Understanding the Lightning Network, Part 1: Building a Bidirectional Bitcoin Payment Channel

by **Aaron van Wirdum** May 31, 2016 12:06 PM EST

The Lightning Network is probably the most highly anticipated technological innovation to be deployed on top of Bitcoin. The payment layer, first proposed by Joseph Poon and Tadge Dryja about a year ago, promises to support a virtually unlimited number of off-chain transactions among users, at nearly no cost – while leveraging the security offered by Bitcoin.

At least three companies – Poon and Dryja's Lightning, Blockstream and Blockchain – are currently working on implementations of the technology. But few outside this small technological frontline fully grasp how the "future of micropayments" is set to boost Bitcoin's capabilities.

In this three-part series, *Bitcoin Magazine* lays out the basic building blocks of the Lightning Network, and shows how they fit together to realize this upcoming protocol layer.

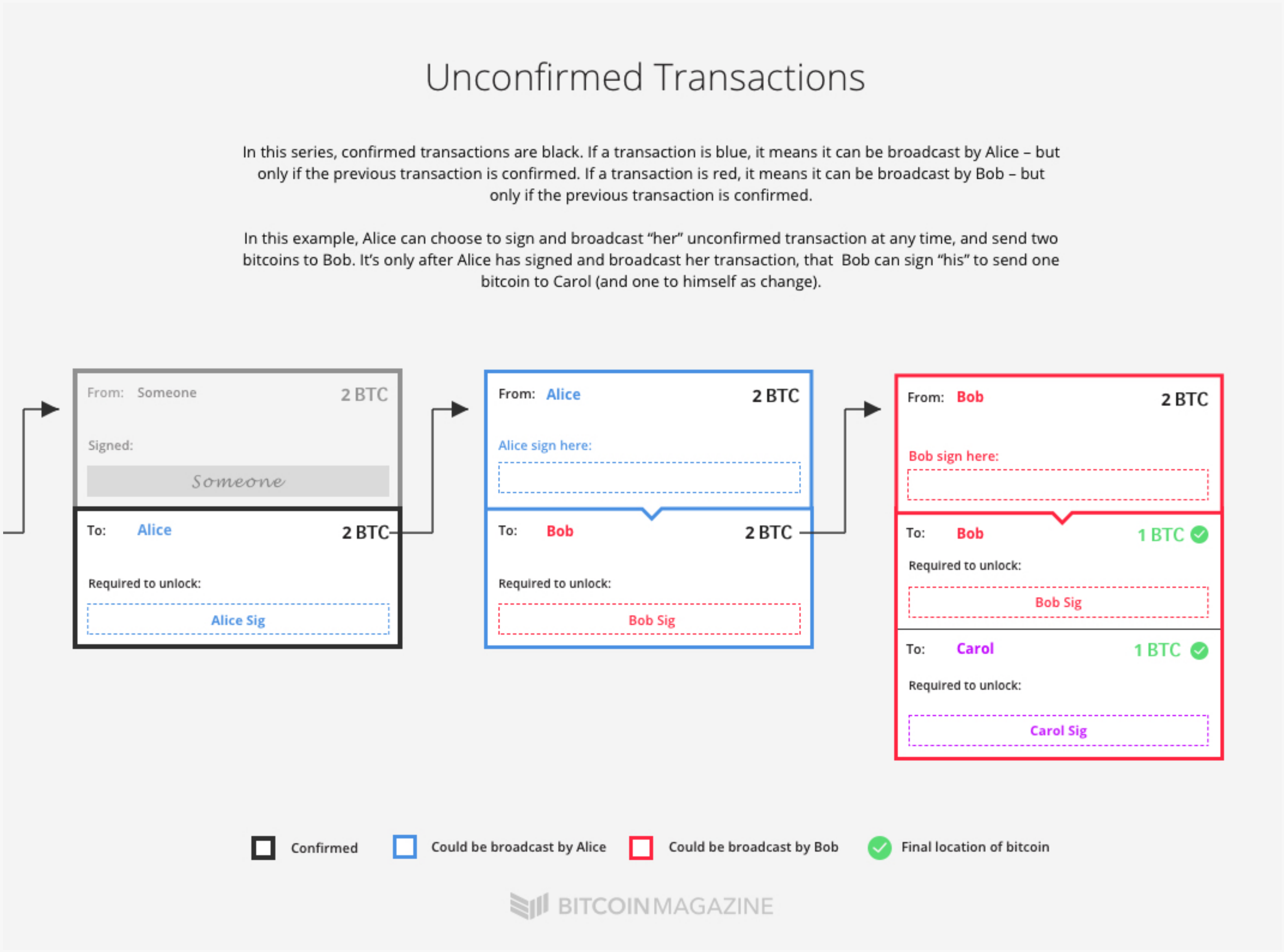
This first part of the series establishes the necessary building blocks, and shows channel.

(Note: Anyone with a solid understanding of Bitcoin can skip the building blocks.)

