，Emma可以使用Fabian已经签署的任何承诺交易，并发送到区块链上。  
如果她可以发送承诺交易 # 1，只用支付1秒的钱，那为什么还要支付600秒的钱呢？  
这个通道是失败的，因为Emma可以通过广播对她比较有利的先前的承诺来欺骗。

Both of these problems can be solved with timelocks—let's look at how we could use transaction-level timelocks (nLocktime).

这两个问题都可以用时间锁(timelocks)来解决，

我们来看看我们如何使用交易级时间锁（nLocktime）。

Emma cannot risk funding a 2-of-2 multisig unless she has a guaranteed refund. To solve this problem, Emma constructs the funding and refund transaction at the same time. She signs the funding transaction but doesn't transmit it to anyone. Emma transmits only the refund transaction to Fabian and obtains his signature.

除非Emma可以获得退款保证，否则她不能冒险进行2-2 签名。

为了解决这个问题，Emma同时建立了资金交易和退款交易。

她签署资金交易，但不发送给任何人。

Emma只将退款交易传送给Fabian，并获得他的签名。

The refund transaction acts as the first commitment transaction and its timelock establishes the upper bound for the channel's life. In this case, Emma could set the nLocktime to 30 days or 4320 blocks into the future. All subsequent commitment transactions must have a shorter timelock, so that they can be redeemed before the refund transaction.

退款交易作为第一个承诺交易，其时间锁确立了这个通道的生命上限。

在本例中，Emma可以将nLocktime设置为30天或将来的第4320个区块。

所有后续承诺交易必须具有较短的时间锁，以便在退款交易之前能把它们赎回。

Now that Emma has a fully signed refund transaction, she can confidently transmit the signed funding transaction knowing that she can eventually, after the timelock expires, redeem the refund transaction even if Fabian disappears.

既然Emma已经有一个完全签名的退款交易，她可以自信地发送签署过的资金交易，因为她知道，即使Fabian消失也不会有问题，因为她可以在时间到期后赎回退款。

Every commitment transaction the parties exchange during the life of the channel will be timelocked into the future. But the delay will be slightly shorter for each commitment so the most recent commitment can be redeemed before the prior commitment it invalidates. Because of the nLockTime, neither party can successfully propagate any of the commitment transactions until their timelock expires. If all goes well, they will cooperate and close the channel gracefully with a settlement transaction, making it unnecessary to transmit an intermediate commitment transaction. If not, the most recent commitment transaction can be propagated to settle the account and invalidate all prior commitment transactions.

在通道生命期中，双方交换的每个承诺交易都会被时间锁定到未来。

但是，每个承诺的延迟时间都更短一些，所以最新的承诺可以在被它废止的前一承诺之前被赎回。

由于nLocktime，任何一方都只有在时间到期后才能成功传播任何承诺交易。

如果一切顺利，他们将合作，并通过结算交易优雅地关闭通道，这样就没有必要发送中间的承诺交易了。

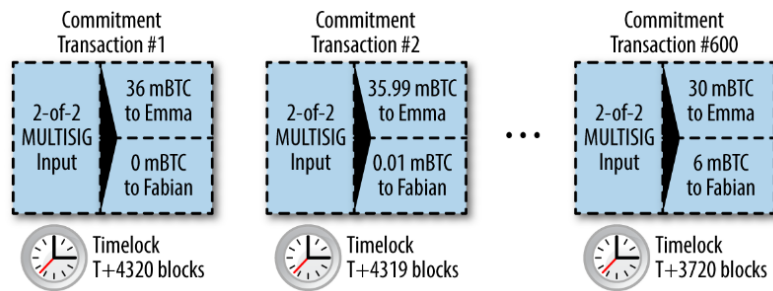
如果没有，可以传播最近的承诺交易来结算账户，并使所有先前承诺交易无效。

For example, if commitment transaction #1 is timelocked to 4320 blocks in the future, then commitment transaction #2 is timelocked to 4319 blocks in the future. Commitment transaction #600 can be spent 600 blocks before commitment transaction #1 becomes valid.

例如，如果将来承诺交易 # 1被时间锁定到未来的第4320个区块，则承诺交易 # 2被锁定到将来的4319个区块。可以在承诺交易#1有效之前的600个区块，花费承诺交易 # 600。

[Each commitment sets a shorter timelock, allowing it to be spent before the previous commitments become valid](#) shows each commitment transaction setting a shorter timelock, allowing it to be spent before the previous commitments become valid.

图12-7显示每个承诺交易设置一个更短的时间锁，允许在它在之前的承诺变为有效前被花费。



Each subsequent commitment transaction must have a shorter timelock so that it may be broadcast before its predecessors and before the refund transaction. The ability to broadcast a commitment earlier ensures it will be able to spend the funding output and preclude any other commitment transaction from being redeemed by spending the output. The guarantees offered by the bitcoin blockchain, preventing double-spends and enforcing timelocks, effectively allow each commitment transaction to invalidate its predecessors. 每个后续的承诺交易必须有一个更短的时间锁，这样，就可以在其前一承诺交易之前和退款交易之前进行广播。

能够尽早广播一个承诺交易的能力，确保了能够花费资金输出，并阻止其它承诺交易被赎回。

比特币区块链提供的保证（即防止双重支付和执行时间锁定），有效地允许每个承诺交易废止了其前一承诺交易的有效性。

State channels use timelocks to enforce smart contracts across a time dimension. In this example we saw how the time dimension guarantees that the most recent commitment transaction becomes valid before any earlier commitments. Thus, the most recent commitment transaction can be transmitted, spending the inputs and invalidating prior commitment transactions. The enforcement of smart contracts with absolute timelocks protects against cheating by one of the parties. This implementation needs nothing more than absolute transaction-level timelocks (nLocktime). Next, we will see how script-level timelocks, CHECKLOCKTIMEVERIFY and CHECKSEQUENCEVERIFY, can be used to construct more flexible, useful, and sophisticated state channels.

状态通道使用时间锁来在时间维度上执行智能合约。

在这个例子中，我们看到时间维度如何保证最近的承诺交易在任何早先的承诺之前变得有效。

因此，最近的承诺交易可以发送、消费输入，并使先前的承诺交易无效。

使用绝对时间锁执行智能合约可以防止其中任何一方的欺骗。

这个实现只需要绝对的交易级时间锁（nLocktime）。

接下来，我们将看到如何使用脚本级时间锁（CHECKLOCKTIMEVERIFY和CHECKSEQUENCEVERIFY），可用来构建更灵活、有用和复杂的状态通道。

The first form of unidirectional payment channel was demonstrated as a prototype video streaming application in 2015 by an Argentinian team of developers. You can still see it at [streamium.io](https://streamium.io).

第一种形式的单向支付通道由一个阿根廷开发团队在2015年作为一个原型视频流应用被演示。

你仍然可以在streamium.io看到它。

Timelocks are not the only way to invalidate prior commitment transactions. In the next sections we will see how a revocation key can be used to achieve the same result. Timelocks are effective but they have two distinct disadvantages. By establishing a maximum timelock when the channel is first opened, they limit the lifetime of the channel. Worse, they force channel implementations to strike a balance between allowing long-lived channels and forcing one of the participants to wait a very long time for a refund in case of premature closure. For example, if you allow the channel to remain open for 30 days, by setting the refund timelock to 30 days, if one of the parties disappears immediately the other party must wait 30 days for a refund. The more distant the endpoint, the more distant the refund.

