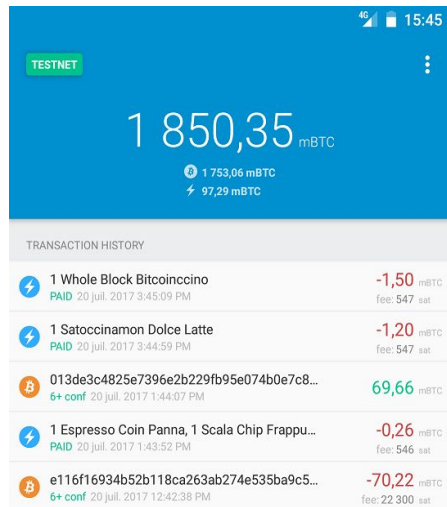


Payment UX

Chaincode LN Residency - NY 2019

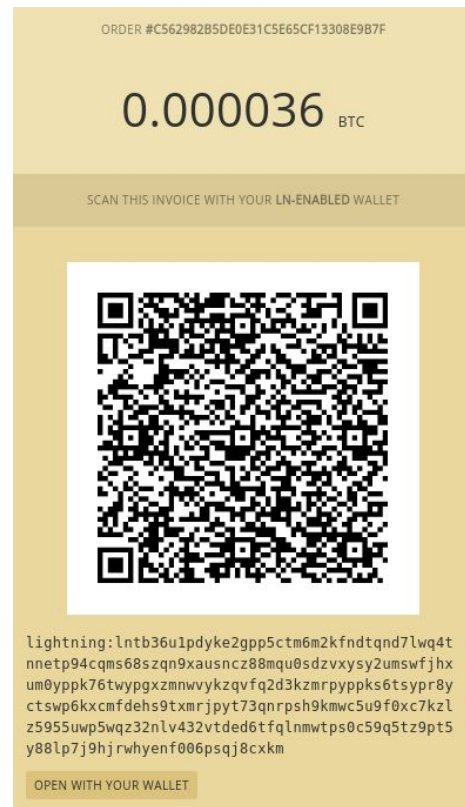
Fabrice Drouin <fabrice.drouin@acing.fr> - <https://acing.co/>

How do Payments work ?

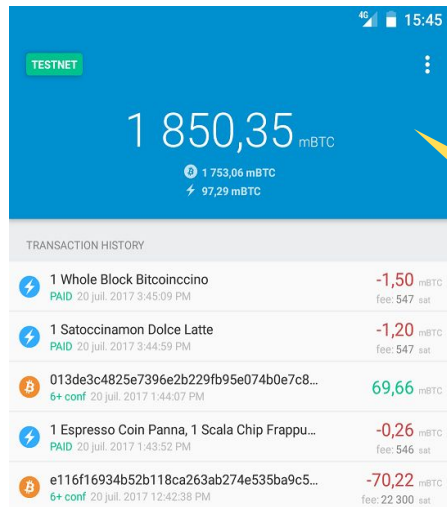


Scan Payment Request

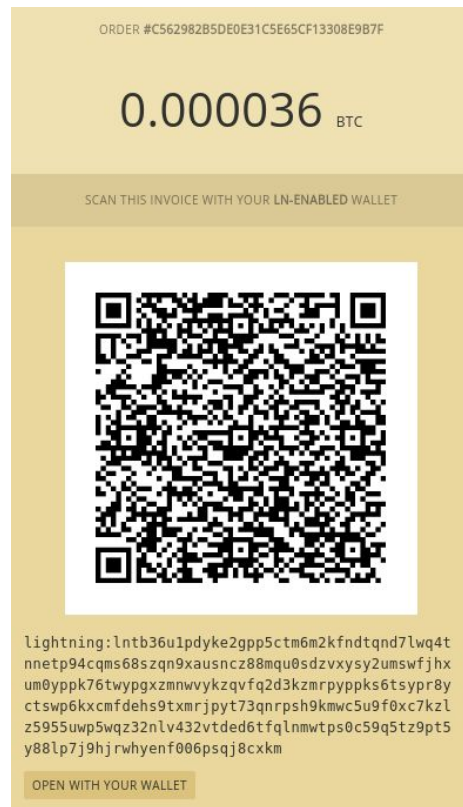
Extract amount, destination, expiry and payment hash



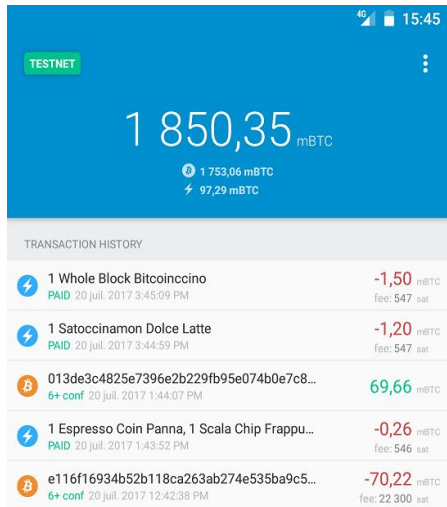
How do Payments work ?