Building Block #1: Unconfirmed Transactions

At its heart, the Bitcoin protocol consists of transactions, that are typically linked to previous transactions, and potentially to future transactions. Each transaction contains *inputs*, which refer to addresses bitcoins are sent *from*, and *outputs*, which refer to addresses bitcoins are sent *to*. Additionally, inputs must include the requirements to send the bitcoins, like signatures that prove “ownership” of the input-addresses. Outputs, meanwhile, establish the new requirements, that must be included in the input of a *subsequent* transaction.

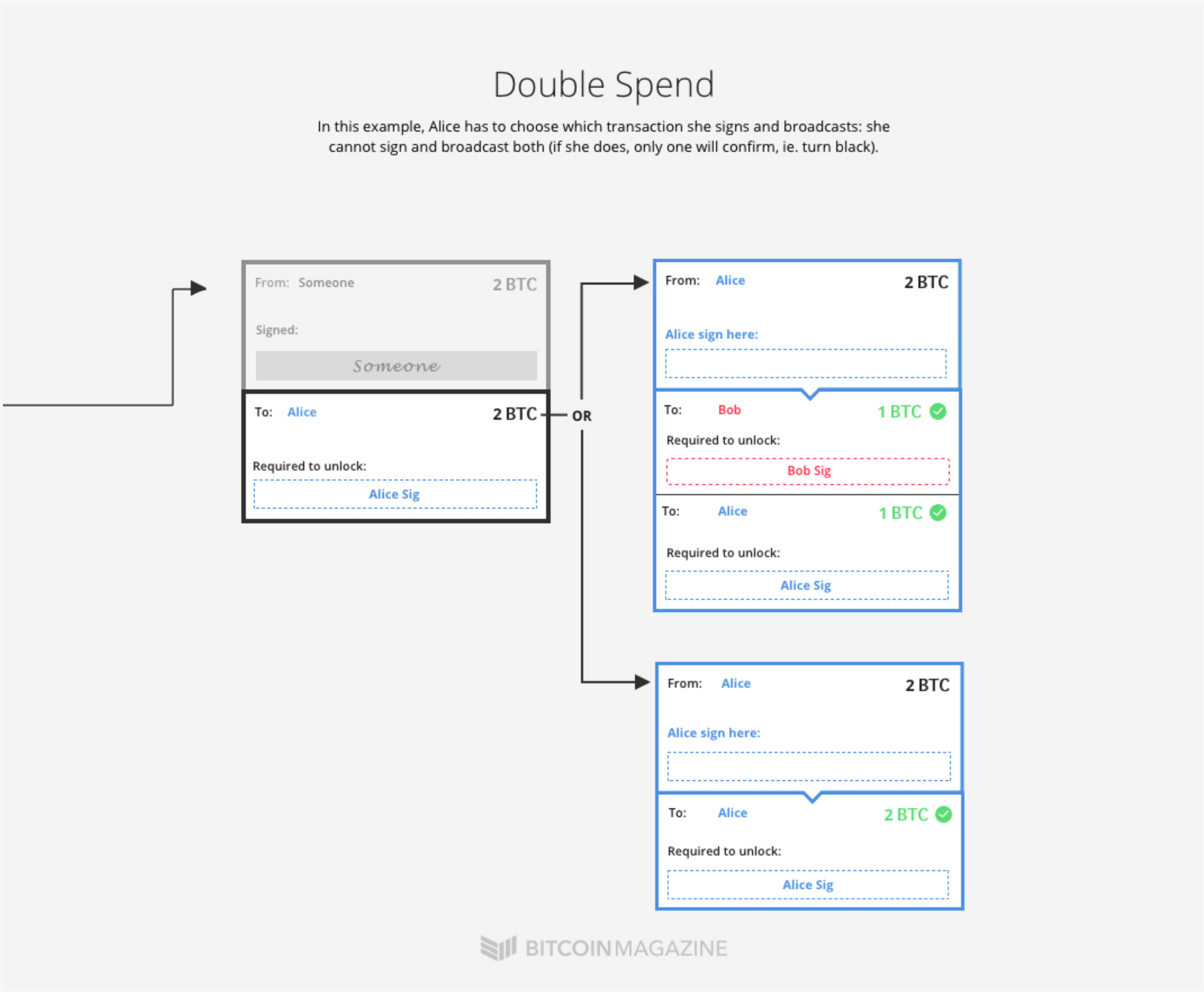
As one of its key features, the Lightning Network is built up from more or less regular Bitcoin transactions. It's just that these transactions are typically not actually broadcast over the Bitcoin network. Instead, they are stored locally, on the nodes of users - but they can be broadcast over the network at any time.