时间锁并不是使先前的承诺交易无效的唯一方法。

在接下来的章节中，我们将看到，如何使用撤销密钥来实现相同的效果。

时间锁是有效的，但有两个明显的缺点。

在通道首次打开时建立了最大时间锁，它们限制了通道的生命周期。



更糟糕的是，他们迫使通道实现达成一个平衡：在允许长期存在的通道，和迫使一位参与者在提前关闭的情况下等待很长时间才能收回退款。

例如，如果允许通道保持开放30天（将退款时间设置为30天），如果其中一方立即消失，则另一方必须等待30天才能收回退款。终点设置越远，退款时间越长。

The second problem is that since each subsequent commitment transaction must decrement the timelock, there is an explicit limit on the number of commitment transactions that can be exchanged between the parties. For example, a 30-day channel, setting a timelock of 4320 blocks into the future, can only accommodate 4320 intermediate commitment transactions before it must be closed. There is a danger in setting the timelock commitment transaction interval at 1 block. By setting the timelock interval between commitment transactions to 1 block, a developer is creating a very high burden for the channel participants who have to be vigilant, remain online and watching, and be ready to transmit the right commitment transaction at any time.

第二个问题是，由于每个后续的承诺交易必须减短时间锁，所以在双方之间可以交换的承诺交易数量有明确的限制。

例如，一个30天的通道，设置了位于未来第4320个块的时间锁，在必须被关闭前只能容纳4320个中间承诺交易。

将时间锁定承诺交易的间隔设置为1个区块存在危险。通过将承诺交易之间的时间锁设置为1个区块，开发者给通道参与者带来了非常高的负担，参与者必须保持警惕，保持在线并监视，并随时准备传送正确的承诺交易。

Now that we understand how timelocks can be used to invalidate prior commitments, we can see the difference between closing the channel cooperatively and closing it unilaterally by broadcasting a commitment transaction. All commitment transactions are timelocked, therefore broadcasting a commitment transaction will always involve waiting until the timelock has expired. But if the two parties agree on what the final balance is and know they both hold commitment transactions that will eventually make that balance a reality, they can construct a settlement transaction without a timelock representing that same balance. In a cooperative close, either party takes the most recent commitment transaction and builds a settlement transaction that is identical in every way except that it omits the timelock. Both parties can sign this settlement transaction knowing there is no way to cheat and get a more favorable balance. By cooperatively signing and transmitting the settlement transaction they can close the channel and redeem their balance immediately. Worst case, one of the parties can be petty, refuse to cooperate, and force the other party to do a unilateral close with the most recent commitment transaction. But if they do that, they have to wait for their funds too.

既然我们理解了如何使用时间锁来使先前的承诺无效，我们可以看到合作关闭通道和通过广播承诺交易单方面关闭通道之间的区别。

所有承诺交易都被时间锁定，因此，广播承诺交易总是要等待时间到期。

但是，如果双方同意最后的余额是多少，并且知道他们都持有最终实现余额的承诺交易，那么他们可以构建一个没有时间锁代表相同余额的结算交易。

在合作关闭中，任一方都可以提取最近的承诺交易，并建立一个各方面完全相同的结算交易，唯一差别就是结算交易省略了时间锁。双方都可以签署这笔结算交易，因为知道无法作弊以得到更多的余额。通过合作签署和发送结算交易，可以立即关闭通道并兑换余额。

最差情况下，一方可能是卑鄙小人，拒绝合作，迫使另一方用最近的承诺交易单方面关闭。但是如果他们这样做，他们也必须等待他们的资金。

## 12.6.4非对称的可撤销承诺

A better way to handle the prior commitment states is to explicitly revoke them. However, this is not easy to achieve. A key characteristic of bitcoin is that once a transaction is valid, it remains valid and does not expire. The only way to cancel a transaction is by double-spending its inputs with another transaction before it is mined. That's why we used timelocks in the simple payment channel example above to ensure that more recent commitments could be spent before older commitments were valid. However, sequencing

commitments in time creates a number of constraints that make payment channels difficult to use.

处理先前承诺状态的更好方法是明确地撤销它们。但是，这不容易实现。

比特币的一个关键特征是，一旦交易有效，它一直有效，不会过期。

取消交易的唯一方法是，在交易被挖矿前用另一笔交易双重支付它的输入。

这就是为什么我们在上述简单支付通道示例中使用时间锁定，以确保最新的承诺交易可以在旧承诺生效之前被花费。然而，把承诺在时间上排序造成了许多限制，使得支付通道难以使用。

Even though a transaction cannot be canceled, it can be constructed in such a way as to make it undesirable to use. The way we do that is by giving each party a *revocation* key that can be used to punish the other party if they try to cheat. This mechanism for revoking prior commitment transactions was first proposed as part of the Lightning Network.

虽说一个交易无法被取消，但是它可以被构造成不想再使用的样子。

方法是给每一方一个撤销密钥，如果他们试图欺骗，可以用来进行惩罚另一方。

这个撤销先前承诺交易的机制，是作为闪电网络的一部分被首先提出。

To explain revocation keys, we will construct a more complex payment channel between two exchanges run by Hitesh and Irene. Hitesh and Irene run bitcoin exchanges in India and the USA, respectively. Customers of Hitesh's Indian exchange often send payments to customers of Irene's USA exchange and vice versa. Currently, these transactions occur on the bitcoin blockchain, but this means paying fees and waiting several blocks for confirmations. Setting up a payment channel between the exchanges will significantly reduce the cost and accelerate the transaction flow.

为了解释撤销密钥，我们将在由Hitesh和Irene经营的两个交易所之间构建一个更复杂的支付通道。

Hitesh和Irene分别在印度和美国运营比特币交易所。

Hitesh的客户与Irene的客户之间经常相互发送付款。

目前，这些交易都发生在比特币区块链上，但这意味着支付小费，并等待几个区块确认。

在交易所之间设置支付通道将大大降低成本，并加快交易流程。

Hitesh and Irene start the channel by collaboratively constructing a funding transaction, each funding the channel with 5 bitcoin. The initial balance is 5 bitcoin for Hitesh and 5 bitcoin for Irene. The funding transaction locks the channel state in a 2-of-2 multisig, just like in the example of a simple channel.

Hitesh和Irene通过合作构建一个建立资金交易来启动这个通道，每一方向这个通道注资5个比特币。

初始余额为，Hitesh有5比特币，Irene有5比特币。

这个资金交易将通道状态锁定在2-2多签名中，就像在简单通道的例子中一样。

The funding transaction may have one or more inputs from Hitesh (adding up to 5 bitcoin or more), and one or more inputs from Irene (adding up to 5 bitcoin or more). The inputs have to slightly exceed the channel capacity in order to cover the transaction fees. The transaction has one output that locks the 10 total bitcoin to a 2-of-2 multisig address controlled by both Hitesh and Irene. The funding transaction may also have one or more outputs returning change to Hitesh and Irene if their inputs exceeded their intended channel contribution. This is a single transaction with inputs offered and signed by two parties. It has to be constructed in collaboration and signed by each party before it is transmitted.

