# Bitcoin Payment Channels Analysis

**July 30, 2019**

Joël Gugger[1]

TrueLevel
joel@truelevel.io

**Abstract.** Payment channels are one part of the scalability solution and user experience enhancement. Various schemes have been proposed in the past years with specific use cases. We propose new definitions to help classifying payment channels and analyze various Bitcoin channel schemes. We summarize the analyzed protocols and highlight their strong points.

**Keywords:** Bitcoin, payment channels, layer-2 schemes

## 1 Introduction

Payment channels or micropayment channels are one part of the scalability solution. The idea of payment channels was suggested by Satoshi in an email to Mike Hearn. Since then, various schemes to construct such structures have been proposed. To have a better understanding of the differences between various channel schemes, and to be able to analyze a channel scheme objectively, we propose a list of characteristic definitions for payment channel construction. An analysis of different commonly exposed payment channel constructions is done following these definitions. The list does not contain all the payment channel schemes, some of them might be missing. However, the list contains a fairly good representation of the different existing constructions.

## 2 Payment channel definitions

These definitions specify the necessary and sufficient conditions for payment channels to be qualified as a member of a specific set. They set boundaries or limits that separate the term from any other term. The following definitions qualify properties that micropayment channels can endorse with the view of a particular participant.

Sets of transactions represent transitions that modify the channel state. Transactions can be broadcast into the network to effectively affect the on-chain channel state or been kept by all participants off-chain waiting new state transitions. Participants may own funds in the given blockchain and work collectivly to create and update the channel state.

Channel funds must be owned by one and only one participant at a time. Owning funds in a channel for a given participant is defined as holding a set of transactions that might not yet be broadcast but allowing the participant to claim the funds.

**Definition 1 (Participant).** *A participant $p \in \mathcal{P}$ may own funds in the channel and work collectivly to create and update the channel state.*

**Definition 2 (Step).** *A step $s \in \mathcal{S}$ is an atomic opperation performed by one particular participant in the protocol.*

**Definition 3 (Transaction).** *Transactions $t \in \mathcal{T}$ are blockchain layer's enforcement mechanism. A transaction can be generated by one or more steps depending on interactivity level required by the transaction itself, i.e. transaction with multiple signatures require multiple participants and thus multiple steps.*

**Definition 4 (Operation).** *An operation $\mathcal{O}$ is a set of steps, i.e. $\mathcal{O} \subset \mathcal{S}$, that can be started by one participan. We define a payment operation that allow one participant $p \in \mathcal{P}$ to send money through the channel to another participant $p' \in \mathcal{P}$ for $p \neq p'$.*

The channel state is modified by participants performing operations. After each operations, participants might get a new transaction set $\mathcal{T}'$ that replace the previous one. Some protocol may define more complex operations such as conditional payments.

**Definition 5 (Exercise).** *Exercising is the process of applying the current channel state on-chain by broadcasting a transaction set $\mathcal{T}$ with one or more transactions.*

We define two exercises that constitue the participants' capabilities: (i) settlement operation, and (ii) refund operation. Participants can endorse one or more capabilities in the channel depending if they send or receive payments.

**Definition 6 (Channel).** *A channel is composed by a set of participants $\mathcal{P}$ that collaborate to execute the steps $\mathcal{S}$ of a protocol by performing operations $\mathcal{O}$. Each participans can exercise these operations' output.*

## 2.1 Types of payment channels

We can distinguish two type of channels: (i) unidirectional channels that allow one user to send money to another user in one and only one direction and (ii) bidirectional channels that allow participans to send in either direction. Bidirectional channels must allow to send back and forth multiple time the same amount and then are more optimized than two independent unidirectional channels.

Unidirectional channels have asymetric participant's properties, but not bidirectional channels. Thus it is important to analyze well in which cases unidirectional and bidirectional channels should be used and list which properties hold for which participant.

We denote in channel $p_1 \rightarrow p_2$ such that $p_1, p_2 \in \mathcal{P}$ if $p_1$ can send money through the channel to $p_2$.