Unconfirmed Transactions
In this series, confirmed transactions are black. If a transaction is blue, it means it can be broadcast by Alice – but only if the previous transaction is confirmed. If a transaction is red, it means it can be broadcast by Bob – but only if the previous transaction is confirmed.
In this example, Alice can choose to sign and broadcast "her" unconfirmed transaction at any time, and send two bitcoins to Bob. It's only after Alice has signed and broadcast her transaction, that Bob can sign "his" to send one bitcoin to Carol (and one to himself as change).
From: Someone 2 BTC
Signed:
Someone
To: Alice 2 BTC
Required to unlock:
Alice Sig
From: Alice 2 BTC
Alice sign here:
To: Bob 2 BTC
Required to unlock:
Bob Sig
From: Bob 2 BTC
Bob sign here:
To: Bob 1 BTC
Required to unlock:
Bob Sig
To: Carol 1 BTC
Required to unlock:
Carol Sig
Confirmed
Could be broadcast by Alice
Could be broadcast by Bob
Final location of bitcoin
BITCOIN MAGAZINE

Building Block #2: Double-Spend Protection

The second building block for the Lightning Network probably doesn't require much explaining, as it's arguably the *raison d'être* for Bitcoin itself: double-spend protection. If two transactions (or: inputs) rely on the same output, only one can confirm.

The important thing to keep in mind here is that even unconfirmed transactions can be conflicting, meaning only one can ever confirm.

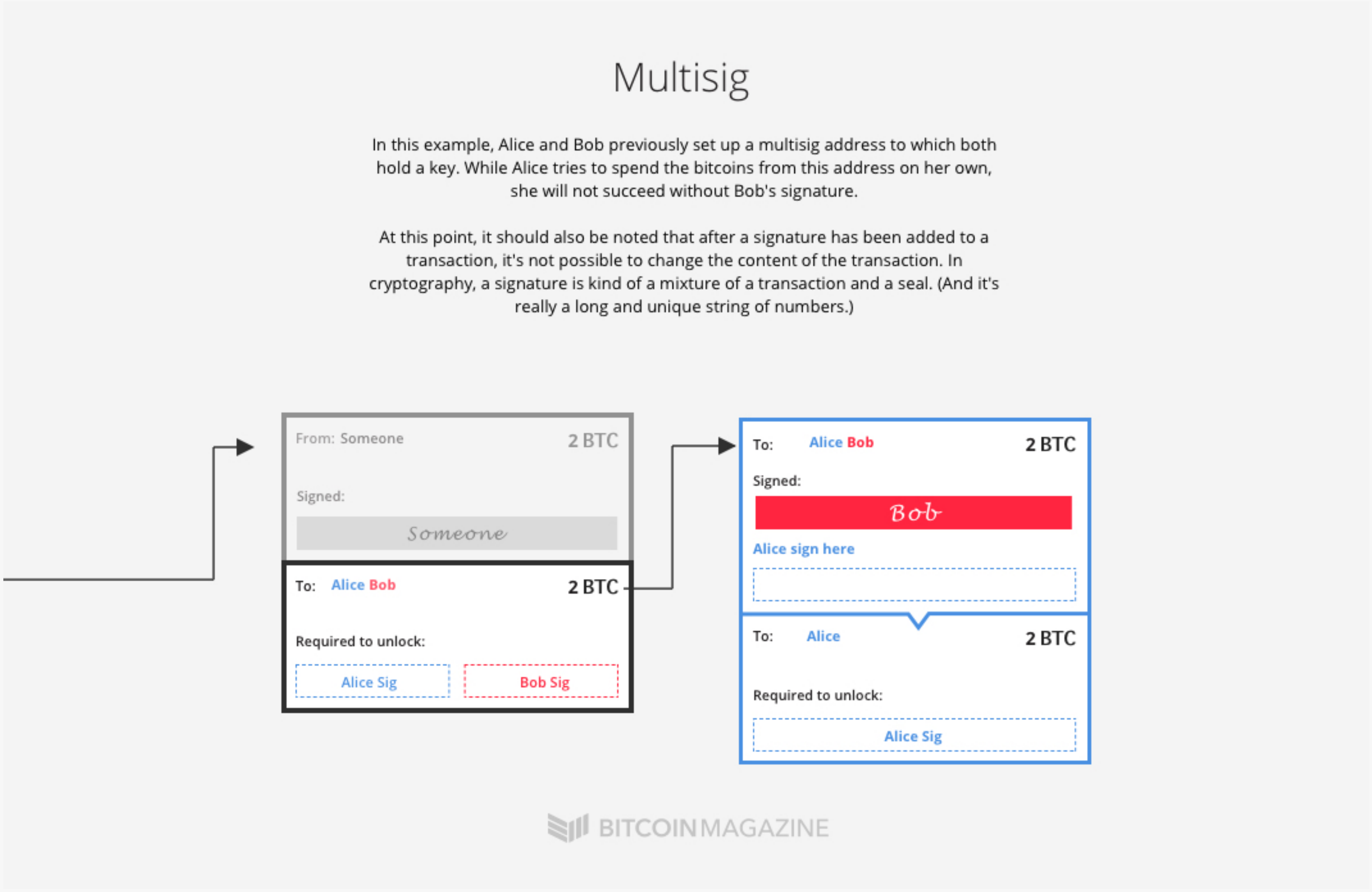


Building Block #3: Multisig

The third building block of the Lightning Network is also a straightforward one: multisignature (multisig) addresses. (Or more generally: P2SH-addresses.)