这个资金交易可能有一个或多个来自Hitesh的输入（加起来5个比特币或更多），以及Irene的一个或多个输入（加起来5个比特币或更多）。

输入必须略微超过通道容量，才够支付交易费用。该交易有一个输出，它将10个比特币锁定到由Hitesh和Irene控制的2-2多签名地址。

如果他们的输入超过他们需要贡献的数值，资金交易也可能有一个或多个输出将找零返回给Hitesh和Irene。

这是由双方提供和签署的多个输入形成的单一交易。

在发送之前，它必须被合作构建，并且由各方签署。

Now, instead of creating a single commitment transaction that both parties sign, Hitesh and Irene create two different commitment transactions that are *asymmetric*.

现在，不是创建一个双方都签名的承诺交易，而是Hitesh和Irene创造了两个不对称的承诺交易。

Hitesh has a commitment transaction with two outputs. The first output pays Irene the 5 bitcoin she is owed *immediately*. The second output pays Hitesh the 5 bitcoin he is owed, but only after a timelock of 1000 blocks. The transaction outputs look like this:

Hitesh有一个承诺交易，它有两个输出。

第一个输出立即支付他欠Irene的5比特币。

第二个输出支付他欠Hitesh的5比特币，但条件是只有在1000个区块的时间锁之后。

交易输出如下所示：

```
Input: 2-of-2 funding output, signed by Irene
```

```
Output 0 <5 bitcoin>:
```

```
<Irene's Public Key> CHECKSIG
```

```
Output 1:
```

```
<1000 blocks>
```

```
CHECKSEQUENCEVERIFY
```

```
DROP
```

```
<Hitesh's Public Key> CHECKSIG
```

Irene has a different commitment transaction with two outputs. The first output pays Hitesh the 5 bitcoin he is owed immediately. The second output pays Irene the 5 bitcoin she is owed but only after a timelock of 1000 blocks. The commitment transaction Irene holds (signed by Hitesh) looks like this:

Irene也有一个承诺交易，它有两个输出。

第一个输出支付他欠Hitesh的5比特币。

第二个输出支付他欠Irene的的比特币，但同样有经过1000个区块的时间锁。

Irene持有的承诺交易（由Hitesh签署）看起来是这样：

```
Input: 2-of-2 funding output, signed by Hitesh
```

```
Output 0<5 bitcoin>:
```

```
<Hitesh's Public Key> CHECKSIG
```

```
Output 1:
```

```
<1000 blocks>
```

```
CHECKSEQUENCEVERIFY
```

```
DROP
```

```
<Irene's Public Key> CHECKSIG
```

This way, each party has a commitment transaction, spending the 2-of-2 funding output. This input is signed by the *other* party. At any time the party holding the transaction can also sign (completing the 2-of-2) and broadcast. However, if they broadcast the commitment transaction, it pays the other party immediately whereas they have to wait for a short timelock to expire. By imposing a delay on the redemption of one of the outputs, we put each party at a slight disadvantage when they choose to unilaterally broadcast a commitment transaction. But a time delay alone isn't enough to encourage fair conduct.

这样，双方各有一个承诺交易，花费2-2的资金输出。这个输入是由对方签名。

在任何时候，持有承诺交易的一方也可以签名（完成2-2签名）并进行广播。

然而，如果他们广播承诺交易，承诺交易会立即支付对方，而他们自己必须等待短时间锁到期。

通过在其中一个输出上强制执行赎回延迟，我们可以做到让各方在选择单方面广播承诺交易时处于轻微的不利地位。但是，只靠时间延迟还不足以鼓励公平的行为。

[Two asymmetric commitment transactions with delayed payment for the party holding the transaction](#) shows two asymmetric commitment transactions, where the output paying the holder of the commitment is delayed.

下图12-8显示两个不对称承诺交易，其中，支付给承诺持有人的输出被延迟了。



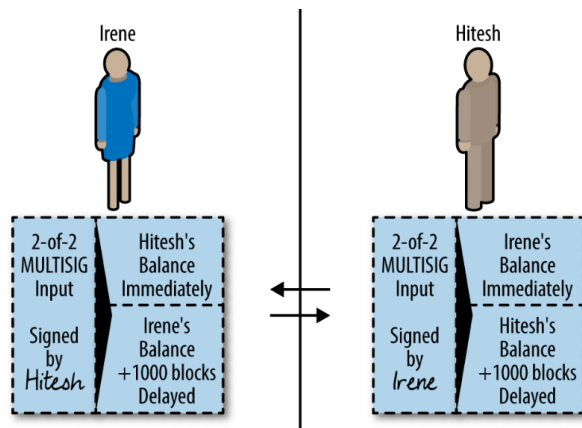


Figure 8. Two asymmetric commitment transactions with delayed payment for the party holding the transaction

Now we introduce the final element of this scheme: a revocation key that prevents a cheater from broadcasting an expired commitment. The revocation key allows the wronged party to punish the cheater by taking the entire balance of the channel.

现在我们介绍这个方案的最后一个要素：一个撤销密钥，它防止欺骗者广播一个过期的承诺。

撤销密钥允许被伤害的一方通过获取通道的所有余额来惩罚骗子。

The revocation key is composed of two secrets, each half generated independently by each channel participant. It is similar to a 2-of-2 multisig, but constructed using elliptic curve arithmetic, so that both parties know the revocation public key but each party knows only half the revocation secret key.

撤销密钥由两个秘密组成，每一半是由每个通道参与者独立生成的。

它类似于2-2多签名，但使用椭圆曲线算法构造，这样，双方都知道这个撤销公钥，但每一方都只知道撤销私钥的一半。

In each round, both parties reveal their half of the revocation secret to the other party, thereby giving the other party (who now has both halves) the means to claim the penalty output if this revoked transaction is ever broadcast.

在每一轮中，双方都将其撤销秘密的一半透露给另一方，从而让另一方能够要求罚金输出，如果该撤销的交易被广播。

Each of the commitment transactions has a "delayed" output. The redemption script for that output allows one party to redeem it after 1000 blocks, or the other party to redeem it if they have a revocation key, penalizing transmission of a revoked commitment.

每个承诺交易都有一个“延迟”的输出。

该输出的兑换脚本允许一方在1000个区块之后兑换它，或者另一方如果拥有撤销密钥也可兑换它，惩罚传输被撤销的承诺。。

So when Hitesh creates a commitment transaction for Irene to sign, he makes the second output payable to himself after 1000 blocks, or to the revocation public key (of which he only knows half the secret). Hitesh constructs this transaction. He will only reveal his half of the revocation secret to Irene when he is ready to move to a new channel state and wants to revoke this commitment.

所以，当Hitesh为Irene签署承诺交易时，他将把第二个输出定义为在1000块之后可输出支付给自己，或者是任何可以出示撤销密钥的人。

Hitesh构建了这个交易。当他准备转移到新的通道状态，并希望撤销这一承诺时，他只透漏撤销密钥的他的一半给Irene。

The second output's script looks like this:

第二个输出的脚本如下：

```
Output 0<5 bitcoin>:
  <Irene's Public Key> CHECKSIG
```

```
Output 1<5 bitcoin>:
IF # Revocation penalty output
  <Revocation Public Key>
ELSE
  <1000 blocks>
  CHECKSEQUENCEVERIFY
  DROP
  <Hitesh's Public Key>
ENDIF
CHECKSIG
```

Irene can confidently sign this transaction, since if transmitted it will immediately pay her what she is owed. Hitesh holds the transaction, but knows that if he transmits it in a unilateral channel closing, he will have to wait 1000 blocks to get paid.