**Unidirectional** In a two-participant unidirectional channel, there is a payer, later referred to as participant-one or client, and a payee, later referred to as participant-two or provider. We describe unidirectional channles as $p_1 \rightarrow p_2$ and $p_2 \not\rightarrow p_1$, i.e. it is not possible to transfer money back in the reverse direction through the channel. Payers can exercise refunds and payees settlements.

These channels are property asymmetric, i.e. each participant benefits different properties. Thus analysis must be done in the view of each channel participants $p_i \in \mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$.

**Bidirectional** We describe two-participant bidirectional channels as $p_1 \rightarrow p_2$ and $p_2 \rightarrow p_1$, i.e. participants can send in either direction. Each participant are payee and payer, thus exercising settlement or refund is the same.

Bidirectional channel can have specific structures or be pairings of existing unidirectional channels. These channels have properties symmetry, each participant $p_i \in \mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ benefits from the same channel properties.

## 2.2 Payment channel properties

We define properties and corollaries to help classifying unidirectional and bidirectional payment channels.

**Definition 7 (Trustless).** *A channel is trustless for a participant $p_i \in \mathcal{P}$ if and only if his financial safety at each step $\mathcal{S}$ of the protocol does not depend on the behavior of participants $\mathcal{P}' = \{\, p' \,|\, p' \neq p_i \,\}$.*

All payment channel are trustless, otherwise the scheme has no value but some have on-chain requirements that might not be active on certain chains such as Segregated Witness (SegWit).

**Definition 8 (Optimal).** *A channel is optimal for a participant $p_i \in \mathcal{P}$ if and only if the transaction set $\mathcal{T}'$ to exercise has a cardinality of one, i.e. $|\mathcal{T}'| = 1$.*

**Definition 9 (Open-ended).** *A channel is open-ended for a participant $p_i \in \mathcal{P}$ if and only if there is no predetermined channel lifetime at the setup.*

A channel that is not open-ended can have a refresh mechanism on-chain with a designated transaction to extend the lifetime.

**Definition 10 (Undelayed).** *A channel is undelayed for a participant $p_i \in \mathcal{P}$ if and only if this participant can exercise a transaction set at any time.*

If one transaction is delayed in a set then the set is considered as delayed. A transaction can have a fixed delay in the future, e.g. can be exercised only after the 1st of May 2020, or can have a relative delay, e.g. can be exercised only 3 hours after this other transaction have been exercised.

**Definition 11 (Non-interactive).** *A channel is non-interactive for a participant $p_i \in \mathcal{P}$ if and only if this participant does not have the burden to watch other participants exercises and reacts in order to guarantee his safety.*

With these definitions it is possible to infer some significant corollaries. If a channel is undelayed for a participant this participant can exercise his latest state at any time, and if this channel is also optimal for the same participant only one transaction is needed to transfer the funds out. If only one transaction is needed to transfer the funds and this transaction can be exercised at any time, then the channel is instantaneous for the participant.

**Corollary 1 (Instantaneous).** *A channel is instantaneous for a participant $p_i \in \mathcal{P}$ if the channel is undelayed and optimal for this participant.*

We also define offline safety with non-interactive open-ended channels. If the channel is open-ended its lifetime is bounded to the blockchain lifetime without requiring interactivity and if the channel is non-interactive the participant does not have to watch and react, then the channel have offline safety.

**Corollary 2 (Offline safety).** *A channel is offline safe for a participant $p_i \in \mathcal{P}$ if the channel is non-interactive and open-ended for this participant.*

We emphasise that offline-safety in channels is an important feature for real world application when it comes to business to clients models. Burdening interactivity on clients implies technical restrictions or extra economics to handle third party policing outsourcing.

In Bitcoin private keys management is an important topic. One can loose money if private keys are lost and backups are not available. Payment channels also inherit that management in addition to managing the channel states. Again one can loose money in a channel if channel states are lost. However it is possible to create trustless state crash recovery mechanism for some participants depending on channel properties if private keys are backup.