Multisig addresses are Bitcoin addresses that – as the name suggests - require multiple private keys to “unlock” and spend bitcoins from. Multisig addresses can be set up under all sorts of conditions. For instance to require two out of three possible keys, or fifteen out of fifteen, or just about any other combination.

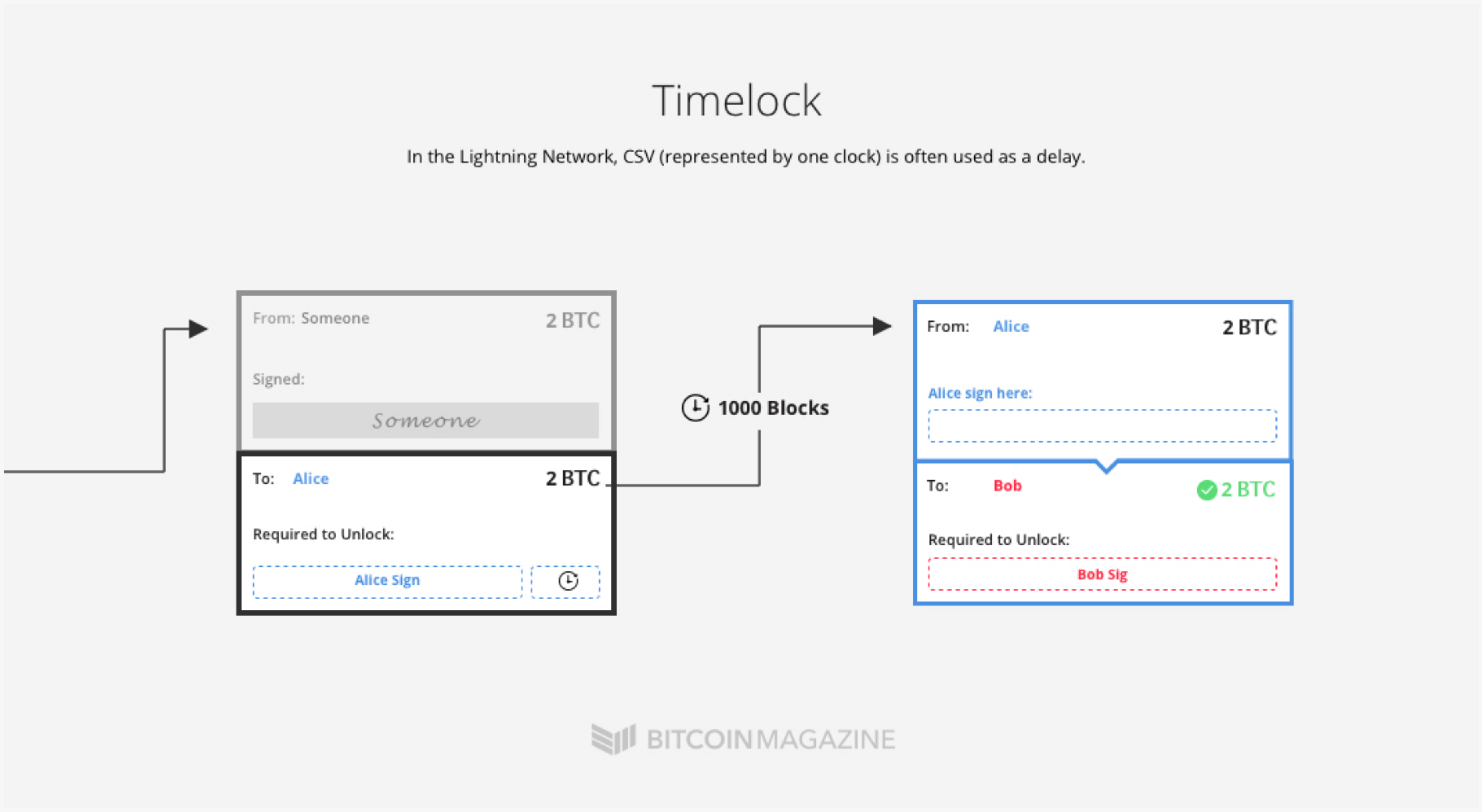
The Lightning Network often uses two out of two (2-of-2) multisig set-ups. Unlocking bitcoins from 2-of-2 multisig addresses requires two signatures, from two dedicated keys.


Multisig
In this example, Alice and Bob previously set up a multisig address to which both hold a key. While Alice tries to spend the bitcoins from this address on her own, she will not succeed without Bob's signature.
At this point, it should also be noted that after a signature has been added to a transaction, it's not possible to change the content of the transaction. In cryptography, a signature is kind of a mixture of a transaction and a seal. (And it's really a long and unique string of numbers.)
From: Someone
2 BTC
Signed:
Someone
To: Alice Bob
2 BTC
Required to unlock:
Alice Sig
Bob Sig
To: Alice Bob
2 BTC
Signed:
Bob
Alice sign here
To: Alice
2 BTC
Required to unlock:
Alice Sig
BITCOIN MAGAZINE

Building Block #4: Time-Locks

The fourth building block is the time-lock. Time-locks can "lock bitcoins up" in an output, to make them spendable (to be included in a subsequent input) only at some point in the future.