Irene可以自信地签署这笔交易，因为一旦被发送它将立即支付她被欠的欠款。

Hitesh持有这个交易，但知道，如果他在单方通道关闭时发送这个交易，他必须等待1000个区块才能获得支付。

When the channel is advanced to the next state, Hitesh has to *revoke* this commitment transaction before Irene agrees to sign the next commitment transaction. To do that, all he has to do is send his half of the *revocation* key to Irene. Once Irene has both halves of the revocation secret key for this commitment, she can sign the next commitment with confidence. She knows that if Hitesh tries to cheat by publishing the prior commitment, she can use the revocation key to redeem Hitesh's delayed output. *If Hitesh cheats, Irene gets BOTH outputs.* Meanwhile, Hitesh only has half the revocation secret for that revocation public key and can't redeem the output until 1000 blocks. Irene will be able to redeem the output and punish Hitesh before the 1000 blocks have elapsed.

当这个通道进入下一个状态时，Hitesh必须在Irene同意签署下一个承诺交易之前撤销此承诺交易。要做到这一点，他所要做的就是将撤销密钥发送给Irene。一旦Irene拥有这一承诺的撤销密钥，她就可以自信地签署下一个承诺。她知道，如果Hitesh试图通过发布先前的承诺交易来作弊，她可以使用撤销密钥来兑换Hitesh的延迟输出。如果Hitesh作弊，Irene会得到BOTH（两方）输出。

The revocation protocol is bilateral, meaning that in each round, as the channel state is advanced, the two parties exchange new commitments, exchange revocation secrets for the previous commitments, and sign each other's new commitment transactions. As they accept a new state, they make the prior state impossible to use, by giving each other the necessary revocation secrets to punish any cheating.

撤销协议是双边的，这意味着在每一轮中，随着通道状态的进一步发展，双方交换新的承诺，交换用于之前承诺的撤销密钥，并签署彼此的承诺交易。当他们接受新的状态时，他们通过给予对方必要的撤销密钥来惩罚任何作弊行为，使先前的状态不可能再被使用。

Let's look at an example of how it works. One of Irene's customers wants to send 2 bitcoin to one of Hitesh's customers. To transmit 2 bitcoin across the channel, Hitesh and Irene must advance the channel state to reflect the new balance. They will commit to a new state (state number 2) where the channel's 10 bitcoin are split, 7 bitcoin to Hitesh and 3 bitcoin to Irene. To advance the state of the channel, they will each create new commitment transactions reflecting the new channel balance.

我们来看一个它的工作例子。Irene的客户之一希望向Hitesh的客户发送2比特币。要通过通道传输2比特币，Hitesh和Irene必须更新通道状态以反映新的余额。他们将承诺一个新的状态（状态号2），通道的10个比特币分裂，7个比特币属于Hitesh和3个比特币属于Irene。为了更新通道的状态，他们将各自创建反映新通道余额的新承诺交易。

As before, these commitment transactions are asymmetric so that the commitment transaction each party holds forces them to wait if they redeem it. Crucially, before signing new commitment transactions, they must first exchange revocation keys to invalidate the prior commitment. In this particular case, Hitesh's interests are aligned with the real state of the channel and therefore he has no reason to broadcast a prior state. However, for Irene, state number 1 leaves her with a higher balance than state 2. When Irene gives Hitesh the revocation key for her prior commitment transaction (state number 1) she is effectively revoking her ability to profit from regressing the channel to a prior state

because with the revocation key, Hitesh can redeem both outputs of the prior commitment transaction without delay. Meaning if Irene broadcasts the prior state, Hitesh can exercise his right to take all of the outputs.

如上述内容所说，这些承诺交易是不对称的，所以每一方所持的承诺交易都迫使他们等待兑换。至关重要的是，在签署新的承诺交易之前，他们必须首先交换撤销密钥以使先前的承诺无效。在这种情况下，Hitesh的利益与通道的真实状态是一致的，因此他没有理由广播先前的状态。然而，对于Irene来说，状态号1中留给她的余额比状态2中的更高。当Irene给予Hitesh她以前的承诺交易（状态号1）的撤销密钥时，她实际上废除了自己可以回滚通道状态到前一状态而从中获益的能力。因为有了撤销密钥，Hitesh可以毫不拖延地兑换先前承诺交易的两个输出。也就是说一旦Irene广播先前的状态，Hitesh可以行使其占有所有输出的权利。

Importantly, the revocation doesn't happen automatically. While Hitesh has the ability to punish Irene for cheating, he has to watch the blockchain diligently for signs of cheating. If he sees a prior commitment transaction broadcast, he has 1000 blocks to take action and use the revocation key to thwart Irene's cheating and punish her by taking the entire balance, all 10 bitcoin.

重要的是，撤销不会自动发生。虽然Hitesh有能力惩罚Irene的作弊行为，但他必须勤勉地观察区块链中作弊的迹象。如果他看到先前的承诺交易广播，他有1000个区块时间采取行动，并使用撤销密钥来阻止Irene的欺骗行为并占有所有余额也就是全部10比特币来惩罚她。

Asymmetric revocable commitments with relative time locks (CSV) are a much better way to implement payment channels and a very significant innovation in this technology. With this construct, the channel can remain open indefinitely and can have billions of intermediate commitment transactions. In prototype implementations of Lightning Network, the commitment state is identified by a 48-bit index, allowing more than 281 trillion ( $2.8 \times 10^{14}$ ) state transitions in any single channel!

带有相对时间锁（csv）的不对称可撤销承诺是实现支付通道的更好方法，也是区块链技术非常重要的创新。通过这种结构，通道可以无限期地保持开放，并且可以拥有数十亿的中间承诺交易。在闪电网络的原型实现中，承诺状态由48位索引识别，允许在任何单个通道中有超过281兆（ $2.8 \times 10^{14}$ ）个状态转换！

## 12.6.5 哈希时间锁合约（HTLC）

Payment channels can be further extended with a special type of smart contract that allows the participants to commit funds to a redeemable secret, with an expiration time. This feature is called a *Hash Time Lock Contract*, or *HTLC*, and is used in both bidirectional and routed payment channels.

支付通道可以通过特殊类型的智能合约进一步扩展，以允许参与者将资金用于可赎回的具有到期时间的秘密（secret）。此功能称为哈希时间锁定合约或HTLC，并用于双向和路由的支付通道。

Let's first explain the "hash" part of the HTLC. To create an HTLC, the intended recipient of the payment will first create a secret R. They then calculate the hash of this secret H:

首先我们来解释HTLC的“哈希”部分。要创建一个HTLC，预期的收款人将首先创建一个秘密（secret）R。他们然后计算这个R的哈希H：

$$H = \text{Hash}(R)$$

This produces a hash H that can be included in an output's locking script. Whoever knows the secret can use it to redeem the output. The secret R is also referred to as a *preimage* to the hash function. The preimage is just the data that is used as input to a hash function.

这步产生可以包含在输出的锁定脚本中的哈希H。知道秘密的任何人可以用它来兑换输出。秘密R也被称为哈希函数的前图像。前图像就是用作哈希函数输入的数据。

The second part of an HTLC is the "time lock" component. If the secret is not revealed, the payer of the HTLC can get a "refund" after some time. This is achieved with an absolute time lock using CHECKLOCKTIMEVERIFY.