Computes route to destination and creates an onion packet



How do Payments work ?

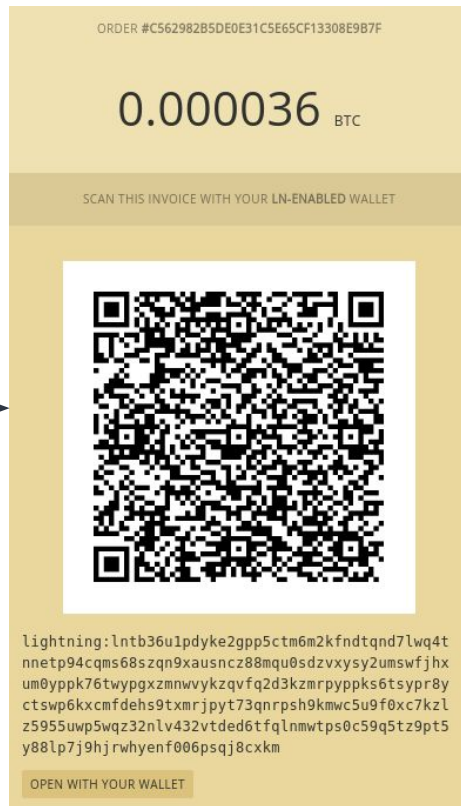


HTLC

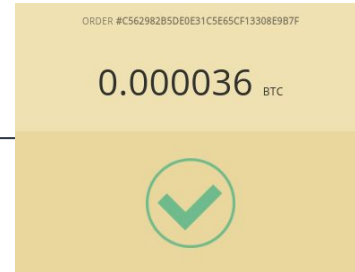
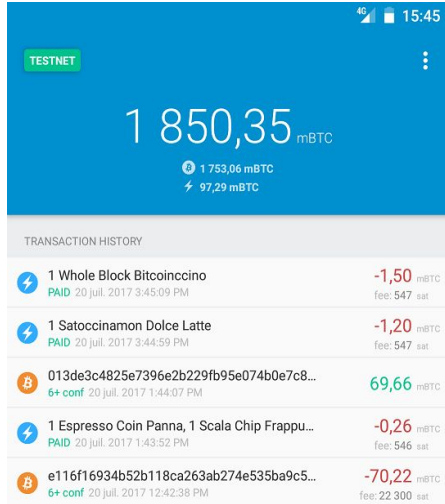
HTLC

HTLC

Payment hash is used as payment identifier



How do Payments work ?



Lightning: The good

- Full ownership
- Privacy minded
- Instant payments
- Fees are proportional to the amount

Lightning: The Bad

- Onboarding and channel creation
- Double balance
- Inbound capacity
- Network sync
- Spontaneous payments
- What if... payment error feedback

Lightning: The Bad: Onboarding

- User is expected to know a lot already:
 - Channel? On-chain?
- Which node do I open a channel with?
 - Autopilot? Lots of nodes are not reliable
 - Ranking system? Which one? Incentive?
 - **Mistakes are costly!**
- I have to wait before my channel is active
 - Delay depends on the on-chain fee
- How much do I put in my channel?

Lightning: The Bad: Onboarding

How do I join the Lightning Network ?


Low-level workflow:

- Must have bitcoins
- Open channel
 - With which nodes .
- Wait for channel to be confirmed
- Use channel.....

How do you start from scratch ?



How do you choose ?



Can I receive too ?



Lightning: The Bad: Double balance

In fact, user must handle 3 balances:

- **On-chain** balance / **off-chain** balance
- Off-chain balance **IN** / off-chain balance **OUT**
- Confusing for the user: are those off-chain funds real Bitcoin?
- “Expert features”
 - UTXO selection
 - fee management
 - bech32/legacy/p2sh-of-p2wpkh addresses
 - Mixers/CoinJoin

Lightning: The Bad: Inbound capacity

- I've painfully opened a channel but I can't receive?
 - All the funds are on my side!
- Liquidity service (ACINQ, Thor, Loop, ...)
- Soon™: Dual funding
 - Onboarding more complex?

Lightning: The Bad: Network sync

- User must have a good view of the whole network
- Device is often offline and must sync before it can pay
- User must also sync his on-chain view to make sure channel is still active