There are two different types of time-locks: the absolute type, called CheckLockTimeVerify (CLTV), and the relative type, CheckSequenceVerify (CSV). CLTV locks bitcoins up until a (more or less) concrete time in the future: an actual time and date, or a specific block height. CSV, instead, uses relative time. Once a CVS-output is recorded on the blockchain, it takes a specific amount of blocks from *that* point on before the bitcoins can be spent again.

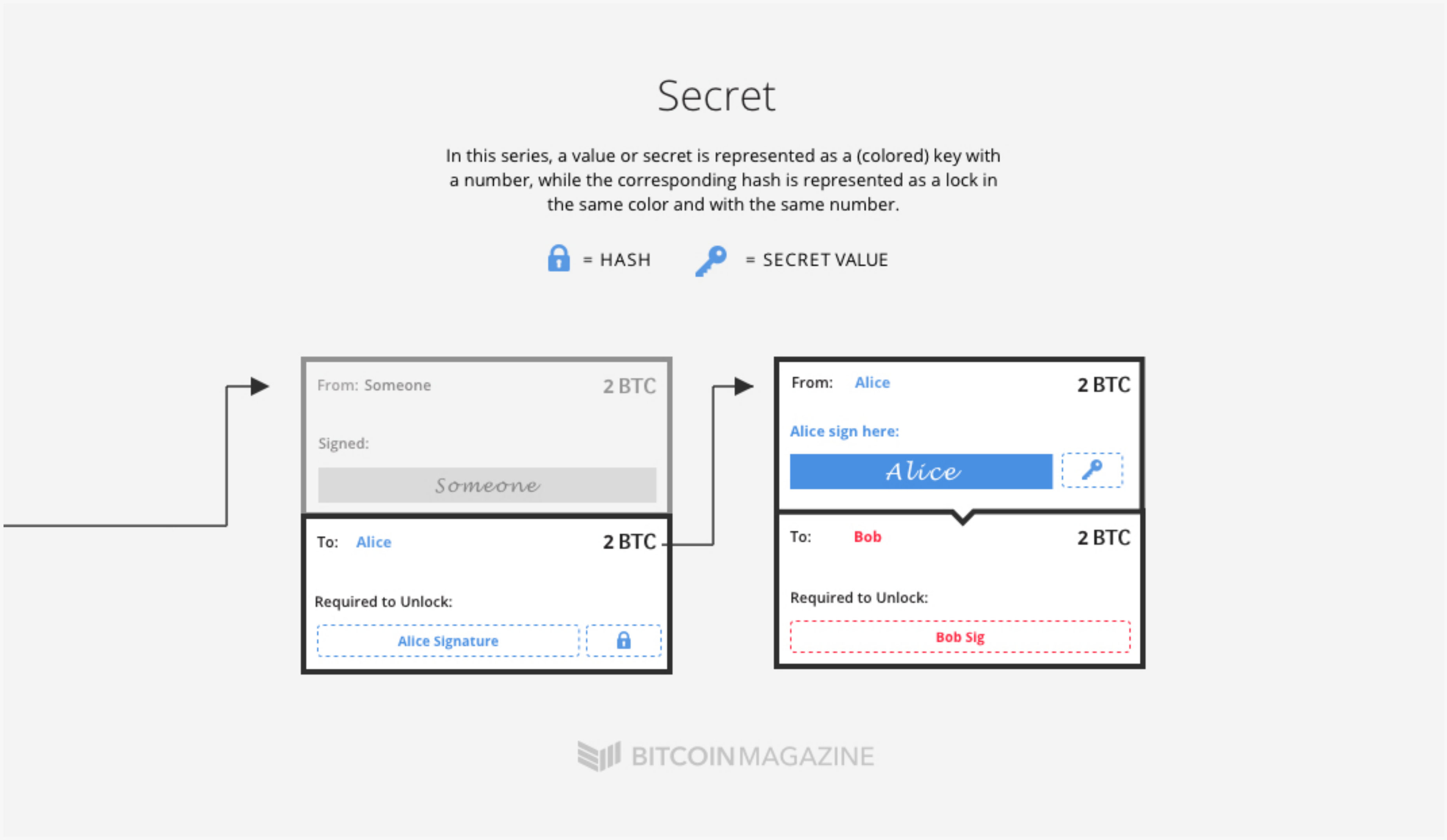


Building Block #5: Hash Values and Secrets

The fifth and final building block – cryptography - is the most fundamental building block of Bitcoin itself. But in the Lightning Network, it's applied in a new way.

In short, a "value" or "secret" is a long and unique string of numbers that is practically impossible to guess, even for a computer with infinite tries. With a special calculation, this value (or secret) can be "hashed" into a different string of numbers, a "hash." And here's the trick: anyone who knows the value can easily reproduce the hash. But this doesn't work the other way around; it's a one way street.

This trick can be utilized in Bitcoin itself, again to “lock bitcoins up.” (In fact, it's really how Bitcoin works.) For example, a hash can be included in an output, and require the subsequent input to include the corresponding value in order to be spendable.