HTLC的第二部分是“时间锁”组件。如果秘密没有被透露，HTLC的付款人可以在一段时间后得到“退款”。这是通过使用绝对时间锁CHECKLOCKTIMEVERIFY来实现的。实现HTLC的脚本可能如下所示：

The script implementing an HTLC might look like this:

```
IF
  # Payment if you have the secret R
  HASH160 <H> EQUALVERIFY
ELSE
  # Refund after timeout.
  <locktime>
  CHECKLOCKTIMEVERIFY DROP
  <Payee Pubic Key> CHECKSIG
ENDIF
```

Anyone who knows the secret R, which when hashed equals to H, can redeem this output by exercising the first clause of the IF flow.

任何知道可以让哈希等于h的对应秘密R的人，可以通过行使IF语句的第一个子句来兑换该输出。

If the secret is not revealed and the HTLC claimed, after a certain number of blocks the payer can claim a refund using the second clause in the IF flow.

如果秘密没有被透露，HTLC中写明了，在一定数量的块之后，收款人可以使用IF语句中的第二个子句申请退款。

This is a basic implementation of an HTLC. This type of HTLC can be redeemed by *anyone* who has the secret R. An HTLC can take many different forms with slight variations to the script. For example, adding a CHECKSIG operator and a public key in the first clause restricts redemption of the hash to a named recipient, who must also know the secret R.

这是HTLC的基本实现。任何拥有秘密R的人都可以兑换这种类型的HTLC。通过对脚本进行微调，HTLC可以采用许多不同的形式。例如，在第一个子句中添加一个CHECKSIG运算符和一个公钥来限制将哈希值兑换成一个指定的收件人，这个人必须知道秘密R。

## 12.7 路由支付通道（闪电网络）

The Lightning Network is a proposed routed network of bidirectional payment channels connected end-to-end. A network like this can allow any participant to route a payment from channel to channel without trusting any of the intermediaries. The Lightning Network was [first described by Joseph Poon and Thadeus Dryja in February 2015](#), building on the concept of payment channels as proposed and elaborated upon by many others.

闪电网络是一种端到端连接的，双向支付通道的路由网络。

这样的网络可以允许任何参与者在通道之间路由一个支付，而不需要信任任何中间方。

闪电网络由Joseph Poon和Thadeus Dryja于2015年2月首次描述，其基础是许多其他人提出和阐述的“支付通道”概念。

"Lightning Network" refers to a specific design for a routed payment channel network, which has now been implemented by at least five different open source teams. The independent implementations are coordinated by a set of interoperability standards described in the [Basics of Lightning Technology \(BOLT\) paper](#).

“闪电网络”指的是路由支付通道网络的一个具体设计，现已由至少五个不同的开源团队实施。

这些的独立实施是由“闪电技术基础”（BOLT）论文中描述的一组互通性标准进行协作。

Prototype implementations of the Lightning Network have been released by several teams. For now, these implementations can only be run on testnet because they use segwit, which is not activated on the main bitcoin blockchain (mainnet).

闪电网络的原型实施已经由几个团队发布。

现在，这些实现只能在testnet上运行，因为它们使用segwit，还没有在比特币区块mainnet上激活。

The Lightning Network is one possible way of implementing routed payment channels. There are several other designs that aim to achieve similar goals, such as Teechan and Tumblebit.

闪电网络是实现路由支付通道的一种可能方式。

还有其他几种设计实现类似的目标，例如Teechan和Tumblebit。

### 12.7.1 闪电网络示例

Let's see how this works.

让我们看看它是如何工作的。

In this example, we have five participants: Alice, Bob, Carol, Diana, and Eric. These five participants have opened payment channels with each other, in pairs. Alice has a payment channel with Bob. Bob is connected to Carol, Carol to Diana, and Diana to Eric. For simplicity let's assume each channel is funded with 2 bitcoin by each participant, for a total capacity of 4 bitcoin in each channel.

在这个例子中，我们有五个参与者：Alice, Bob, Carol, Diana, Eric。

这五名参与者已经彼此之间开设了支付通道。

Alice和Bob有支付通道，Bob连接Carol，Carol连接到Diana，Diana连接Eric。

为了简单起见，我们假设每个通道每个参与者都注资2个比特币资金，每个通道的总容量为4个比特币。

[A series of bidirectional payment channels linked to form a Lightning Network that can route a payment from Alice to Eric](#) shows five participants in a Lightning Network, connected by bidirectional payment channels that can be linked to make a payment from Alice to Eric ([Routed Payment Channels \(Lightning Network\)](#)).

下图12-9显示一系列通过双向支付的通道连接在一起形成闪电网络以支持一笔从Alice到Eric的付款，展示了闪电网络中五名参与者，通过双向支付通道连接，可从Alice付款到Eric（路由支付通道（闪电网络））。

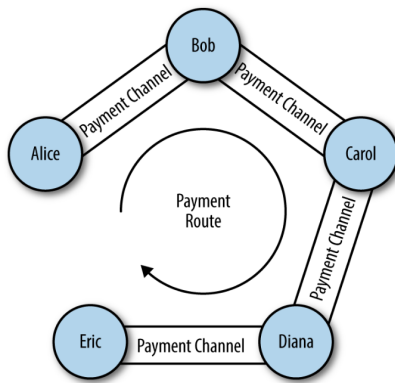


Figure 9. A series of bidirectional payment channels linked to form a Lightning Network that can route a payment from Alice to Eric

Alice wants to pay Eric 1 bitcoin. However, Alice is not connected to Eric by a payment channel. Creating a payment channel requires a funding transaction, which must be committed to the bitcoin blockchain. Alice does not want to open a new payment channel and commit more of her funds. Is there a way to pay Eric, indirectly?

Alice想要支付给Eric 1个比特币。不过，Alice并未通过支付通道连接到Eric。

创建支付通道需要资金交易，而这笔交易必须首先提交给比特币区块链。

Alice不想打开一个新的支付通道并支出更多的手续费。有没有办法间接支付Eric？

[Step-by-step payment routing through a Lightning Network](#) shows the step-by-step process of routing a payment from Alice to Eric, through a series of HTLC commitments on the payment channels connecting the participants.