**Corollary 3 (Crash recovery).** *An unidirectional channel $p_1 \rightarrow p_2$ can implement a crash recovery mechanism for a participant $p_1$ if the on-chain state can be refreshed.*

In unidirectional channels $p_1 \rightarrow p_2$ participant $p_2$ has an incentive to only submit the latest state because his ownings can only increase. If $p_1$ wants to recover the state of a channel with $p_2$ he can ask for the latest state (transaction set $\mathcal{T}$ and metadata) and verify the signatures. If they match they can refresh on-chain with the same information, thus invalidating states that might have been created later.

## 3 Analysis of payment channels

We analyze various types of unidirectional and bidirectional Bicoin payment channels from the old and simplest one to the more recent ones.

### 3.1 Spilman-style payment channels

Spilman-style payment channels, proposed by Jeremy Spilman in 2013 [1], are the most simple construction of a unidirectional payment channel. They have a finite lifetime predefined at the setup phase and the payer cannot exercise his refund before the end of the channel lifetime. Thus he can receive his funds back if the payee settles the channel before the end of the lifetime.

The channel is one-time use. When the payer or the payee get their funds, the channel is closed. Neither the payer nor the payee need to watch the blockchain to react to events during the lifetime of the channel because only the payee can exercise the settlement, so both do not need to watch the blockchain to be safe.

| Participant | Trustless | Optimal | Open-ended | Undelayed | Non-interactive |
|---|---|---|---|---|---|
| Payer | Yes | Yes | No | No | Yes |
| Payee | Yes | Yes | No | Yes | Yes |

Table 1: Summary of Spilman-style payment channel properties

This scheme is the most optimal in number of transactions needed both sides but with a lack of flexibility. According to the previous definitions, Spilman-style ephemeral payment channels are optimal non-interactive for the payer, and instantaneous non-interactive channels for the payee.

It is worth noting that, without a proper fix to transaction malleability like SegWit this scheme is not secure [2, 3, 4, 5].

### 3.2 `CLTV`-style payment channels

Introduced in 2015, `CLTV`-style payment channels offer a solution to the malleability problem in Spilman-style payment channels. With the new `OP_CODE` check locktime verify (`OP_CHECKLOCKTIMEVERIFY`), redefining the `OP_NOP2`, it is possible to enforce the non-spending of a transaction output until some time in the future. With `OP_CHECKLOCKTIMEVERIFY` a transaction output can enforce the spending transaction to have a `nLockTime` later or equal to the specified value in the script [6].

Instead of creating a funding transaction and a refund transaction vulnerable to transaction malleability attacks, the client creates the funding transaction output with a script (Listing **??**) that allows the provider and the client to spend the funds with co-operation or after a lock time the client can spend the funds without the co-operation of the provider.

`CLTV`-style payment channels have the same properties as Spilman-style payment channels but require `OP_CHECKLOCKTIMEVERIFY`.

### 3.3 Decker-Wattenhofer duplex payment channels

Decker-Wattenhofer duplex payment channels [7], also called Duplex Micropayment Channels (DMC), proposed in 2015, are bidirectional channels based on pairs of Spilman-style unidirectional channels. The construction has a finite lifetime predefined at the setup phase but can be refreshed on-chain to keep the channel open with an updated state. During the refresh process, it is possible to refill the channel, and the scheme allows payment routing with Hashed Timelock Contracts (HTLC).

DMC payment channels are not optimal, uncooperative closing of the channel requires $d + 2$ transactions (where $d$ is equal to the revocation tree depth). They are not undelayed, without other participants cooperation the funds are recovered after the `nLockTime` values. DMC are not by default open-ended, but a dedicated transaction needs to be broadcast before the end of `nLockTime`.

| Trustless | Optimal | Open-ended | Undelayed | Non-interactive |
|---|---|---|---|---|
| Yes | No | (Yes) | No | No |

Table 2: Summary of Decker-Wattenhofer duplex payment channel properties

### 3.4 Poon-Dryja payment channels

Poon-Dryja payment channels, also called Lightning Network, are an HTLC based bidirectional payment channels which allow payments to be securely routed across multiple peer-to-peer payment channels [8] and conditional payments.

The scheme is trustless (assuming that SegWit has been implemented), open-ended, and undelayed but not optimal when the channel closes without co-operation nor non-interactive.