The First Challenge: Bidirectional Payment Channels

Even before the Lightning Network was presented, the concept of payment channels had been around for some time. Typical payment channels are useful for certain purposes, but also limited: they are one-directional. Alice can pay Bob several off-chain transactions, but Bob cannot pay Alice through the same channel at all.

As a key feature of the Lightning Network, Poon and Dryja proposed trustless bidirectional payment channels.

Opening the Channel

To set up a bidirectional payment channel, both parties involved must first agree on an opening transaction. This opening transaction determines how many bitcoins each deposits into the channel.

Let's say Alice wants to send one bitcoin to Bob. Since Alice and Bob expect to transact more frequently, they decide to open up a bidirectional payment channel, and use this to send the bitcoin. (Sending a whole bitcoin is probably a lot for a payment channel, as these might be more useful for micropayments – but it's perfectly possible.)

To open the channel, Alice and Bob each send five bitcoins to a 2-of-2 multisig address. This is the “opening transaction.” Bitcoins can only be spent from this address if both Alice and Bob sign a subsequent transaction.

Additionally, Alice and Bob both create a secret (a string of numbers), and exchange the hash.

Alice now immediately creates a subsequent transaction from the opening transaction. This is a “commitment transaction.” With the commitment transaction, Alice sends four bitcoins to herself, and six bitcoins to a second multisig address. This second multisig address is a bit funky. It can be unlocked by Bob on his own, but only after 1000 extra blocks have been mined after it’s included on the blockchain; it includes a CSV-lock. *Or*, it can be opened by Alice on her own, but only if she *also* includes the secret for which Bob has just now given her the hash. (Of course, Alice has no idea what this secret is – she only knows hash – so there's no way she can make use of this option right now.)

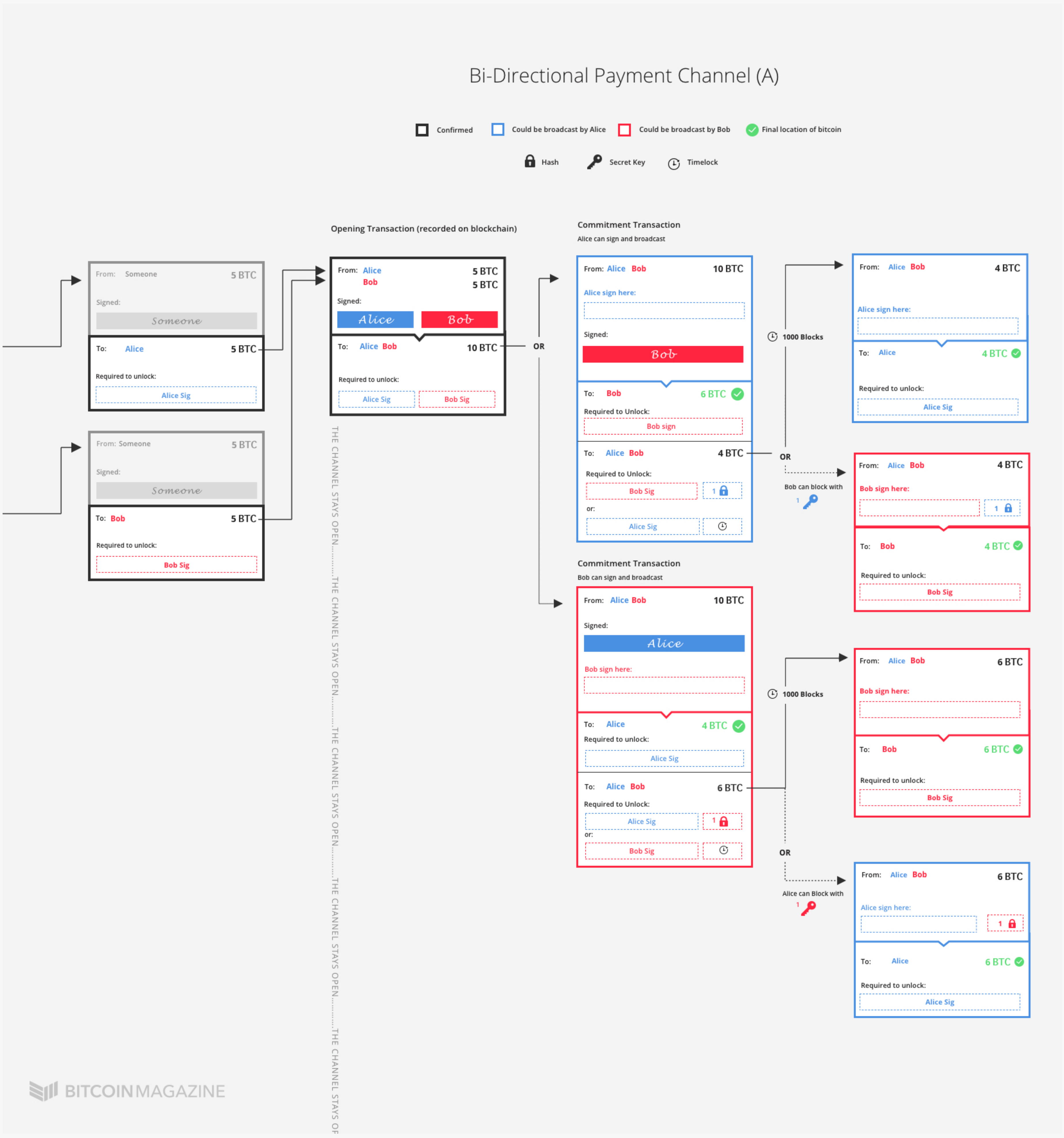
Alice signs her end of this commitment transaction. But she doesn't broadcast it! Instead, she gives it to Bob.