下图12-10 显示了通过在连接各方参与者的支付通道上通过一系列HTLC承诺将付款从Alice路由到Eric的逐步过程。

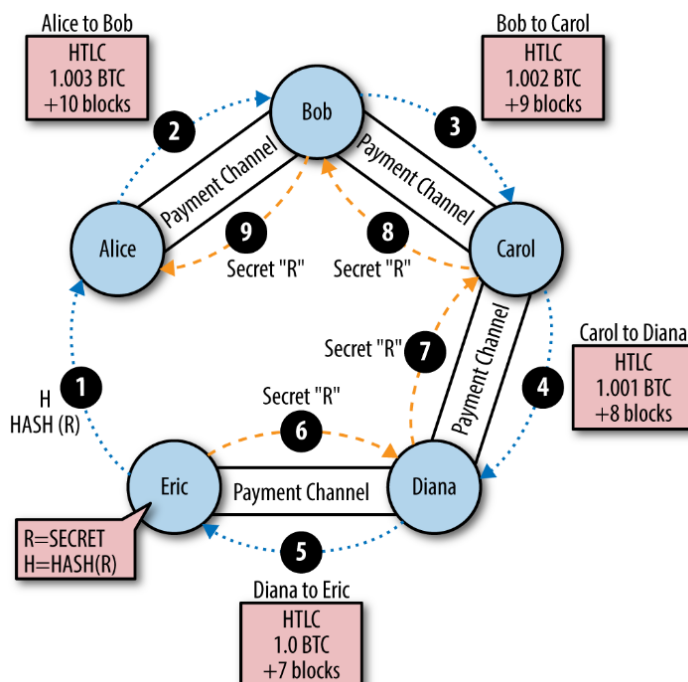


Figure 10. Step-by-step payment routing through a Lightning Network

Alice is running a Lightning Network (LN) node that is keeping track of her payment channel to Bob and has the ability to discover routes between payment channels. Alice's LN node also has the ability to connect over the internet to Eric's LN node. Eric's LN node creates a secret R using a random number generator. Eric's node does not reveal this secret to anyone. Instead, Eric's node calculates a hash H of the secret R and transmits



this hash to Alice's node (see [Step-by-step payment routing through a Lightning Network](#) step 1).

Alice正在运行闪电网络（LN）节点，该节点正在跟踪其向Bob的付费通道，并且能够发现支付通道之间的路由。Alice的LN节点还具有通过互联网连接到Eric的LN节点的能力。

Eric的LN节点使用随机数生成器创建一个秘密R。Eric的节点没有向任何人泄漏这个秘密。相反，Eric的节点计算秘密R对应的哈希H，并将此哈希发送到Alice的节点（请参阅图12-10步骤1）。

Now Alice's LN node constructs a route between Alice's LN node and Eric's LN node. The routing algorithm used will be examined in more detail later, but for now let's assume that Alice's node can find an efficient route.

现在Alice的LN节点构建了Alice的LN节点和Eric的LN节点之间的路由。所使用的路由算法将在后面进行更详细的解释，但现在我们假设Alice节点可以找到一个高效的路由。

Alice's node then constructs an HTLC, payable to the hash H, with a 10-block refund timeout (current block + 10), for an amount of 1.003 bitcoin (see [Step-by-step payment routing through a Lightning Network](#) step 2). The extra 0.003 will be used to compensate the intermediate nodes for their participation in this payment route. Alice offers this HTLC to Bob, deducting 1.003 bitcoin from her channel balance with Bob and committing it to the HTLC. The HTLC has the following meaning: *"Alice is committing 1.003 of her channel balance to be paid to Bob if Bob knows the secret, or refunded back to Alice's balance if 10 blocks elapse."* The channel balance between Alice and Bob is now expressed by commitment transactions with three outputs: 2 bitcoin balance to Bob, 0.997 bitcoin balance to Alice, 1.003 bitcoin committed in Alice's HTLC. Alice's balance is reduced by the amount committed to the HTLC.

然后，Alice的节点构造一个HTLC，支付到哈希H，具有10个区块时间的退款超时（当前块+10），数量为1.003比特币（参见图12-10的步骤2）。额外的0.003将用于补偿参与此支付路由的中间节点。

Alice将此HTLC提供给Bob，从和Bob之间的通道余额中扣除1.003比特币，并将其提交给HTLC。该HTLC具有以下含义：“如果Bob知道秘密，Alice将其通道余额的1.003支付给Bob，或者如果超过10个区块时间后，则退还入Alice的余额”。

Alice和Bob之间的通道余额现在由承诺交易表示，其中有三个输出：Bob的2比特币余额，Alice的0.997比特币余额，Alice的HTLC中承诺的1.003比特币。承诺在HTLC中的金额从Alice的余额中被减去。

Bob now has a commitment that if he is able to get the secret R within the next 10 blocks, he can claim the 1.003 locked by Alice. With this commitment in hand, Bob's node constructs an HTLC on his payment channel with Carol. Bob's HTLC commits 1.002 bitcoin to hash H for 9 blocks, which Carol can redeem if she has secret R (see [Step-by-step payment routing through a Lightning Network](#) step 3). Bob knows that if Carol can claim his HTLC, she has to produce R. If Bob has R in nine blocks, he can use it to claim Alice's HTLC to him. He also makes 0.001 bitcoin for committing his channel balance for nine blocks. If Carol is unable to claim his HTLC and he is unable to claim Alice's HTLC, everything reverts back to the prior channel balances and no one is at a loss. The channel balance between Bob and Carol is now: 2 to Carol, 0.998 to Bob, 1.002 committed by Bob to the HTLC.

Bob现在有一个承诺，如果他能够在接下来的10个区块生产时间内获得秘密R，他可以获取Alice锁定的1.003。手上有了这一承诺，Bob的节点在和Carol的支付通道上构建了一个HTLC。Bob的HTLC提交1.002比特币到哈希H共9个区块时间，这个HTLC中如果Carol有秘密R她可以兑换（参见图12-10步骤3）。Bob知道，如果Carol要获取他的HTLC，她必须出示秘密R。如果Bob在9个区块的时间内有R，他可以用它来获取Alice的HTLC给自己。通过承诺自己的通道余额9个区块的时间，他也赚了0.001比特币。如果Carol无法获取他的HTLC，并且他也无法获取Alice的HTLC，那么一切都恢复到以前的通道余额，没有人会亏损。Bob和Carol之间的通道余额现在是：2比特币给Carol，0.998给Bob，1.002由Bob承诺给HTLC。

Carol now has a commitment that if she gets R within the next nine blocks, she can claim 1.002 bitcoin locked by Bob. Now she can make an HTLC commitment on her channel with Diana. She commits an HTLC of 1.001 bitcoin to hash H, for eight blocks, which Diana can redeem if she has secret R (see [Step-by-step payment routing through a Lightning Network](#) step 4). From Carol's perspective, if this works she is 0.001 bitcoin better off and if it doesn't she loses nothing. Her HTLC to Diana is only viable if R is revealed, at which

point she can claim the HTLC from Bob. The channel balance between Carol and Diana is now: 2 to Diana, 0.999 to Carol, 1.001 committed by Carol to the HTLC.

Carol现在有一个承诺，如果她在接下来的9个区块时间内获得R，她可以获取Bob的锁定1.002比特币。现在她可以在她与Diana的通道上构建HTLC承诺。她提交了一个1.001比特币的HTLC到哈希H，共计8个区块时间，如果Diana有秘密R，她就可以兑换（参见图12-10步骤4）。从Carol的角度来看，如果能够实现，她就可以获得的0.001比特币，否则也没有失去任何东西。她提交给Diana的HTLC，只有在R被泄露的情况下才可行，到那时候她可以从Bob那里索取HTLC。Carol和Diana之间的通道余额现在是：2给Diana，0.999给Carol，1.001由Carol承诺给HTLC。

Finally, Diana can offer an HTLC to Eric, committing 1 bitcoin for seven blocks to hash H (see [Step-by-step payment routing through a Lightning Network](#) step 5). The channel balance between Diana and Eric is now: 2 to Eric, 1 to Diana, 1 committed by Diana to the HTLC.