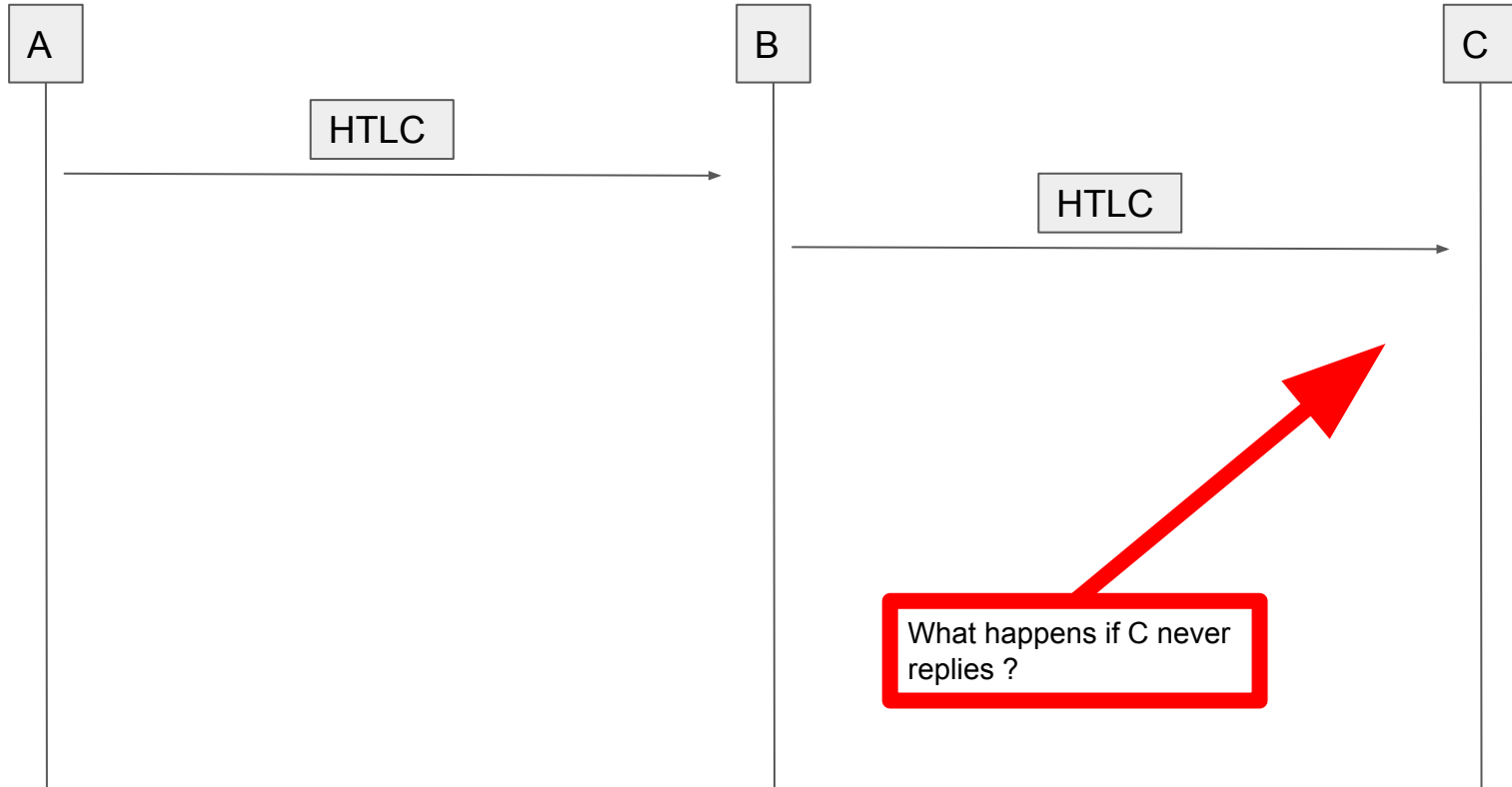
Lightning: The Bad: Spontaneous payments

- Donation use case with static address
 - Sender includes preimage ?
 - Better: Sender computes a preimage that the recipient can guess
- Offline payments (vending machine...)
 - Sender computes a preimage that its node can guess using a “fake” channel id

Lightning: The Bad: Error feedback

- Many possible errors:
 - No route to the payee
 - Route to the payee but insufficient balance
 - Route to the payee but channel disabled (peer offline...)
- But same outcome for the user
 - What should the user do? Open a channel?
- Dangerous to reuse payments!

Lightning: The Bad: Error feedback



Lightning: The Ugly

- Timeouts: my payment is stuck somewhere
 - Who do I yell at? Probably the wallet developer!
 - I can't do anything but wait
 - If a channel close, my payment is redeemed in a separate TX
- Conflict with peer (onchain fees)
 - I have to wait a very long time

Wallet UX

- Expert mode vs hide everything
- Choose UTXO, Coin Selection
- CoinJoin/Mixers
- Address types (legacy, segwit/bech32)
- RBF, CPFP

Wallet UX

- Unify offchain/onchain balances
- Fees
- Confirmation time
- Use 0-conf tx ?
- Closing transactions ?

Onboarding UX

- Entry point ?
- Autopilot:
 - How to bootstrap ?
 - Heuristics ?
- LN node “rankings”
 - Lots of “LN explorers”, each with their own ranking systems.

Spontaneous Payments/ Offline payments

- Invoiceless payments
 - Sender includes preimage if outgoing payment
- Offline payments
 - Recipient offline: how to provide proof of payment
 - Example: vending machine
 - Setup with vendor's node Id
 - Create payment request that include a hint "node Id -> fake channel id"
 - Fake channel id used to compute shared secret with vendor's node
 - Can be reused for invoiceless payments