Meanwhile, Bob does the same, but mirrored. He creates a commitment transaction as well, from which he sends six bitcoins to himself, and four to a funky new multisig-address. Alice can unlock this address if she waits an additional 1000 blocks, or Bob can unlock it with Alice using her secret.

Bob signs this half, and gives it to Alice.

After all this exchanging of “half-valid” commitment transactions and hashes of secrets, they both sign and broadcast the opening transaction, to make sure it's recorded on the blockchain. The channel is now officially open.

At this point, both Alice and Bob could sign and broadcast the half-valid commitment transaction they got from the other. If Alice does, Bob gets six bitcoins immediately. If Bob does, Alice gets four bitcoins immediately. But whomever signs and broadcasts the transaction will have to wait 1000 blocks to unlock the subsequent multisig-address, and claim the remaining bitcoins.


Bi-Directional Payment Channel (A)
Confirmed
Could be broadcast by Alice
Could be broadcast by Bob
Final location of bitcoin
Hash
Secret Key
Timelock
Opening Transaction (recorded on blockchain)
From: Someone 5 BTC
Signed: Someone
To: Alice 5 BTC
Required to unlock: Alice Sig
From: Someone 5 BTC
Signed: Someone
To: Bob 5 BTC
Required to unlock: Bob Sig
From: Alice 5 BTC Bob 5 BTC
Signed: Alice Bob
To: Alice Bob 10 BTC
Required to unlock: Alice Sig Bob Sig
OR
THE CHANNEL STAYS OPEN.......THE CHANNEL STAYS OPEN.......THE CHANNEL STAYS OPEN.......THE CHANNEL STAYS OPEN.......THE CHANNEL STAYS OF
Commitment Transaction
Alice can sign and broadcast
From: Alice Bob 10 BTC
Alice sign here:
Signed: Bob
To: Bob 6 BTC
Required to Unlock: Bob sign
To: Alice Bob 4 BTC
Required to Unlock: Bob Sig 1
or: Alice Sig
1000 Blocks
OR
Bob can block with 1
From: Alice Bob 4 BTC
Alice sign here:
To: Alice 4 BTC
Required to unlock: Alice Sig
From: Alice Bob 4 BTC
Bob sign here: 1
To: Bob 4 BTC
Required to unlock: Bob Sig
Commitment Transaction
Bob can sign and broadcast
From: Alice Bob 10 BTC
Signed: Alice
Bob sign here:
To: Alice 4 BTC
Required to unlock: Alice Sig
To: Alice Bob 6 BTC
Required to Unlock: Alice Sig 1
or: Bob Sig
1000 Blocks
OR
Alice can Block with 1
From: Alice Bob 6 BTC
Bob sign here:
To: Bob 6 BTC
Required to unlock: Bob Sig
From: Alice Bob 6 BTC
Alice sign here: 1
To: Alice 6 BTC
Required to unlock: Alice Sig
BITCOIN MAGAZINE

However, and this is the key trick of a payment channel: neither sign and broadcast their half of the transaction at all.

Updating the Channel

A little later, Bob wants to send Alice one bitcoin back. They want to update the channel state, to make the balance five-five again. To accomplish this, Alice and Bob do two things.

First, both repeat the process as described above (except that the opening transaction is already recorded on the blockchain; that part is skipped). This time, both Alice and Bob attribute themselves five bitcoins, and both attribute five bitcoins to funky multisig-addresses. The conditions for these multisig-addresses are similar, except that they require *new* secrets: both Alice and Bob provide each other with *new* hashes. They both sign their new half valid commitment transaction, and give it to each other.

Second, Alice and Bob hand each other their *first* secrets, as used in the *first* set-up.

At this point, again, both Alice and Bob could sign and broadcast the new "half valid" commitment transaction they just got. Their counterparty would get five bitcoins immediately, while the broadcaster would have to wait 1000 blocks. As such, the channel is updated.