最后，Diana可以提供给Eric一个HTLC，承诺1比特币，7个区块时间，到哈希H（参见图12-10的步骤5）。Diana与Eric之间的通道余额现在是：2给Eric，1给Diana，1由Diana承诺给HTLC。

However, at this hop in the route, Eric has secret R. He can therefore claim the HTLC offered by Diana. He sends R to Diana and claims the 1 bitcoin, adding it to his channel balance (see [Step-by-step payment routing through a Lightning Network](#) step 6). The channel balance is now: 1 to Diana, 3 to Eric.

然而，在这条路上，Eric拥有秘密R，他可以获取Diana提供的HTLC。他将R发送给Diana，并获取1个比特币，添加到他的通道余额中（参见图12-10的步骤6）。通道平衡现在是：1给Diana，3给Eric。

Now, Diana has secret R. Therefore, she can now claim the HTLC from Carol. Diana transmits R to Carol and adds the 1.001 bitcoin to her channel balance (see [Step-by-step payment routing through a Lightning Network](#) step 7). Now the channel balance between Carol and Diana is: 0.999 to Carol, 3.001 to Diana. Diana has "earned" 0.001 for participating in this payment route.

现在，Diana有秘密R，因此，她现在可以获取来自Carol的HTLC。Diana将R发送给Carol，并将1.001比特币添加到其通道余额中（参见图12-10的步骤7）。现在Carol与Diana之间的通道余额是：0.999给Carol，3.001给Diana。Diana已经“赚了”参与这个付款路线0.001比特币。

Flowing back through the route, the secret R allows each participant to claim the outstanding HTLCs. Carol claims 1.002 from Bob, setting the balance on their channel to: 0.998 to Bob, 3.002 to Carol (see [Step-by-step payment routing through a Lightning Network](#) step 8). Finally, Bob claims the HTLC from Alice (see [Step-by-step payment routing through a Lightning Network](#) step 9). Their channel balance is updated as: 0.997 to Alice, 3.003 to Bob.

通过路由回传，秘密R允许每个参与者获取未完成的HTLC。Carol从Bob那里获取1.002个比特币，将他们通道余额设为：0.998给Bob，3.002给Carol（参见闪电网络步骤8）。最后，Bob获取来自Alice的HTLC（参见闪电网络步骤9）。他们的通道余额更新为：0.997给Alice，3.003给Bob。

Alice has paid Eric 1 bitcoin without opening a channel to Eric. None of the intermediate parties in the payment route had to trust each other. For the short-term commitment of their funds in the channel they are able to earn a small fee, with the only risk being a small delay in refund if the channel was closed or the routed payment failed.

在没有向Eric打开通道的情况下，Alice已经支付了Eric 1比特币。付款路线中的中间方不必要互相信任。在他们的通道内做一个短时间的资金承诺，他们可以赚取一小笔费用，唯一的风险是，如果通道关闭或路由付款失败，退款有段短短的延迟时间。

## 12.7.2 闪电网络传输和路由

All communications between LN nodes are encrypted point-to-point. In addition, nodes have a long-term public key that they [use as an identifier and to authenticate each other](#). LN节点之间的所有通信都是点对点加密的。另外，节点有一个长期公钥，它们用作标识符并且彼此认证对方。



Whenever a node wishes to send a payment to another node, it must first construct a *path* through the network by connecting payment channels with sufficient capacity. Nodes advertise routing information, including what channels they have open, how much capacity each channel has, and what fees they charge to route payments. The routing information can be shared in a variety of ways and different routing protocols are likely to emerge as Lightning Network technology advances. Some Lightning Network implementations use the IRC protocol as a convenient mechanism for nodes to announce routing information. Another implementation of route discovery uses a P2P model where nodes propagate channel announcements to their peers, in a "flooding" model, similar to how bitcoin propagates transactions. Future plans include a proposal called [Flare](#), which is a hybrid routing model with local node "neighborhoods" and longer-range beacon nodes. 每当节点希望向另一个节点发送支付时，它必须首先通过连接具有足够容量的支付通道来构建通过网络的路径。节点宣传路由信息，包括他们已经打开了什么通道，每个通道拥有多少容量，以及他们收取多少路由支付费用。路由信息可以以各种方式共享，并且随着闪电网络技术的进步，不同的路由协议可能会出现。一些闪电网络实施使用IRC协议作为节点宣布路由信息的一种方便的机制。路由发现的另一种实现方式是使用P2P模型，其中节点将通道宣传传播给他们的对等体，在“洪水泛滥”模型中，这类似于比特币传播交易的方法。未来的计划包括一个名为Flare的建议，它是一种具有本地节点“邻居”和较长距离的信标节点的混合路由模型。

In our previous example, Alice's node uses one of these route discovery mechanisms to find one or more paths connecting her node to Eric's node. Once Alice's node has constructed a path, she will initialize that path through the network, by propagating a series of encrypted and nested instructions to connect each of the adjacent payment channels.

在我们前面的例子中，Alice的节点使用这些路由发现机制之一来查找将她的节点连接到Eric的节点的一个或多个路径。一旦Alice的节点构建了路径，她将通过网络初始化该路径，传播一系列加密和嵌套的指令来连接每个相邻的支付通道。

Importantly, this path is only known to Alice's node. All other participants in the payment route see only the adjacent nodes. From Carol's perspective, this looks like a payment from Bob to Diana. Carol does not know that Bob is actually relaying a payment from Alice. She also doesn't know that Diana will be relaying a payment to Eric.

重要的是，这个路径只有Alice的节点才知道。付款路线上的所有其他参与者只能看到相邻的节点。从Carol的角度来看，这看起来像是从Bob到Diana的付款。Carol不知道Bob实际上是中继转发Alice的汇款。她也不知道Diana将会向Eric中继转发付款。

This is a critical feature of the Lightning Network, because it ensures privacy of payments and makes it very difficult to apply surveillance, censorship, or blacklists. But how does Alice establish this payment path, without revealing anything to the intermediary nodes? 这是闪电网络的一个重要特征，因为它确保了付款的隐私，并且使得很难应用监视，审查或黑名单。但是，Alice如何建立这种付款途径，而不向中间节点透露任何内容？

The Lightning Network implements an onion-routed protocol based on a scheme called [Sphinx](#). This routing protocol ensures that a payment sender can construct and communicate a path through the Lightning Network such that:

闪电网络实现了一种基于称为Sphinx的方案的洋葱路由协议。

该路由协议确保支付发送者可以通过闪电网络构建和通信路径，使得：

- Intermediate nodes can verify and decrypt their portion of route information and find the next hop.
- Other than the previous and next hops, they cannot learn about any other nodes that are part of the path.
- They cannot identify the length of the payment path, or their own position in that path.
- Each part of the path is encrypted in such a way that a network-level attacker cannot associate the packets from different parts of the path to each other.

- Unlike Tor (an onion-routed anonymization protocol on the internet), there are no "exit nodes" that can be placed under surveillance. The payments do not need to be transmitted to the bitcoin blockchain; the nodes just update channel balances.