Poon-Dryja scheme is the currently the most advanced payment channel network implemented runnig on a mainnet in Bitcoin.

| Trustless | Optimal | Open-ended | Undelayed | Non-interactive |
|---|---|---|---|---|
| Yes | No | Yes | Yes | No |

Table 3: Summary of Poon-Dryja payment channel properties

### 3.5 Shababi-Gugger-Lebrecht payment channel

Shababi-Gugger-Lebrecht unidirectional payment channel is a modified version of other layer-two applications, such as "Yours Lightning Protocol" or Lightning Network [8, 9].

The scheme is specially designed for client to provider scenario, where one provider has multiple clients through multiple channels. The core design aims to be as cheap as possible for the provider with instantaneous settlement and to garantee offline safety for the client, thus enabling easier wallet integration.

We designed an unidirectional channel with these specific properties because we think that a mix of unidirectional and bidirectional channels, with HTLC to route payments between them, is the best solution to provide the best user experience and guarantee client safety and recovery.

| Participant | Trustless | Optimal | Open-ended | Undelayed | Non-interactive |
|---|---|---|---|---|---|
| Payer | Yes | No | Yes | No | Yes |
| Payee | Yes | Yes | Yes | Yes | No |

Table 4: Summary of Shababi-Gugger-Lebrecht payment channel properties

## 4 Summary

We summarize the different properties of the proposed definitions of common Bitcoin channel schemes. We denote Decker-Wattenhofer channels as opend-end in parentheses because an on-chain refresh mechanism exists.

| Channel | Type | Optimal | Open-ended | Undelayed | Non-inter. |
|---|---|---|---|---|---|
| Spilman-style | Uni | Yes/Yes | No/No | No/Yes | Yes/Yes |
| CLTV-style | Uni | Yes/Yes | No/No | No/Yes | Yes/Yes |
| Decker-Wattenhofer DMC | Bi | No | (Yes) | No | No |
| Poon-Dryja | Bi | No | Yes | Yes | No |
| Shababi-Gugger-Lebrecht | Uni | No/Yes | Yes/Yes | No/Yes | Yes/No |

Table 5: Summary of analyzed payment channels

## References

[1] Jeremy Spilman. *[Bitcoin-development] Anti DoS for tx replacement.* 2013. URL: `https://lists.linuxfoundation.org/pipermail/bitcoin-dev/2013-April/002433.html` (visited on 02/04/2018).

[2] Eric Lombrozo, Johnson Lau, and Pieter Wuille. *Segregated Witness.* URL: `https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki` (visited on 02/02/2018).

[3] Pieter Wuille. *Dealing with malleability.* 2014. URL: `https://github.com/bitcoin/bips/blob/master/bip-0062.mediawiki` (visited on 02/04/2018).

[4] Marcin Andrychowicz et al. "How to deal with malleability of BitCoin transactions". In: *CoRR* abs/1312.3230 (2013). arXiv: `1312.3230`. URL: `http://arxiv.org/abs/1312.3230`.

[5] Christian Decker and Roger Wattenhofer. "Bitcoin Transaction Malleability and MtGox". In: *CoRR* abs/1403.6676 (2014). arXiv: `1403.6676`. URL: `http://arxiv.org/abs/1403.6676`.

[6] Peter Todd. *CHECKLOCKTIMEVERIFY.* 2014. URL: `https://github.com/bitcoin/bips/blob/master/bip-0065.mediawiki` (visited on 02/05/2018).

[7] Christian Decker and Roger Wattenhofer. "A Fast and Scalable Payment Network with Bitcoin Duplex Micropayment Channels". In: *17th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS), Edmonton, Canada.* Aug. 2015.

[8] Joseph Poon and Thaddeus Dryja. "The bitcoin lightning network: Scalable off-chain instant payments". In: *draft version 0.5* 9 (2016).

[9] Ryan X. Charles and Clemens Ley. *Yours Lightning Protocol.* 2016. URL: `https://github.com/yoursnetwork/yours-channels/blob/master/docs/yours-lightning.md`.