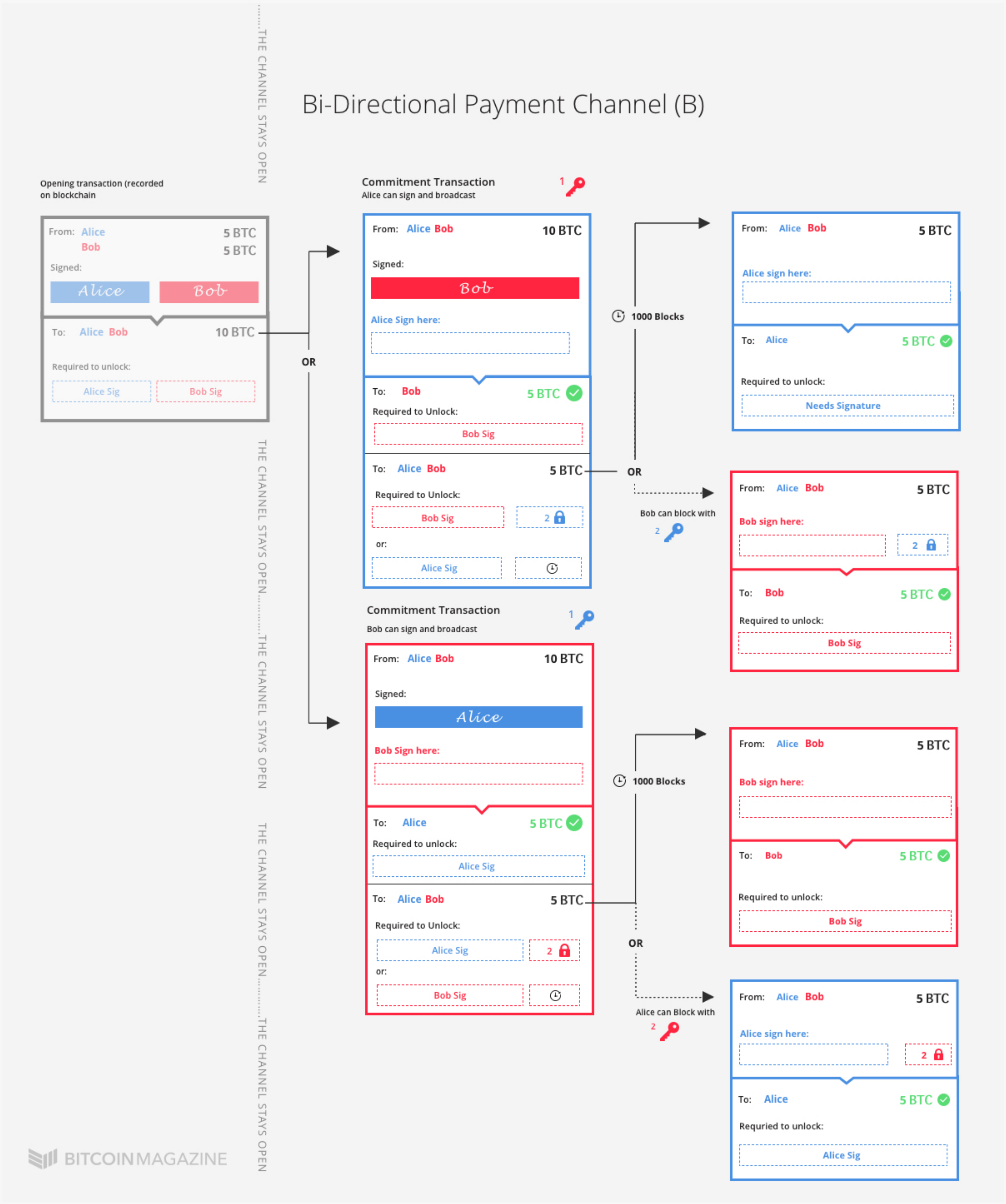
But what's stopping Bob from broadcasting the older commitment transaction instead? That commitment transaction led to a path that paid him six bitcoins, instead of five….

What's stopping Bob, of course, is his first secret, which he has now given to Alice.

Bob cannot safely sign and broadcast the older commitment transaction any more, because Alice now knows Bob's first secret. If Bob were to sign and broadcast that commitment transaction, he would immediately send four bitcoins to Alice... and he would have to wait 1000 blocks to claim his own six bitcoins. That's a problem, because now that Alice knows his secret, she could use this time to beat Bob to the punch, and claim the other six bitcoins as well!

And since Bob has Alice’s secret too, this is just as true for the other way around. If Alice tries to sign and broadcast an old commitment transaction, Bob can steal all the bitcoins in the channel.

This of course means that both Alice and Bob are strongly incentivized to play fair, and only ever sign and broadcast the most recent state of the channel.



Next, this bidirectional payment channel set-up needs to expand to allow payments over a network. This is covered in the second article of this series.

Thanks to Rusty Russell and Joseph Poon for added feedback.

#TUTORIAL #SCALING #BLOCKCHAIN #DOUBLE SPEND #SEGWIT #TIMELOCK #MULTISIG #PAYMENT CHANNELS #LIGHTNING NETWORK #UNCONFIRMED TRANSACTION

RELATED ARTICLES:

Understanding the Lightning Network, Part 2: Creating the Network

by Aaron van Wirdum

Subscribe

Store

Call for Writers

We are always looking for talented writers to join our team. If you have an article you'd like to have published to our audience please reach out to

editor@bitcoinmagazine.com