- 中间节点可以验证和解密其部分路由信息，并找到下一跳。
- 除了上一跳和下一跳，他们不能了解作为路径一部分的任何其他节点。
- 他们无法识别支付路径的长度，或者他们自己在该路径中的位置。
- 路径的每个部分被加密，使得网络级攻击者不能将来自路径的不同部分的数据包彼此关联。
- 不同于Tor（互联网上的洋葱路由匿名协议），没有可以被监视的“退出节点”。付款不需要传输到比特币区块链，节点只是更新通道余额。

Using this onion-routed protocol, Alice wraps each element of the path in a layer of encryption, starting with the end and working backward. She encrypts a message to Eric with Eric's public key. This message is wrapped in a message encrypted to Diana, identifying Eric as the next recipient. The message to Diana is wrapped in a message encrypted to Carol's public key and identifying Diana as the next recipient. The message to Carol is encrypted to Bob's key. Thus, Alice has constructed this encrypted multilayer "onion" of messages. She sends this to Bob, who can only decrypt and unwrap the outer layer. Inside, Bob finds a message addressed to Carol that he can forward to Carol but cannot decipher himself. Following the path, the messages get forwarded, decrypted, forwarded, etc., all the way to Eric. Each participant knows only the previous and next node in each hop.

使用这种洋葱路由协议，Alice将路径的每个元素包裹在一层加密中，从尾端开始倒过来运算。她用Eric的公钥加密了Eric的消息。该消息包裹在加密到Diana的消息中，将Eric标识为下一个收件人。给Diana的消息包裹在加密到Carol的公钥的消息中，并将Diana识别为下一个收件人。对Carol的消息被Bob的密钥加密。这样一来，Alice已经构建了这个加密的多层“洋葱”的消息。她发送给Bob，他只能解密和解开外层。在里面，Bob发现一封给Carol的消息，他可以转发给Carol，但不能自己破译。按照路径，消息被转发，解密，转发等，一路到Eric那里。每个参与者只知道各自这一跳的前一个和下一个节点。

Each element of the path contains information on the HTLC that must be extended to the next hop, the amount that is being sent, the fee to include, and the CLTV locktime (in blocks) expiration of the HTLC. As the route information propagates, the nodes make HTLC commitments forward to the next hop.

路径的每个元素包含承载于HTLC的必须扩展到下一跳的信息，HTLC中的要发送的数量，要包括的费用以及CLTV锁定到期时间（以块为单位）。随着路由信息的传播，节点将HTLC承诺转发到下一跳。

At this point, you might be wondering how it is possible that the nodes do not know the length of the path and their position in that path. After all, they receive a message and forward it to the next hop. Doesn't it get shorter, allowing them to deduce the path size and their position? To prevent this, the path is always fixed at 20 hops and padded with random data. Each node sees the next hop and a fixed-length encrypted message to forward. Only the final recipient sees that there is no next hop. To everyone else it seems as if there are always 20 more hops to go.

在这一点上，您可能会想知道节点怎么知道路径的长度及其在该路径中的位置。毕竟，他们收到一个消息，并将其转发到下一跳。难道它不会将路径缩短，或者允许他们推断出路径大小和位置？为了防止这种情况，路径总是固定在20跳，并用随机数据填充。每个节点都会看到下一跳和一个要转发的固定长度的加密消息。只有最终的收件人看得到没有下一跳。对于其他人来说，似乎总是有20多跳要走。

### 12.7.3闪电网络优势

A Lightning Network is a second-layer routing technology. It can be applied to any blockchain that supports some basic capabilities, such as multisignature transactions, timelocks, and basic smart contracts.

A Lightning Network is a second-layer routing technology. It can be applied to any blockchain that supports some basic capabilities, such as multisignature transactions, timelocks, and basic smart contracts.

闪电网络是一种第二层路由技术。

它可以应用于支持一些基本功能的任何区块链，这些基本功能如多重签名交易、时间锁、基本智能合约。

If a Lightning Network is layered on top of the bitcoin network, the bitcoin network can gain a significant increase in capacity, privacy, granularity, and speed, without sacrificing the principles of trustless operation without intermediaries:

如果闪电网络建在在比特币网络之上，则比特币网络可以大大提高容量、隐私性、粒度和速度，而不会牺牲无中介机构的无信任操作的原则：

**Privacy:** Lightning Network payments are much more private than payments on the bitcoin blockchain, as they are not public. While participants in a route can see payments propagated across their channels, they do not know the sender or recipient.

**隐私：**闪电网络付款比比特币区块链的付款更私密，因为它们不是公开的。

虽然路由中的参与者可以看到在其通道上传播的付款，但他们并不知道发送者或接收者。

**Fungibility:** A Lightning Network makes it much more difficult to apply surveillance and blacklists on bitcoin, increasing the fungibility of the currency.

**可替换性：**闪电网络使得在比特币上应用监视和黑名单变得更加困难，从而增加了货币的可替换性。

**Speed:** Bitcoin transactions using Lightning Network are settled in milliseconds, rather than minutes, as HTLCs are cleared without committing transactions to a block.

**速度：**使用闪电网络的比特币交易将以毫秒为单位结算，而不是分钟，因为HTLC在不用提交交易到区块上的情况下被结算。

**Granularity:** A Lightning Network can enable payments at least as small as the bitcoin "dust" limit, perhaps even smaller. Some proposals allow for subsatoshi increments.

**粒度：**闪电网络可以使支付至少与比特币“灰尘”限制一样小，甚至更小。

一些建议允许子聪级增量 (subsatoshi increments)。

**Capacity:** A Lightning Network increases the capacity of the bitcoin system by several orders of magnitude. There is no practical upper bound to the number of payments per second that can be routed over a Lightning Network, as it depends only on the capacity and speed of each node.

**容量：**闪电网络将比特币系统的容量提高了几个数量级。

每秒可以通过闪电网络路由的支付数量没有实际上限，因为它仅取决于每个节点的容量和速度。

**Trustless Operation:** A Lightning Network uses bitcoin transactions between nodes that operate as peers without trusting each other. Thus, a Lightning Network preserves the principles of the bitcoin system, while expanding its operating parameters significantly.

**无信任操作：**闪电网络在不需要互相信任的节点之间使用比特币交易。因此，闪电网络保留了比特币系统的原则，同时显著扩大了它的操作参数。

Of course, as mentioned previously, the Lightning Network protocol is not the only way to implement routed payment channels. Other proposed systems include Tumblebit and Teechan. At this time, however, the Lightning Network has already been deployed on testnet. Several different teams have developed competing implementations of LN and are working toward a common interoperability standard (called BOLT). It is likely that Lightning Network will be the first routed payment channel network to be deployed in production.

当然，如前所述，闪电网络协议不是实现路由支付通道的唯一方法。

其他被提出的系统包括Tumblebit和Teechan。

然而，现在，闪电网络已经部署在testnet上了。几个不同的团队已经开发了正在竞争的LN实现，并且正在努力实现一个通用的互操作性标准（称为BOLT）。闪电网络很可能是第一个部署在生产实际中的路由支付通道网络。



## 12.8结论

We have examined just a few of the emerging applications that can be built using the bitcoin blockchain as a trust platform. These applications expand the scope of bitcoin beyond payments and beyond financial instruments, to encompass many other applications where trust is critical. By decentralizing the basis of trust, the bitcoin blockchain is a platform that will spawn many revolutionary applications in a wide variety of industries.

我们只研究了几个新兴应用，可以使用比特币区块链作为一个信任平台来构建这些应用。

这些应用把比特币的范围扩展到支付和金融工具之外的地方，涵盖了许多其它应用，在这些应用中信任至关重要。

通过将信任的基础去中性化，比特币区块链成为一个平台，它将在各个行业中引起许多革命性的应用。

